Ατομική Διπλωματική Εργασία

Μελέτη και Χρήση Αρχιτεκτονικής Ενδιάμεσου Λογισμικού για Υλοποίηση Εφαρμογής Επίγνωσης Πλαισίου (Colloquium Room)

Ανδρέας Κωνσταντίνου

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μάιος 2013

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΥΛΟΠΟΙΗΣΗ COLOQUIUM ROOM ΜΕ ΤΗ ΧΡΗΣΗ ΤΟΥ RSCM

Ανδρέας Κωνσταντίνου

Επιβλέπων Καθηγητής
Γιώργος Παπαδόπουλος

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

Μάιος 2013

Ευχαριστίες

Θα ήθελα καταρχάς να ευχαριστήσω τον επιβλέποντα καθηγητή μου κο Γιώργος Παπαδόπουλος ο οποίος μου ανάθεσε ένα θέμα εξαιρετικά ενδιαφέρον και ο οποίος με κάποιο τρόπο συνέλαβε στη διεύρυνση των γνώσεων μου.

Τέλος θα ήθελα να ευχαριστήσω ιδιαίτερα τους βοηθούς μου Αχιλλέα και Χρήστο για την πολύτιμη βοήθεια τους και την άμεση ανταπόκριση τους σε οπουδήποτε πρόβλημα αντιμετώπισα.

ΠΕΡΙΛΗΨΗ

Σκοπός αυτής της διπλωματικής εργασίας, είναι η παρουσίαση μιας πλαισίου εφαρμογής(context aware), βασισμένη σε ένα είδους ενδιάμεσου λογισμικού (middleware), το RSCM(Really Simple Context Middleware).Περιγράφοντας τα στάδια ανάπτυξης της εφαρμογής, σε πλατφόρμα Android, κύριο μέλημα είναι πώς η χρήση του RSCM(Really Simple Context Middleware), απλοποιεί και βοηθάει την ανάπτυξη μίας εφαρμογής πλαισίου (context-aware).Παράλληλα, εξετάζοντας την αρχιτεκτονική του ενδιάμεσου λογισμικού(middleware), και πιο συγκεκριμένα του RSCM, έχει ως σκοπό η εφαρμογή μέσα από τα στάδια ανάπτυξης της, να τονίσει τα οφέλη και τις οποιεσδήποτε ευκολίες που είναι σε θέση να μας παρέχει το RSCM όχι μόνο προγραμματιστικά (σε μορφή κώδικα), αλλά και εκμαιεύοντας τη πλήρη εφαρμογή του ως κύριο συστατικό σε υλοποίηση μιας Android εφαρμογής. Επιπρόσθετα, μέσα από τη υλοποίηση του πλαισίου εφαρμογής (context aware), κύριο μέλημα ήταν να τεθούν επιχειρήματα για το λόγο που ένας οποιοσδήποτε προγραμματιστής, θα μπορούσε να διαλέξει αυτού του είδους ενδιάμεσου λογισμικού(middleware) για να υλοποιήσει τη δική του εφαρμογή πλαισίου (context-aware).

Περιεχόμενα

Κεφάλαιο 1

Εισαγωγή……..1-2

1.1 Βασικό Υπόβαθρο……..……2

1.2 Περίγραμμα Εργασίας…….1

Κεφάλαιο 2

Πλαίσιο(Context)……3-5

2.1 Ορισμοί…….3

2.2 Εξέλιξη…….….4

2.3. Προκλήσεις Context……4

Κεφάλαιο 3

Android…….. ……..6-12

3.1 Ορισμός…….6

3.2 Αρχιτεκτονική Android…….6

3.3 Εκδόσεις Android……8

3.4 Κατασκευαστές συσκευών Android……..10

3.5 Χαρακτηριστικά πλατφορμών Android……...10

3.6 Android SDK-IDE-JAVA……11

Κεφάλαιο 4

Ενδιάμεσο Λογισμικό(Middleware)…….……..13-20

4.1 Ορισμός……...13
4.2 Ο Ρόλος του ενδιάμεσου Λογισμικού (Middleware)……..14
4.3 Πλεονεκτήματα και Μειονεκτήματα Middleware……..14
4.4 Προέλευση……. ..15
4.5 Χαρακτηριστικά Ενδιάμεσου Λογισμικού (Middleware)…….. 16
4.6 Τύποι Ενδιάμεσου Λογισμικού (Middleware)…….17
4.7 Ορισμός Osgi…….17
4.8 Αρχιτεκτονική και υπηρεσίες Osgi……19
4.9 IST-MUSIC(Middleware)……..19

Κεφάλαιο 5

RSCM (Really Simple Context Middleware) ……21-24

5.1 Ορισμός……21
5.2 Κατηγοριοποίηση του RSCM……22
5.3 Κύρια περιεχόμενα του RSCM…….23

Κεφάλαιο 6

Υλοποίηση της εφαρμογής μας με χρήση RSCM ……25-35

6.1 Γενική αναφορά στη εφαρμογή ……..25
6.2 Context plug-ins βασισμένα στη λειτουργία του RSCM……..26

6.3 Golloqioum room, η εφαρμογή μας βασισμένη στο RSCM…28
6.4 Δυσκολίες κατά την υλοποίηση της εφαρμογής ……..34

Κεφάλαιο 7

Υλοποίηση της εφαρμογής μέσω διπροσωπίας για το χρήστη……36-42

7.1 Διπροσωπία χρήστη μέσω android εφαρμογών……..36
7.2 Σκοπός χρήσης battery plug in…….40
7.3 Σκοπός χρήσης battery plug in…….41

ΚΕΦΑΛΑΙΟ 8 Συμπεράσματα και μελλοντικές επεκτάσεις…….43-45

8.1 Συμπεράσματα……43
8.2 Μελλοντικές Προεκτάσεις……..44

Β ι β λ ι ο γ ρ α φ ί α …… 46

Παράρτημα Α : **Κώδικας του main activity και προεκτάσεις εφαρμογής…Α1-14**

Παράρτημα Β : Κώδικας Battey plug-in…….. Β1-2

Παράρτημα Γ: Κώδικας Wi Fi plug in…….....Γ 1-2

Εισαγωγή

1.1 Βασικό Υπόβαθρο

1.2 Περίγραμμα Εργασίας

1.1 Βασικό Υπόβαθρο

Στη σημερινή εποχή, είναι πλέον δεδομένο ότι οι ηλεκτρονικοί υπολογιστές και οι ηλεκτρονικές συσκευές καταλαμβάνουν ένα μεγάλο μέρος της ζωής του ανθρώπου, καθιστώντας τη χρήση τους ως ένα είδος ρουτίνας στη καθημερινότητα μας. Έχοντας αυτό ως δεδομένο, η ανάπτυξη εφαρμογών επίγνωσης πλαισίου δίνει νέες δυνατότητες και προοπτικές για διευκόλυνση του χρήστη στην εκτέλεση καθημερινών εργασιών. Το μεγάλο ενδιαφέρον και ζήτηση που υπάρχει στη αγορά, καθώς και ο ανταγωνισμός καθιστούν ως αναγκαιότητα την ανάπτυξη όλο και περισσότερων εφαρμογών, οι οποίες είναι σε θέση να αντιλαμβάνονται ανά πάσα στιγμή και να χρησιμοποιούν πληροφορίες σχετικά με το περιβάλλον που αλληλεπιδρά ο χρήστης, το προφίλ του χρήστη καθώς και οποιεσδήποτε άλλες πληροφορίες.

1.2 Περίγραμμα Εργασίας

Αυτή η διπλωματική παρουσιάζει την περιοχή των συστημάτων επίγνωσης πλαισίου (context awareness), καθώς και το ενδιάμεσο λογισμικό (middleware) RSCM(Really Simple Context Middleware) και την αρχιτεκτονική του το οποίο χρησιμεύει στην υλοποίηση εφαρμογών επίγνωσης πλαισίου (context aware). Αυτή του είδους αρχιτεκτονική στην οποία είναι βασισμένη η εφαρμογή που έχει υλοποιηθεί στα

πλαίσια της διπλωματικής έχει ως σκοπό να αποδείξει τα διάφορα οφέλη και τις ευκολίες που παρέχονται στους προγραμματιστές που ασχολούνται με την ανάπτυξη εφαρμογών επίγνωσης πλαισίου.

Κεφάλαιο 2

ΠΛΑΙΣΙΟ (CONTEXT)

2.1 Ορισμοί
2.2 Εξέλιξη
2.3. Προκλήσεις του Context

2.1. Ορισμοί

Το πλαίσιο (context) αποτελεί ένα από τα σημαντικότερα σημεία αναφοράς, στην ανάπτυξη διάφορων εφαρμογών πληροφοριακών συστημάτων, και ειδικότερα στην υλοποίηση εφαρμογών για κινητές συσκευών. Το πλαίσιο (context) μπορεί να χρησιμοποιηθεί για την αλληλεπίδραση του χρήστη με την εφαρμογή, ή να παρέχει γενικές υπηρεσίες και πληροφορίες στο χρήστη. Η εφαρμογή του πλαισίου (context) γίνεται ακόμη πιο σημαντική, καθώς οι ηλεκτρονικοί υπολογιστές δεν είναι σε θέση να παρέχουν πληροφορίες, ανάλογα με το προφίλ, τις συνθήκες(π.χ. χρόνος, τόπος) και τις δραστηριότητες του χρήστη. Η ουσία του πλαισίου (context) επεκτείνεται ιδιαίτερα, στις φορητές συσκευές, όπου δεδομένα όπως π.χ., η τοποθεσία, εργασία που επιτελεί(π.χ. στο γραφείο, σε συνάντηση) και το περιβάλλον χρήσης της εφαρμογής αλλάζουν δυναμικά.

Σημαντικό για την κατανόηση και το ρόλο των εφαρμογών επίγνωσης πλαισίου (context aware) είναι να δώσουμε ένα ορισμό για το πλαίσιο (context) και για όλες τις εφαρμογές του, καθώς και για το πώς μπορεί να χρησιμοποιηθεί.
Πλαίσιο (context) είναι οποιαδήποτε πληροφορία, η οποία χρησιμοποιείται για να προσδιορίσει μια κατάσταση μιας οντότητας. Μια οντότητα μπορεί να είναι ένα άτομο, μια περιοχή (location) ή ακόμη και ένα αντικείμενο, τα οποία πρέπει να είναι άμεσα συσχετιζόμενα όσο αφορά την αλληλεπίδραση μεταξύ του χρήστη και μιας εφαρμογής,

συμπεριλαμβανομένου και του ίδιου του χρήστη αλλά και της ίδιας της εφαρμογής. Σύστημα μπορεί να χαρακτηρισθεί ως σύστημα επίγνωσης πλαισίου, εάν χρησιμοποιεί το πλαίσιο για να παρέχει σχετικές πληροφορίες ή υπηρεσίες στο χρήστη, ανάλογα με την εφαρμογή .Ένα σύστημα σύστημα επίγνωσης πλαισίου, που χρησιμοποιεί την θέση (location) του χρήστη σαν παράμετρο (contextual parameter), είναι σε θέση να αναγνωρίσει το που βρίσκεται ο χρήστης.

Για παράδειγμα ένα τέτοιο σύστημα είναι σε θέση να ξέρει το ότι ο χρήστης βρίσκεται σε μία διάσκεψη, δηλαδή έχει επικεντρωθεί περισσότερο στις πληροφορίες σχετικά με την επίγνωση τοποθεσίας του χρήστη κάποιας χρονικής στιγμής.

2.2 Εξέλιξη

Για ένα μεγάλο χρονικό διάστημα, η εγκατάσταση μίας εφαρμογής επίγνωσης πλαισίου (context aware) ήταν δύσκολη και περίπλοκη διότι δεν υπήρχαν οι κατάλληλες συσκευές για τη υποστήριξη της. Με τα σημερινά δεδομένα, σχεδόν όλες οι φορητές συσκευές έχουν ενσωματωμένους αισθητήρες και επικοινωνιακά συστήματα, τα οποία μπορούν να παρέχουν πληροφορίες σχετικά με το περιβάλλον που αλληλεπιδρά ο χρήστης, με τέτοιο τρόπο ώστε να αποτελούν δεδομένα εισόδου για τις υπό ανάπτυξη εφαρμογές. Επιπλέον μια ενδιαφέρον εξέλιξη που έχει μία εφαρμογή επίγνωσης πλαισίου (context aware) τα τελευταία χρόνια είναι ότι μπορούν να κάνουν πάντοτε τις σωστές υποθέσεις για τη τοποθεσία και δραστηριότητα του χρήστη σε μια συγκεκριμένη στιγμή.

2.3 Προκλήσεις του Context

Ενδιαφέρον προκαλούν οι διάφορες προκλήσεις και ευκαιρίες που υπάρχουν για την ανάπτυξη μίας context-aware εφαρμογής.

Είναι γεγονός ότι πολλοί χρήστες έξυπνων κινητών συσκευών μπαίνουν στο πειρασμό να διαβάζουν την δουλειά που έχουν σε ώρες αναψυχής Για παράδειγμα τα email και

τα sms messages από τους συναδέλφους και προϊσταμένους έρχονται όταν η τοποθεσία του χρήστη έλθει και πάλι στη κατάσταση “εργασίας” (δηλαδή ότι ο χρήστης βρίσκεται στη εργασία του), έτσι ώστε να αποφεύγεται ο πειρασμός αυτός.

Ένα είδος πρόκλησης που αντιμετωπίζει μια context aware εφαρμογή είναι η έλλειψη επαναχρησιμοποίησης ενός μηχανισμού επίγνωσης πλαισίου, καθώς οποιαδήποτε εφαρμογή απαιτεί και χρειάζεται ένα μηχανισμό για να είναι σε θέση να υποστήριξη διάφορες συγκεντρώσεις πληροφοριών. Επιπρόσθετα μία άλλη πρόκληση που αξίζει να αναφέρουμε είναι η μυστικότητα (privacy). Είναι λογικό οποιοσδήποτε χρήστης να ανησυχεί και να έχει αμφιβολίες αν τα προσωπικά του δεδομένα είναι εξασφαλισμένα με τη χρήση αυτού του είδους πληροφοριακών συστημάτων. Μία εφαρμογή επίγνωσης πλαισίου (context aware) είναι σε θέση με τη κατασκευή της δομής της να εξασφαλίζει σε ένα μεγάλο ποσοστό αυτή τη μυστικότητα δεδομένων που επιθυμεί να έχει ο κάθε χρήστης για τη εφαρμογή που θέλει να χρησιμοποιήσει.

Κεφάλαιο 3

ANDROID

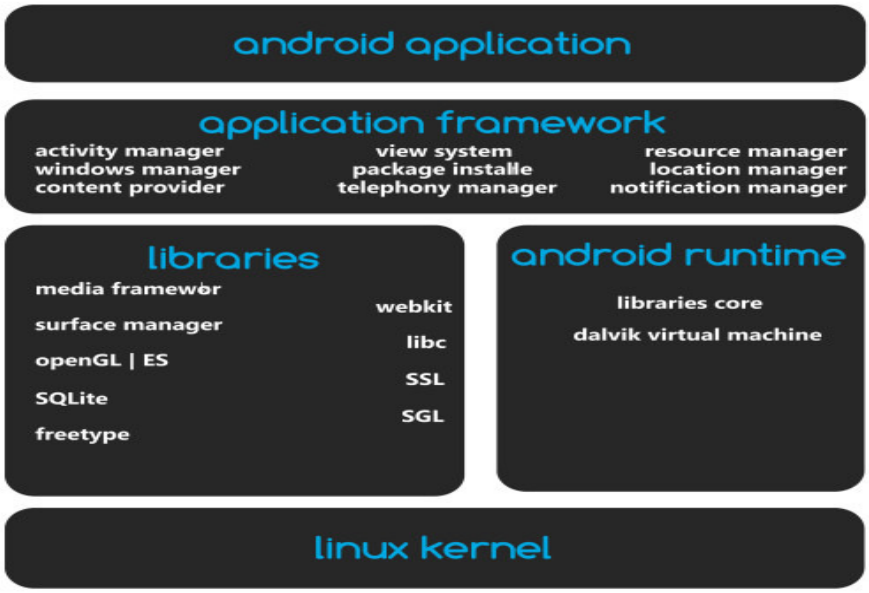
3.1 Ορισμός
3.2 Αρχιτεκτονική Android
3.3 Εκδόσεις Android
3.4 Κατασκευαστές συσκευών Android
3.5 Χαρακτηριστικά πλατφόρμων Android
3.6 Android SDK-IDE-JAVA

3.1 Ορισμός

Το Android είναι μια στοίβα λογισμικού για κινητές συσκευές η οποία περιλαμβάνει λειτουργικό σύστημα, ενδιάμεσο λογισμικό (middleware) και βασικές εφαρμογές. Το Android τρέχει στον πυρήνα του λειτουργικού Linux και μέσω της δικιάς του εργαλειοθήκης ανάπτυξης συστήματος λογισμικού (Software Development Kit), επιτρέπει στους κατασκευαστές να δημιουργούν πρωτοποριακές εφαρμογές. Αρχικά αναπτύχθηκε από την Google και αργότερα συνεχίστηκε σε συνεργασία με την Open Handset Alliance (OHA). Η πρώτη παρουσίαση της πλατφόρμας Android έγινε στις 5 Νοεμβρίου 2007, παράλληλα με την ανακοίνωση της ίδρυσης του οργανισμού OHA, μιας κοινοπραξίας 48 τηλεπικοινωνιακών εταιριών, εταιριών λογισμικού καθώς και κατασκευής υλικού, οι οποίες είναι αφιερωμένες στην ανάπτυξη και εξέλιξη ανοιχτών προτύπων στις συσκευές ανοιχτής τηλεφωνίας.

3.2 Αρχιτεκτονική Android

Όπως προανέφερα, το Android είναι μια στοίβα λογισμικού. Η λογική πίσω από αυτήν την έκφραση και σε όλη την φιλοσοφία του Android, κρύβεται στο ακόλουθο διάγραμμα με τα βασικά συστατικά του (Σχήμα 3.1).



Σχήμα 3.1(Δομή της Αρχιτεκτονικής Android)

Αποτελείται από :

Linux Kernel: Αυτός είναι ο Kernel (ο πυρήνας του λειτουργικού) στον οποίο βασίζεται το Android και βρίσκεται στο χαμηλότερο επίπεδο. Παρέχει τους Drivers τους οποίους χρειάζεται για να τρέξει το σύστημα, όπως της οθόνης, της κάμερας κ.α. Μια παρομοίωση για να γίνει ποιο κατανοητό, αν το Android είναι ένα Robot, τότε ο Kernel είναι ο σκελετός του.

Libraries: Οι βιβλιοθήκες (Libraries) περιέχουν όλο τον κώδικα που περιέχει το Android OS. Παραδείγματος χάριν, η SQLite βιβλιοθήκη παρέχει υποστήριξη έτσι ώστε μια εφαρμογή να χρησιμοποιήσει την αποθήκευση δεδομένων, η Webkit βιβλιοθήκη παρέχει λειτουργίες για το διαδικτυακό σερφάρισμα.

Application Framework: Εκθέτει διάφορες δυνατότητες του Android στους προγραμματιστές των εφαρμογών ώστε να τις χρησιμοποιήσουνε στις εφαρμογές τους.

Android OS: Παραδείγματος χάριν, η SQLite βιβλιοθήκη παρέχει υποστήριξη έτσι ώστε μια εφαρμογή να χρησιμοποιήσει την αποθήκευση δεδομένων, η Weskit βιβλιοθήκη παρέχει λειτουργίες για το διαδικτυακό σερφάρισμα.

Android Runtime: Στο ίδιο επίπεδο με τις βιβλιοθήκες, το Android Runtime παρέχει ένα σύνολο βασικών βιβλιοθηκών που επιτρέπουν στους προγραμματιστές να γράψουν εφαρμογές χρησιμοποιώντας JAVA.

Επίσης περιλαμβάνει την Dalvik virtual μηχανή, που επιτρέπει κάθε εφαρμογή να τρέξει την δικιά της εργασία, μαζί με την δικιά της ξεχωριστή Dalvik virtual μηχανή.

Applications: Το πιο υψηλό επίπεδο, εδώ βρίσκουμε εφαρμογές που έρχονται μαζί με την Android συσκευή σου (όπως τηλέφωνο, επαφές, μουσική κ.α.), όπως επίσης εφαρμογές που κάνει ο χρήστης εγκατάσταση. Οποιαδήποτε εφαρμογή που έχει εγκατεστημένη είναι σε αυτό το επίπεδο.

3.3 Εκδόσεις Android

Η πρώτη έκδοση του Android SDK, τον Νοέμβριο του 2007, χαρακτηρίστηκε από τους κατασκευαστές του σαν μια πρώτη ματιά στο SDK του Android, κάτι το οποίο πολλοί παράβλεψαν και βιάστηκαν να κατακρίνουν το Android σαν ένα προβληματικό σύστημα. Στην ουσία όμως το Android δεν παρουσίαζε προβλήματα τα οποία δεν παρουσιάζει οποιοδήποτε σύστημα σε τέτοια πρώιμη φάση.

Το Σεπτέμβριο του 2008, η T-Mobile ανακοινώνει την διαθεσιμότητα του T-Mobile G1, του πρώτου έξυπνου τηλεφώνου (smartphone), βασισμένο στην πλατφόρμα του Android.Λίγες μέρες αργότερα (Οκτώβριο 2008), η Google ανακοινώνει την απελευθέρωση του SDK Release Candidate 1.0. Ακολούθησε τον Φεβρουάριο του 2009 η έκδοση 1.1 σαν μια ανανεωμένη έκδοση του 1.0. Μέχρι τότε το Android δεν

υποστήριζε ακόμη την χρήση κουμπιών αφής, παρά μόνο την χρήση των κλασσικών ‘σκληρών’ κουμπιών της συσκευής.

Το 2009 έχουμε την έκδοση Android 1.5, τη λεγόμενη ‘Cupcake’.Τα κύρια χαρακτηριστικά αυτής της έκδοσης ήταν η ικανότητα υποστήριξης Bluetooth και οι κινούμενες μεταβάσεις οθόνης. Μετά έκανε την εμφάνιση του το Android 1.6 το “Donut”, όπου παρείχε ένα βελτιωμένο Android market, ενσωματωμένη μηχανή και βιντεοκάμερα, και υπήρχαν βελτιώσεις στη ταχύτητα. Ακολουθεί το ‘Eclair’, Android 2.0 . Ανάμεσα στις άλλες αλλαγές που έφερε αυτή η έκδοση, ήταν και η βελτιωμένη διεπιφάνεια χρήστη, καινούργιες λίστες επαφών, υποστήριξη Microsoft Exchange και υποστήριξη για περισσότερες οθόνες και αναλύσεις. Ακολουθεί το Android 2.2 με το όνομα ‘Froyo’. Η έκδοση ‘Froyo’ ανάμεσα σε άλλες αλλαγές περιλαμβάνει βελτιστοποιήσεις στην ταχύτητα γενικά του λειτουργικού συστήματος, στην μνήμη και στην απόδοση, γρήγορη εναλλαγή ανάμεσα σε πολλαπλές γλώσσες του πληκτρολογίου και των λεξικών τους και πρόσδεση USB και λειτουργία δυναμικής ζώνης (hotspot) WiFi. Ακολουθεί η έκδοση Android 2.3 με το όνομα “Gingerbread” με την επανέκδοση του σε Android 2.3.3 τον Φεβρουάριο του 2011, όπου παρουσιάζεται Βελτιωμένο UI για απλότητα και ταχύτητα**,** βελτιωμένη ενεργειακή διαχείριση και υποστήριξη video κλήσης.

Λίγους μήνες μετά έκανε την εμφάνιση της η έκδοση Android 3.0 με το όνομα “Honeycomb”, όπου το στοιχείο που την ξεχώριζε είναι ότι υποστήριζε διπύρηνους και τετραπύρηνους επεξεργαστές. Στη συνέχεια έχουμε Android 3.1 “Honeycomb”(API 12 LEVEL) και η αναβάθμιση του σε Android 3.2 (API 13LEVEL). Νέα έκδοση Android 4.0 IceCream Sandwich”, το οποίο αποτελεί την προσπάθεια της εταιρίας για ενιαίο λειτουργικό σύστημα για όλες τις συσκευές. Έκδοση με ψηλές προσδοκίες όπως χαρακτηρίστηκε το 2012 ήταν έκδοση android 4.1 η λεγόμενη Jelly Bean(Api 16 level), η οποία παρέχει βελτίωση της εφαρμογής κάμερας ,ψηλή ανάλυση φωτογραφιών ,εφαρμογή του application Google Search, καθώς και χρήση usb audio. Τέλος η τρέχουσα έκδοση για κινητά smartphones, από τον Νοέμβριο του 2012 μέχρι και την στιγμή που γράφεται αυτή η εργασία, είναι η έκδοση Android 4.2 με το όνομα Jelly Bean(Api 17 level), η οποία παρέχει πολλαπλή χρήση λογαριασμών χρηστών και υποστήριξης εμφάνισης wireless.

3.4 Κατασκευαστές συσκευών Android

Τον Οκτώβριο του 2008 κυκλοφόρησε η πρώτη συσκευή Android, την οποία κατασκεύασε η HTC και παρείχε τις απαραίτητες υπηρεσίες τηλεφωνίας η T-Mobile ενώ ονομαζόταν T-Mobile G1. Από εκεί και πέρα πολλές άλλες συσκευές Android ακολούθησαν στα τέλη του 2009 και στις αρχές του 2010 με καλπάζουσα ανάπτυξη και φήμη. Μέσα στους επόμενους 18 μήνες, σε όλο τον κόσμο έκαναν το ντεμπούτο τους από 21 διαφορετικούς κατασκευαστές, 60 διαφορετικές συσκευές Android σε 59 φορείς και σε 48 χώρες. Τον Ιούνιο του 2010 η Google ανακοίνωσε ότι κάθε μέρα ενεργοποιούνται σε ένα σύνολο 60 εκατομμυρίων συσκευών ετησίως περισσότερες από 160.000 συσκευές Android.

Τα πλεονεκτήματα της προσπάθειας των φορέων φαίνεται να βρίσκουν έδαφος και η πλατφόρμα Android ξεχωρίζει. Έχει μπει για τα καλά στην αγορά των mobile phones και ανταγωνίζεται γερά πλατφόρμες όπως Windows Mobile, RIM BlackBerry και iPhone. Αν ανατρέξει κανείς στους αριθμούς το καλοκαίρι του 2010 διαπιστώνει ότι αν και με ποσοστό 31%, το BlackBerry βρίσκεται στην κορυφή των smartphone και το Apple iPhone με ποσοστό 28 % ενώ το Android με ποσοστό 19% στην αγορά, τα δύο πρώτα είναι συνεχώς μειούμενα ενώ το τελευταίο αυξάνεται ταχύτατα. Αξιοσημείωτο είναι το γεγονός ότι το Microsoft Windows βρίσκεται πίσω από το Android.

3.5 Χαρακτηριστικά πλατφορμών Android

Το Android είναι μία ολοκληρωμένη, ανοιχτή και δωρεάν πλατφόρμα κινητών τηλεφώνων.

1) Ολοκληρωμένη: Οι σχεδιαστές βασίστηκαν σ' ένα ασφαλές λειτουργικό σύστημα και κατασκεύασαν ένα δυνατό πλαίσιο λογισμικού το οποίο έχει το πλεονέκτημα την ποικίλα ανάπτυξη εφαρμογών.

2) Ανοιχτή: Μέσω της διαδικασίας ανοιχτής πηγής παρέχεται η πλατφόρμα Android. Είναι ανεμπόδιστη η πρόσβαση στα χαρακτηριστικά των συσκευών όταν οι προγραμματιστές αναπτύσσουν τις εφαρμογές τους.

3) Δωρεάν: Το υποκείμενο λειτουργικό σύστημα του Android έχει κατοχυρωθεί με την άδεια δημόσιας χρήσης GNU General Public License Version 2 (GPLv2), μια ισχυρή άδεια που υποχρεώνει τις βελτιώσεις τρίτων να εξακολουθούν να εμπίπτουν στους όρους των προτύπων ανοιχτής πηγής. Δεν καταβάλλεται κανένα χρηματικό ποσό για την ανάπτυξη εφαρμογών Android, για άδειες χρήσης, για πνευματικά δικαιώματα, για να γίνει κάποιος μέλος της κοινότητας, για τις δοκιμές, για υποτροφίες και για πιστοποιήσεις.

3.6 Android SDK-IDE-JAVA

Οι προγραμματιστές μπορούν να έχουν το Android SDK απ' το website του Android αφού διατίθενται δωρεάν και εφόσον συμφωνήσουν με τους όρους της συμφωνίας άδειας χρήσης του πακέτου ανάπτυξης του λογισμικού του Android. Αν και οι προγραμματιστές έχουν περιθώρια στην επιλογή του ολοκληρωμένου περιβάλλοντος IDE, το πιο δημοφιλές και αυτό που επιλέγουν οι περισσότεροι για την σχεδίαση και την ανάπτυξη εφαρμογών είναι το Eclipse που διατίθεται δωρεάν.

Οι εφαρμογές Android μπορούν να αναπτυχθούν στα εξής λειτουργικά συστήματα:

- Windows XP (32-bit) ή Vista (32-bit ή 64-bit)

- Mac OS X 10.5.8 ή επόμενη έκδοση (μόνο για x86)

- Linux (δοκιμασμένο Linux Ubuntu 8.04 LTS, Hardy Heron)

Η γλώσσα προγραμματισμού που χρησιμοποιείται για την ανάπτυξη εφαρμογών Android είναι η Java. Υπάρχουν όμως σκέψεις για την προσθήκη και άλλων γλωσσών προγραμματισμού, όπως για παράδειγμα η C++ ,σε μετέπειτα εκδόσεις του Android. Πάντως υπάρχει η δυνατότητα με την βοήθεια του εσωτερικού πακέτου ανάπτυξης NDK(Native Development Kit) του Android, η χρησιμοποίηση μιας εφαρμογής που βασίζεται σε κώδικα που γράφεται σε άλλη γλώσσα από την java, όπως C ή C++.

.

Κεφάλαιο 4

Ενδιάμεσο Λογισμικό (Middleware)

4.1 Ορισμός
4.2 Ο Ρόλος του Mmiddleware
4.3 Πλεονεκτήματα και Μειονεκτήματα Middleware
4.4 Προέλευση
4.5 Χαρακτηριστικά Middleware
4.6 Τύποι middleware
4.7 Ορισμός Osgi
4.8 Αρχιτεκτονική και υπηρεσίες Osgi
4.9 IST-MUSIC(Middleware)

4.1 Ορισμός

Στα πληροφορικά συστήματα με το όρο ενδιάμεσο λογισμικό (middleware) εννοούμε το λογισμικό επίπεδο που βρίσκεται ανάμεσα στο λειτουργικό σύστημα και στις εφαρμογές , αλληλεπιδρώντας με αυτά τα επίπεδα. .

Υπάρχουν πολλές εφαρμογές για τη χρησιμοποίηση του ενδιάμεσου λογισμικού (middleware) όπως για παράδειγμα η δυνατότητα επαναχρησιμοποίησης ενός κομματιού κώδικα. Άλλη μία χρήση του ενδιάμεσου λογισμικού είναι στη χρησιμοποίηση στα συστήματα διαμεσολάβησης όπου περίπλοκα συστήματα με πολλαπλές συσκευές συνδέονται με το διαδίκτυο. Μια τελευταία εφαρμογή του

ενδιάμεσου λογισμικού (middleware) έγκειται στη λεγόμενη βάση συστατικών (component-based) αρχιτεκτονική η οποία βασίζεται σε διάφορες διπροσωπίες.

4.2 Ο Ρόλος του Ενδιάμεσου Λογισμικού (Middleware)

Ανακεφαλαιώνοντας, με το πιο πάνω ορισμό ο ρόλος του ενδιαμέσου λογισμικού είναι να παρέχει ενός είδους κοινού αφαιρετικού προγραμματισμού , να είναι σε θέση να κρύβει την ανομοιογένεια και την κατανομή των λειτουργικών συστημάτων και τέλος να κρύβει χαμηλού επιπέδου προγραμματιστικές λεπτομέρειες.

4.3 Πλεονεκτήματα και Μειονεκτήματα Ενδιάμεσου Λογισμικού (Middleware)

Η χρησιμοποίηση του ενδιάμεσου λογισμικού έχει αρκετά ωφελήματα. Αναφέροντας τα σημαντικότερα από αυτά είναι η επαναχρησιμοποίηση του κώδικα, η ανεξαρτησία των προγραμματιστικών γλωσσών και διάφορων πλατφόρμων , η σχετική ευκολία στην ανάπτυξη διάφορων εφαρμογών από την πλευρά των προγραμματιστών και τέλος και ίσως το πιο σημαντικότερο το χαμηλό κόστος ανάπτυξης καθώς και το χαμηλό κόστος συντήρησης του ενδιάμεσου λογισμικού.

Το middleware προσφέρει μοναδικά πλεονεκτήματα στις επιχειρήσεις και στη βιομηχανία. Για παράδειγμα παραδοσιακά συστήματα βάσεων δεδομένων αναπτύχθηκαν σε κλειστά περιβάλλοντα, όπου οι χρήστες έχουν πρόσβαση στο σύστημα μόνο διαμέσου ενός περιορισμένου δικτύου ή ενός διαδικτύου. Το ενδιάμεσο λογισμικό (middleware) διευκολύνει διαφανή πρόσβαση σε παλαιότερα συστήματα βάσεων δεδομένων ή εφαρμογές διαμέσου συστήματα βάσεων δεδομένων ή εφαρμογές διαμέσου web server χωρίς να υπολείπονται συγκεκριμένων χαρακτηριστικών των βάσεων δεδομένων. Οι επιχειρήσεις συχνά χρησιμοποιούν middleware εφαρμογές για να συνδέσουν πληροφορίες από διάφορες βάσεις δεδομένων, όπως μισθοδοσίας, πωλήσεων και λογιστηρίου ή βάσεις δεδομένων οι οποίες εδρεύουν σε διαφορετικές γεωγραφικές τοποθεσίες. Στην υψηλή ανταγωνιστική κοινωνία περίθαλψης, τα εργαστήρια κάνουν εκτεταμένη χρήση ενδιάμεσου λογισμικού (middleware)

εφαρμογών για τη συλλογή πληροφοριών, την αποθήκευση πληροφοριών συστήματος των εργαστηρίων και τον συνδυασμό συστημάτων κατά την συγχώνευση νοσοκομείων. Το ενδιάμεσο λογισμικό (middleware) βοηθάει να γεφυρωθεί το κενό ανάμεσα σε ξεχωριστά συστήματα πληροφοριών εργαστηρίων σε ένα νέο σχηματισμένο δίκτυο περίθαλψης.

Από την άλλη πλευρά όπως κάθε είδους λογισμικού παρουσιάζει και κάποια μειονεκτήματα .Ένα από αυτά είναι η περίπτωση λάθους, η οποία μπορεί να προκύψει από την λανθασμένη αποδοχή μιας μεταβίβασης μηνυμάτων από τα επίπεδα, τα οποία επικοινωνεί το ενδιάμεσο λογισμικό.

4.4 Προέλευση

Προς το τέλος της δεκαετίας του 1970 ο όρος ενδιάμεσου λογισμικού , ήταν σε χρήση. Όμως το 1980 ,αναπτύχθηκε με γρήγορους ρυθμούς , χωρίς να αργήσει να κερδίσει τα βλέμματα των προγραμματιστών. Αρχικά χρησιμοποιούταν ως λύση στο πρόβλημα της επικοινωνίας σύγχρονων εφαρμογών με παλαιότερα συστήματα.

Το ενδιάμεσο λογισμικό (middleware) για αρκετά χρόνια παρέχετουν στη αγορά από δύο κυρίως οργανισμούς, την IBM και την Oracle. Στη μετέπειτα εξέλιξη αυτού του ενδιάμεσου λογισμικού, εμφανίστηκαν και διάφοροι οργανισμοί που σκοπός τους ήταν η παροχή Εργαλείων τύπου middleware προσανατολισμένα στο web . Επιπλέον υπήρχαν και οι ομάδες όπως η Apache Software Foundation που υποστήριζαν την ανάπτυξη ανοικτού κώδικα middleware. Ο τύπος της αρχιτεκτονικής της Microsoft Net, είναι στην ουσία του middleware με τυπικές συναρτήσεις κατανεμημένες ανάμεσα σε ποικίλα προϊόντα με αλληλεπίδραση μεταξύ υπολογιστών με τα πρότυπα της βιομηχανίας, άνοιξε την άδεια λογισμικού APIS.

4.5 Χαρακτηριστικά του Ενδιάμεσου Λογισμικού

Το ενδιάμεσο λογισμικό είναι συνήθως λογισμικό που ενώνει τους παροχής υπηρεσιών

(service suppliers) και τους χρήστες των υπηρεσιών αυτών (service consumers). Παροχέας υπηρεσιών θεωρείται οποιοσδήποτε τύπος δικτυακού κόμβου (υλικό ή λογισμικό) που μπορεί να προσφέρει υπηρεσίες (π.χ. εκτυπωτές, αισθητήρες, βάσεις δεδομένων και εφαρμογές). Αντίστοιχα, χρήστης υπηρεσιών είναι οποιοσδήποτε τύπος δικτυακού κόμβου που απαιτεί υπηρεσίες από ένα παροχέα. Οι εφαρμογές επιπλέον μπορούν να συμπεριφέρονται και σαν παροχής υπηρεσιών αλλά και σαν χρήστες των παρεχόμενων υπηρεσιών ταυτόχρονα .

Χαρακτηριστικά θα μπορούσαμε να αναφέρουμε πως το ενδιάμεσο λογισμικό χαρακτηρίζεται από τα παρακάτω:

• Ανεξαρτησία από το δίκτυο

Το ενδιάμεσο λογισμικό (middleware) συνήθως λειτουργεί σαν μια γέφυρα μμεταξύ πολλαπλών δικτυακών εφαρμογών, ή πρέπει να προσαρμοστεί σε πολλαπλά υποκείμενα δίκτυα. Αυτό περικλείει την ικανότητα να συνδέονται ενσύρματες δικτυακές τεχνολογίες, όπως τοπικά δίκτυα Ethernet και ATM δίκτυα κορμού μεταξύ τους ή με άλλα ασύρματα δίκτυα όπως Bluetooth δίκτυα, 802.11 ή υπέρυθρα ασύρματα δίκτυα.

Πέρα από όλα αυτά ένα ενδιάμεσο λογισμικό για να είναι ευέλικτο θα πρέπει να είναι ανεξάρτητο από το δίκτυο στο οποίο εφαρμόζεται.

• Χρονοπρογραμματισμός

Μερικά περιβάλλοντα έχουν αυστηρούς περιορισμούς σε λειτουργίες που μπορούν να εκτελεστούν σε μια συγκεκριμένη χρονική στιγμή. Το ενδιάμεσο λογισμικό μπορεί να αποφασίσει για την χρονική σειρά των αλληλεπιδράσεων, βάσει προτεραιοτήτων ή περιορισμών του εύρους ζώνης .

• Διαλειτουργικότητα

Σε μερικά συστήματα το ενδιάμεσο λογισμικό δίνει έμφαση στην ανάγκη σύνδεσης μεταξύ πολλαπλών γλωσσών και /ή μεταξύ πλατφόρμων ενδιάμεσου λογισμικού.

4.6 Τύποι middleware

Αποτελεί γεγονός ότι υπάρχουν μια πληθώρα αριθμό από διάφορους τύπους middleware που χρησιμοποιούνται από τους διάφορους προγραμματιστές. Πιο κάτω παράθεσα τους σημαντικότερους τύπους από πλευράς επεκτασιμότητας.

• Remote Procedure Call - ο πελάτης κάνει κλήσεις σε διαδικασίες που τρέχουν σε απομακρυσμένα συστήματα. Μπορεί να είναι σύγχρονες ή ασύγχρονες.

• Message Oriented Middleware - Τα μηνύματα που στέλνονται στον πελάτη συλλέγονται και αποθηκεύονται ενώ ο πελάτης συνεχίζει με άλλες διαδικασίες.

• Object Request Broker - Αυτός ο τύπος ενδιάμεσου λογισμικού (middleware) δίνει τη δυνατότητα στις εφαρμογές να στέλνουν αντικείμενα και υπηρεσίες σε ένα σύστημα προσανατολισμένο στα αντικείμενα.

• SQL - oriented Data Access - είναι το ενδιάμεσο λογισμικό (middleware) ανάμεσα στους severs των εφαρμογών και των βάσεων δεδομένων.

• Embedded Middleware - υπηρεσίες τηλεπικοινωνιών και ολοκληρωμένη διεπαφή λογισμικού που λειτουργεί ανάμεσα σε ενσωματωμένες εφαρμογές και λειτουργικά συστήματα πραγματικού χρόνου.

4.7 Ορισμός Osgi

Osgi υπηρεσία ορίζεται από μία component-based πλατφόρμα για υλοποίηση εφαρμογών γραμμένη σε αντικειμενοστραφείς γλώσσα προγραμματισμού (Java).Osgi

framework παρέχει τις βασικές έννοιες που επιτρέπουν στις εφαρμογές να κατασκευαστούν από μικρά, αξιόπιστα, επαναχρησιμοποιημένα συστατικά.

OSGI απαρτίζεται από μία διευθυντική μονάδα, το λεγόμενο bundle, το οποίο μπορεί να εγκατασταθεί, αναβαθμιστεί, να ξεκινήσει ή να σταματήσει η λειτουργία του χωρίς να χρειάζεται επανεκκίνηση του OSGI framework.Επιπλέον υπάρχουν ψηλού επιπέδου εργαλεία, οι λεγόμενες υπηρεσίες, οι οποίες είναι σε θέση να υποστηρίζουν την αλληλεπίδραση των bundles με το OSGI framework.Αυτές οι υπηρεσίες προσφέρουν μία καλύτερη επικοινωνία μεταξύ διάφορων εργαλείων που απαρτίζουν το OSGI. Με αυτό τον τρόπο παρέχουν ευκαιρίες για αύξηση της αξιοπιστίας της λειτουργίας του OSGI.

4.8 Αρχιτεκτονική και Υπηρεσίες

Η αρχιτεκτονική του OSGI framework απαρτίζεται από τα εξής επίπεδα : Το επίπεδο ασφαλείας, επίπεδο εργαλείων, κύκλου ζωής επίπεδο και το επίπεδο υπηρεσίας. Μεταξύ αυτών το επιπέδων, υπάρχει μία εξάρτηση. Το επίπεδο εργαλείων μπορεί να χρησιμοποιηθεί χωρίς το επίπεδο κύκλου ζωής και υπηρεσίας. Επιπροσθέτως το επίπεδο κύκλου ζωής μπορεί να χρησιμοποιηθεί χωρίς το επίπεδο υπηρεσίας για τη λειτουργία του απαιτεί την υπάρξει όλων το πιο πάνω αναφερόμενων επιπέδων.

Οι υπηρεσίες του Osgi οορίζονται και υλοποιούνται με δύο κύρια συστατικά, τη υπηρεσία διπροσωπίας και την υπηρεσία αντικειμένου. Το πρώτο συστατικό αντιστοιχεί σε μία κλάση διπροσωπίας τύπου public σε αντικειμενοστραφή προγραμματισμό, όπου το σώμα της συνάρτησης χρειάζεται λίγη υλοποίηση, το δεύτερο συστατικό προσδιορίζεται από ένα συγκεκριμένο bundle και η λειτουργία του εξαρτάται σε μεγάλο βαθμό από τη κατάσταση που βρίσκεται το αναφερόμενο bundle μια χρονική στιγμή (ενεργοποιημένο ή απενεργοποιημένο).Γενικά οι OSGI υπηρεσίες είναι σε θέση να παρέχουν σημαντικές πληροφορίες για τη λειτουργία του OSGI framework.

Υπάρχουν σημαντικά οφέλη στη χρησιμοποίηση του OSGI που μπορούν να κάνουν τη ζωή του προγραμματιστή πιο εύκολη .Καταρχάς ο κώδικας είναι πιο εύκολος να το γράψουμε(πιο κατανοητός), τα σφάλματα ανιχνεύονται πιο γρήγορα(κέρδος στο χρόνο) και τέλος η εφαρμογή αυτή είναι γρήγορη στις διάφορες εφαρμογές που θέλει να χρησιμοποιήσει ο χρήστης.

4.9 IST-MUSIC Middleware

Ένα παράδειγμα αρχιτεκτονικής ενδιάμεσου λογισμικού που βασίζεται στην αρχιτεκτονική του Java OSGI και αξίζει να αναφέρουμε είναι το MUSIC(Mobile Users IN Ubiquitous Computing Environments). Το ερευνητικό πρόγραμμα MUSIC, επικεντρώνεται στο να ευκολύνει στη ανάπτυξη context-aware εφαρμογών, όπου δυναμικά προσαρμόζονται στις οποιεσδήποτε αλλαγές του χρήστη, και παράγουν μια MUSIC context εφαρμογή, όπου μπορεί να υλοποιηθεί σε Windows, Linux και Android συσκευές.

Το κίνητρο για δημιουργία του MUSIC project, έγκειται στη αύξηση της χρήσης κινητών συσκευών από τους χρήστες. Όταν οι χρήστες στην καθημερινή τους ζωή επηρεάζονται από τα πληροφορικά συστήματα, πολλές καταστάσεις τους επηρεάζονται από διάφορες υπηρεσίες(όπως για παράδειγμα η συνδεσιμότητα μια Wi Fi σύνδεσης).Σε τέτοιου είδους περιβάλλον οι χρήστες θα μπορούν να ωφεληθούν με τη χρήση του context-awareness και της αυτό-προσαρμοστικότητας. Η ζήτηση για τέτοιου είδους υπηρεσίες, που έχουν την ικανότητα να προσαρμόζονται ανάλογα με το περιβάλλον του χρήστη αυξάνονται διαρκώς, όπου η ανάπτυξη των συγκεκριμένων υπηρεσιών είναι δύσκολη και χρονοβόρα.

Εδώ έρχεται ο ρόλος δημιουργίας του MUSIC project που έγκειται στην απλούστευση της ανάπτυξης των αυτόπροσαρμοστικών εφαρμογών, όπου αυτό το επιτυγχάνει προσφέροντας μια μοντελοκεντρική (model-driver) μεθοδολογία που προσεγγίζει τη δραστηριότητα ανάπτυξης μιας αυτό προσαρμοστικής εφαρμογής και επαναχρησιμοποίηση συστατικών και υπηρεσιών.

Μερικά από τα ωφελήματα του MUSIC project μπορούν να παρατηρηθούν σε ένα ευρύ φάσμα περιοχών:

-Περιπλοκότητα της Αρχιτεκτονικής :Το project αυτό υλοποιείται σε μία εύκολη και κατανοητή αρχιτεκτονική.

-Επαναχρησιμοποίηση κώδικα: Μέσα από διάφορα έτοιμα context plug-ins και reasoners(έτοιμες και άμεσα επαναχρησιμοποιημένες μονάδες κώδικα σε JAR αρχεία JAVA) που έχουν υλοποιηθεί βάση της αρχιτεκτονικής του ενδιάμεσου λογισμικού προϋπάρχουν.

Πηγές διευθέτησης : Υπάρχει μία σωστή διευθέτηση κατανομής της μνήμης(αφού τα plug –ins χρησιμοποιούνται μόνο όταν χρειάζονται).

-Πολλαπλές Εφαρμογές : Τα context-plug-ins κατασκευάζονται απευθείας με βάση την αρχιτεκτονική του ενδιάμεσου λογισμικού (middleware) ,και αυτό κάνει πιο εύκολη τη εγκατάσταση, ενεργοποίηση και χρήση όταν χρειάζονται.

Ανακεφαλαιώνοντας, το MUSIC project, παρέχει μία ανοικτού τύπου πλατφόρμα παροχής υπηρεσιών και εργαλείων για τη δημιουργία εφαρμογών από τους χρήστες λαμβάνοντας υπόψη την ανάγκη για αυτό-προσαρμογή της εφαρμογής βάση πληροφοριών που αφορούν τον χρήστη και την εφαρμογή.

Κεφάλαιο 5

RSCM (Really Simple Context Middleware)

5.1 Ορισμός
5.2 Κατηγοριοποίηση του RSCM
5.3 Κύρια περιεχόμενα του RSCM

5.1 Ορισμός

Το RSCM είναι ένα είδος ενδιάμεσου λογισμικού, το οποίο αποτελεί τη θεμελιώδη βάση για την ανάπτυξη και υλοποίηση της εφαρμογής μας. Βασίζεται στην αρχιτεκτονική του MUSIC, με τη διαφορά ότι δεν γίνεται χρήση Java Osgi αλλά γίνεται χρήση των Android Services.

Είναι ένα λογισμικό που επιτρέπει την ανάπτυξη εφαρμογών πλαισίου (context-aware) για μια Android πλατφόρμα, μέσω υφισταμένων context plug-ins, καθώς και τη δυνατότητα επέκτασης και υλοποίησης context plug-ins. Με την έννοια context-plug-ins εννοούμε διάφορα κομμάτια κώδικα τα οποία το καθένα από αυτά είναι ανεξάρτητα μεταξύ τους και εκτελούν διάφορες λειτουργίες το καθένα καλώντας τα από την κύρια δραστηριότητα (λεπτομέρειες για αυτά θα παρουσιαστούν στο κεφάλαιο για την υλοποίηση της εφαρμογή μας σε πλατφόρμα λογισμικού Android).

Η αρχιτεκτονική αυτού του μοντέλου είναι βασισμένη στη φιλοσοφία των Android

Services, η οποία διευκολύνει τον διαχωρισμό ανάμεσα στα context plug-ins και τις εφαρμογές, καθώς και επιτρέπει τη χρήση των context plug-ins από διαφορετικές εφαρμογές.

Η κύρια ιδέα είναι να έχουμε ατομικά context plug-ins τα οποία είναι επαναχρησιμοποιήσιμα και μπορούν να εγκατασταθούν, ενεργοποιηθούν και εκτελεστούν δυναμικά και όταν χρειάζονται από την εφαρμογή. Αυτά τα plug-ins παράγουν δυναμικές πληροφορίες οι οποίες χρησιμοποιούνται για προσαρμογή των λειτουργιών της εφαρμογής επίγνωσης πλαισίου..

5.2 Κατηγοριοποίηση του RSCM

Τα context-plug-in μπορούν να διαχωριστούν σε 2 κατηγορίες .Στο context sensors και στα context reasoners .Στα context sensors εννοούμε τα αρχικά plugins τα οποία προϋπάρχουν για εξαγωγή δεδομένων.

Context sensors plug ins: Τα context sensors plug ins, η χρήση τους είναι καθαρά για παροχή context πληροφοριών, δηλαδή ανάλογα με τη χρήση του κάθε plug in (π.χ. location plug in, wifi plug in, battery plug in), είναι σε θέση να επεξεργάζεται δεδομένα και όταν κληθεί το συγκεκριμένο plug in από μια δραστηριότητα τότε δίνει τα στοιχεία που υλοποιεί το συγκεκριμένο plug in.Τα context sensors plug ins, λειτουργούν ανεξάρτητα από το πρόγραμμα το οποίο τρέχουν, και τα συστατικά τους στοιχεία βασίζονται στην επαναχρησιμοποίηση του κώδικα, που αλληλεπιδρούν.

Context reasoners plug ins: Είναι η επέκταση των Context sensors plug ins, δηλαδή εξαρτούνται πλήρως από τα context sensors plug ins για να λειτουργήσουν. Συγκεκριμένα παίρνουν τις απαραίτητες πληροφορίες από τα sensors plug ins, και τις επεξεργάζονται κατάλληλα για να υλοποιήσουν την εφαρμογή που τους ανατέθηκε. Για να γίνουμε πιο κατανοητή, αν το context sensor plug in, που χρησιμοποιούμε είναι υπεύθυνο για να μας ενημερώνει σε πια κατάσταση βρίσκεται το Wi Fi, επιστρέφοντας μία Boolean μορφή π.χ.(connected/not connected), τότε χρησιμοποιώντας το Context

reasoner plug in, και παίρνοντας τις απαραίτητες πληροφορίες από το αντίστοιχο context sensor plug in, είναι υπεύθυνο για προέκταση της λειτουργίας του ενημερώνοντας το χρήστη, ονομαστικά σε ποιο Wi Fi, είναι ενωμένος.

5.3 Κύρια περιεχόμενα του RSCM

Rscm Library : Ορίζεται ως ένα από τα πιο κύρια συστατικά στοιχεία λειτουργίας του RSCM. Η βιβλιοθήκη που μας προσφέρει το RSCM εμπεριέχει έτοιμες υλοποιημένες κλάσεις, οι οποίες αυτόματα με την εγκατάσταση τους στην εφαρμογή μας, είναι στη διάθεση του προγραμματιστή να τις χρησιμοποιήσει ανά πάσα στιγμή .Οι κλάσεις αυτές, είναι τύπου αντικειμενοστραφή προγραμματισμού (Java Android) και η κάθε μια διαδραματίζει το δικό της ρόλο, ανάλογα με το είδος της εφαρμογής που θα αναπτύξουμε και υλοποιήσουμε.
Σχεδόν πάντοτε στην ανάπτυξη, μίας context aware εφαρμογής , το rscm_library jar, αρχείο πρέπει να τοποθετηθεί σαν μία εξωτερική βιβλιοθήκη στο android application project.

Rscm Runtime: Περιέχει πληροφορίες, οι οποίες τρέχουν στο παρασκήνιο και επιπλέον διευκολύνει τη μετάβαση ανάμεσα στα plug-ins και στις εφαρμογές. Αυτή η υπηρεσία ξεκινά αυτόματα, όταν χρειάζεται. Αν δεν είναι διαθέσιμη, τότε η εφαρμογή προτρέπει τον χρήστη να κάνει εγκατάσταση, ανακατευθύνοντας στην αντίστοιχη Android Market page .Το RSCM Runtime μπορεί να χαρακτηριστεί ως και η βάση του RSCM, διότι είναι το πρώτο και απαραίτητο plug-in που πρέπει να εγκαταστήσουμε για τη λειτουργιά της εφαρμογής μας.

Εκτός από αυτά τα κύρια χαρακτηριστικά που μας προσφέρει το RSCM, τα οποία είναι έτοιμα προς το χρήστη, υπάρχουν και τα plug-ins.Για παράδειγμα το activity wi-fi plug-in, μας ενημερώνει αν ο χρήστης είναι σε κατάσταση συνδεσιμότητας ή όχι με το wi-fi, και το battery plug-in που μας ενημερώνει για τη διαθέσιμη μπαταριά που έχει η ηλεκτρονική συσκευή μας.(λεπτομερώς για αυτά τα plug-ins θα αναφερθούμε στο κεφάλαιο 6).Επιπλέον σημαντικό μέρος για το RSCM αποτελούν τα λεγόμενα reasoners

plug-ins, τα οποία το καθένα από αυτά παίρνει πληροφορίες από το αντίστοιχο plug-ins ,και τις επεξεργάζεται με τέτοιο τρόπο ώστε να εξάγει πιο λεπτομερής πληροφορίες από το αντίστοιχο plug-in .

Κεφάλαιο 6

Υλοποίηση της εφαρμογή μας με τη χρήση του RSCM

6.1 Γενική αναφορά στη εφαρμογή πλαισίου

6.2 Context plug-ins βασισμένα στη λειτουργία του RSCM

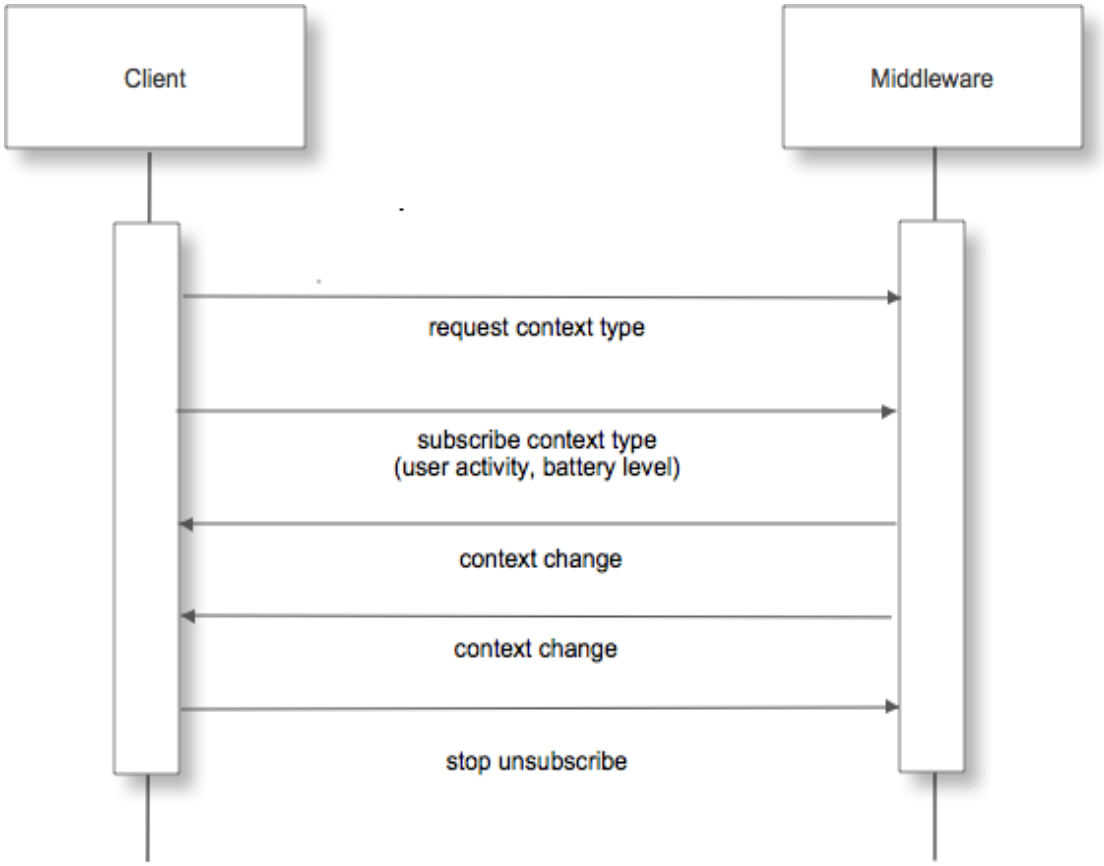
6.3 Colloquium room, μία εφαρμογή βασισμένη στο RSCM

6.4 Δυσκολίες κατά την υλοποίηση της εφαρμογής μας

6.1 Γενική αναφορά στη εφαρμογή πλαισίου

Αυτό το κεφάλαιο παρουσιάζει, τη υλοποίηση μίας εφαρμογής πλαισίου (context aware), χρησιμοποιώντας το RSCM(, το οποίο τρέχει σε πλατφόρμα Android. Οπως, έχουμε προαναφέρει το ενδιάμεσο λογισμικό (middleware) αλληλεπιδρά με το λειτουργικό σύστημα και της εφαρμογής και παρέχει πληροφορίες με ένα εύκολο και αποτελεσματικό τρόπο.

Το εν λόγο σύστημα μας αποτελείται από δύο κύρια συστατικά την εφαρμογή επίγνωσης πλαισίου (Context aware application)και το RSCM, το οποίο όπως έχουμε αναφέρει είναι ένα είδους ενδιάμεσο λογισμικό επίγνωσης πλαισίου (Context Middleware). Στο σχήμα 6.1 παρουσιάζει την επίδραση των 2 πιο πάνω συστατικών που έχουμε προαναφέρει.



Σχήμα 6.1

6.2 Context plug-ins βασισμένα στη λειτουργία του RSCM

Τα context plug-ins είναι επαναχρησιμοποιημένα κομμάτια κώδικα που μαζεύουν δυναμικά πληροφορίες (context) αν και όποτε χρειάζονται. Είναι τα κύρια συστατικά που απαρτίζουν το σύστημα μας.
Μπορούμε να τα κατηγοριοποιήσουμε σε δύο κατηγορίες σε context reasoners και context sensors.

Τα context sensors plug ins, η χρήση τους είναι καθαρά για παροχή πληροφοριών πλαισίου (context).Από την άλλη τα context reasoners plug-ins η χρήση παίρνουν ως είσοδο πληροφορίες πλαισίου (context) από άλλα plug- ins και τις μετατρέπουν σε ψηλού επιπέδου πληροφορίες πλαισίου (context)τα οποία χρησιμοποιούνται σε

εφαρμογές επίγνωσης πλαισίου (context-aware).Τα plug-ins δεν είναι συνεχώς σε μόνιμη λειτουργία παρά μόνο ενεργοποιούνται όταν χρειάζονται και είναι αναγκαία στη υλοποίηση της εφαρμογής. Αυτό αυτόματα μας παραπέμπει σε έξυπνο σύστημα, που είναι σε θέση να αποφασίζει ανάλογα με τα δεδομένα της εφαρμογής, ποια plug ins να ενεργοποιήσουμε και ποια όχι. Τα Context plugs ins δημιουργήθηκαν ως ανεξάρτητα συστατικά ενός συστήματος και η χρήση τους εξαρτάται από την εφαρμογή επίγνωσης πλαισίου.

Στην εφαρμογή μας, κάνουμε χρήση 2 ανεξάρτητων plug ins, και συγκεκριμένα η εφαρμογή μας υποστηρίζει τη χρήση 2 context sensors plug ins :

Battery plug in : Ενημερώνει το χρήστη πόσο επίπεδο της μπαταρίας του έχει απομείνει π.χ. (The battery level is 57%).

Wi-Fi plug-in: Ενημερώνει το χρήστη ποια είναι η κατάσταση συνδεσιμότητας του wifi ανά πάσα στιγμή π.χ. (The Wi-Fi is connected).

Τα context plug ins ως ανεξάρτητα συστατικά του συστήματος παρουσιάζουν ενδιαφέρον, στο τρόπο λειτουργίας τους :

-Υλοποιούν μεθόδους ενεργοποίησης και απενεργοποίησης για να αρχίσουν και να σταματήσουν αντίστοιχα τη λειτουργία του context event.

-Προεκτείνουν τη λειτουργία τους με διάφορες υπηρεσίες και υλοποιημένες κλάσεις για να παράγουν το context που τους ανατέθηκε.

-Η υλοποίηση της Context Changed() μεθόδου, η οποία μέσα από διάφορα δεδομένα εισόδου που επεξεργάζεται επικοινωνεί με το εν λόγο plug in.

6.3 Colloquium room , η εφαρμογή μας βασισμένη στο RSCM

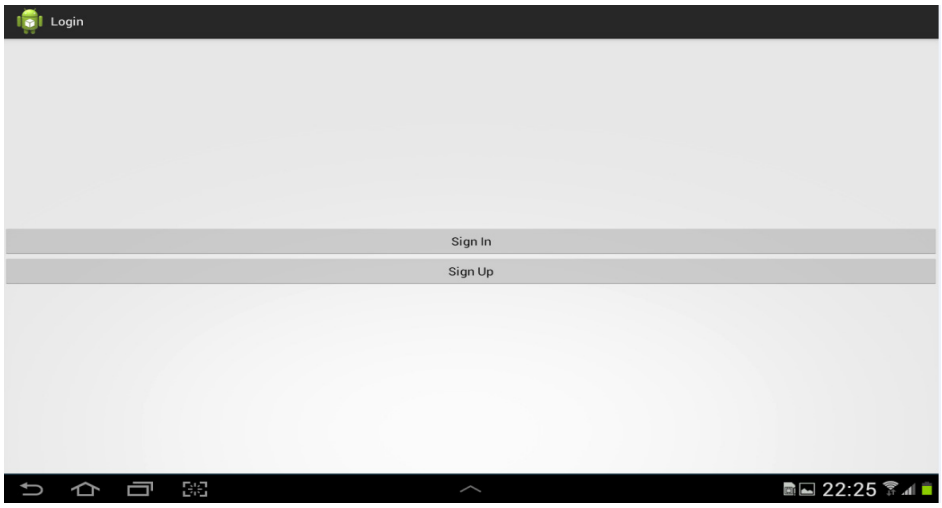
Το colloquium room, η ονομασία της εφαρμογή μας έχει ως στόχο την ανάπτυξη μίας εφαρμογής επίγνωσης πλαισίου (context aware), βασισμένη και υλοποιημένη σε RSCM .Η εφαρμογή μας, χρησιμοποιώντας όλες τις ευκολίες που παρέχει το RSCM (επαναχρησιμοποίηση του κώδικα) που θα δούμε λεπτομερώς στα επόμενα υποκεφάλαια, παρουσιάζει τα events στο χρήστη ανάλογα με τις προτιμήσεις του. Συγκεκριμένα η context aware εφαρμογή μας υλοποιήθηκε κυρίως για φοιτητές του πανεπιστημίου μας.Ανάλογα με τη σπουδή του κάθε φοιτητή π.χ.(Μαθηματικός,Φυσικός κ.α.) , θα μπορεί να έχει εγκαταστημένη τη εφαρμογή μας στο κινητό του, και ανάλογα με τις προτιμήσεις του, θα του παρουσιάζονται λεπτομερώς οι πληροφορίες του event που τον ενδιαφέρουν.

Πιο κάτω παρουσιάζουμε τις λεπτομέρειες της υλοποίησης μας :

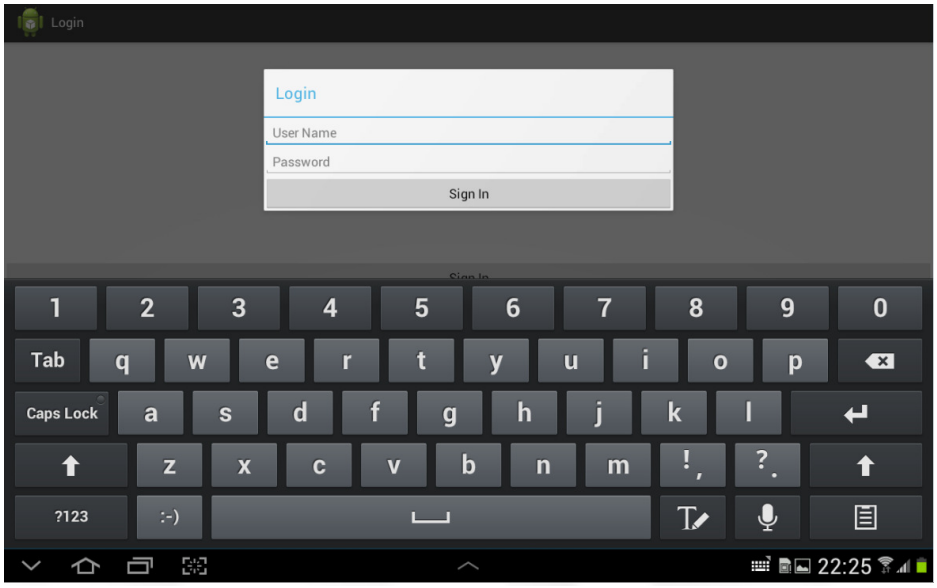
Σύστημα sign in sign up(βλ. εικόνα 6.1)

Ένας καινούργιος χρήστης, έχοντας εγκαταστήσει την εφαρμογή μας στη συσκευή του, μπορεί πατώντας το κουμπί 'SIGN UP' ,(βλ. εικόνα 6.3) που το παρουσιάζεται να γίνεται μέλλος του colloquium room, δίνοντας το username και το password που επιθυμεί. Αυτό επιτυγχάνεται με τη χρησιμοποίηση μίας βάσης δεδομένων, σε SQLite. Από τη στιγμή που ένα μέλος θα κάνει τουλάχιστο μία φορά sign up στη εφαρμογή μας, τότε τα αντίστοιχα στοιχεία του θα αποθηκεύονται στις αντίστοιχες στήλες του πίνακα της βάσης μας. Έχοντας αποθηκευμένα τα στοιχεία αυτά, ο χρήστης είναι έτοιμος για τη λειτουργία της εφαρμογής μας πατώντας το αντίστοιχο κουμπί SIGN IN(,(βλ.εικόνα 6.2), όπου βάζοντας τα στοιχεία του αρχίζει η κύρια λειτουργία της εφαρμογής. Για λόγους ασφάλειας και αξιοπιστίας της εφαρμογής μας, αν ο χρήστης βάλει λάθος στοιχεία κατά τη διάρκεια της εγγραφής του, θα του παρουσιάζεται μήνυμα λάθους, και το σύστημα θα ζητά τα σωστά δεδομένα εγγραφής.

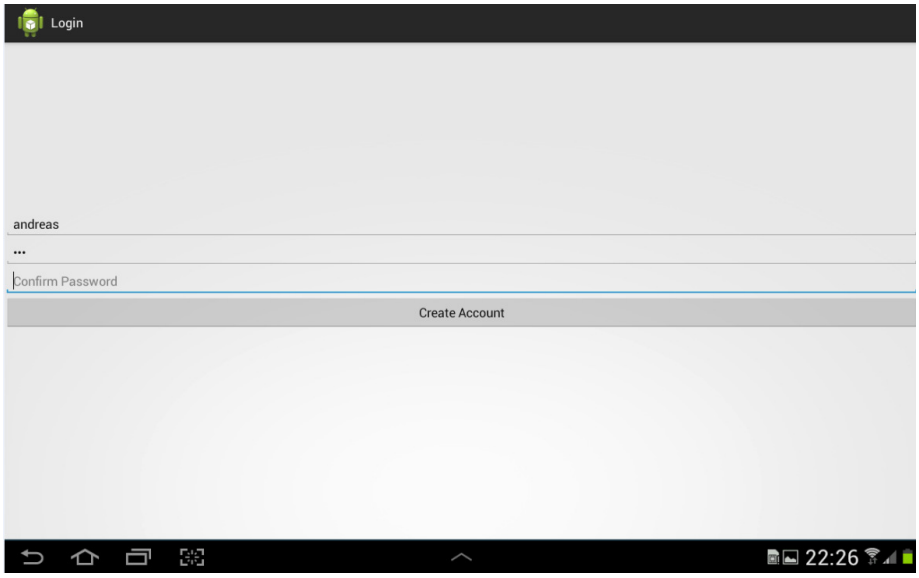
Ο χρήστης, κάνοντας sign in,του παρουσιάζεται στην οθόνη 2 μηνύματα, το πρώτο μήνυμα τον ενημερώνει για το επίπεδο της μπαταριάς π.χ. (The Battery level is 60 %) και το άλλο μήνυμα τον ενημερώνει για την κατάσταση του wifi π.χ. (The wifi is connected).



Εικόνα 6.1 (Λειτουργία Sign in και Sign up)



Εικόνα 6.2 (Λειτουργία Sign in)



Εικόνα 6.3 (Λειτουργία Sign up)

Main Activity του RSCM

Η κλάση μας αυτή προεκτείνει τη λειτουργία της χρησιμοποιώντας τη κλάση ContextListenerActivity, η οποία βρίσκεται υλοποιημένη μέσα στο rscm_library .jar, που έχουμε εγκαταστημένη ως εξωτερική βιβλιοθήκη μέσα στο android project . Καλώντας τη μέθοδο request_scope(), μας επιστρέφει τα scopes των αντίστοιχο plug ins που χρησιμοποιούμε, δηλαδή του battery_level και του wifi_connectivity. Καλώντας στη συνέχεια τη συνάρτηση, "public void onContextValuechanged ()", μας επιστέφει τα αντίστοιχα scope.

Εδώ έχουμε 2 περιπτώσεις να ελέγξουμε:

1) Αν το scope, που μας έχει επιστραφεί είναι το battery.level , τότε παίρνουμε τη τιμή του επιπέδου της μπαταρίας σε μία χρονική στιγμή καλώντας τη

μέθοδο“contextValue.getValueAsInteger()”, η οποία μας επιστρέφει ένα ακέραιο αριθμό, ο οποίος έχουμε αναφέρει είναι σε μορφή % π.χ. (60%),(βλ εικόνα 6.4) , όπου είναι το επίπεδο της μπαταρίας του αντίστοιχου χρήστη κάποιας χρονικής στιγμής.

2) Αν το scope, που μας έχει επιστραφεί είναι το Wi-Fi.connected, τότε παίρνουμε τη Boolean τιμή του wifi καλώντας τη μέθοδο context “Value.getValueAsBoolean” (), η οποία μας επιστρέφει αν είμαστε συνδεδεμένοι στο Wi-Fi (connected) ή αν δεν είμαστε συνδεδεμένοι (not connected), τυπώνοντας και το ανάλογο μήνυμα στη δραστηριότητα μας (The Wi-Fi is connected/not connected) ,(βλ εικόνα 6.4).

Εικόνα 6.4 (Λειτουργία battery and Wi-Fi plug-in)

Για να πραγματοποιηθούν αυτές οι δύο περιπτώσεις ,έχουμε δημιουργήσει και τα αντίστοιχα plug ins :

Battery plug in

Το battery plug-in, όπως και τα άλλα plug-ins βεβαίως η λειτουργία τους βασίζεται στη επαναχρησιμοποίηση έτοιμου κώδικα .Η υλοποίηση αυτού του plug-in βασίζεται στη λειτουργία μιας κλάσης της λεγόμενης Battery Sensor.java η οποία η λειτουργία της προεκτείνεται μέσα από τις υπηρεσίες που προσφέρει το Sensor Service, το οποίο αποτελείται από πολλές υλοποιημένες έτοιμες κλάσεις, και βρίσκεται μέσα στο rscm_library.jar που έχει εγκαταστηθεί σαν εξωτερική βιβλιοθήκη στο Android project της εφαρμογής και σε όλα τα plug-ins που το απαρτίζουν.

Στο σώμα της κλάσης αυτού του plug-in, κύρια λειτουργία κατέχει η μέθοδος notify Listener(lastContextValuebatterylevel).Με εφαρμογή αυτής τη μεθόδου, παίρνουμε στη μεταβλητή ContextValueBatteryLevel ένα ακέραιο αριθμό, το όποίο συμβολίζει το επίπεδο μπαταρίας της εφαρμογής του χρήστη κάποια συγκεκριμένη στιγμή. Τη μεταβλητή αυτή με το ακέραιο αριθμό είμαστε σε θέση να τη εξασφαλίζουμε με τη κλήση της κατάλληλης συνάρτησης notify listener η οποία βρίσκεται μέσα στο Sensor Service και μας επιστέφει την τιμή αυτή.

Σημαντικό στο τρόπο υλοποίησης του battery_plug-in είναι και η κλήση των δύων αντίστοιχων μεθόδων onBind() και on unBind(). Η κλήση της πρώτης μεθόδου επιστρέφει ένα αντικείμενο intent το οποίο είναι τύπου IBinder, και του οποίου λειτουργία του είναι να χρησιμοποιείται στη κλήση των μεθόδων για να πάρουμε το αποτέλεσμα μας, που σε αυτή τη περίπτωση είναι το επίπεδο της μπαταρίας.

Για να τεθεί σε λειτουργία το battery plug in, που λειτουργεί σαν ξεχωριστό project από αυτό της εφαρμογής, είναι υποχρεωτική η εγκατάσταση του rscm_library.jar σαν εξωτερικής βιβλιοθήκη. Επιπλέον μέσα στο αρχείο manifest battery_plug_in.xml, έπρεπε ως επιπρόσθετες εντολές να δηλωθούν τα εξής permissions android: value=”battery. Level, battery.voltage, battery.temp”, για να ενεργοποιηθεί η λειτουργία του battery_plug_in .

Wi-Fi_plug-in

Σε αυτό το plug-in, όπως και στο battery plug-in ο στόχος είναι η επαναχρησιμοποίηση κώδικα. Η λειτουργία του Wi-Fi plug-in, βασίζεται στην ανίχνευση της αλλαγής μίας κατάστασης στη συσκευή του χρήστη, όσο αφορά στις υπηρεσίες παροχής Wi-Fi. Αν ο χρήστης είναι σε κατάσταση συνδεσιμότητας με την Android εφαρμογή του (δηλαδή έχει ενεργοποίηση την υπηρεσία Wi-Fi στη συσκευή του), τότε του εμφανίζεται στη δραστηριότητα της εφαρμογής μήνυμα ότι είναι συνδεδεμένος (The wifi is connected). Σε αντίθετη περίπτωση όπου ο χρήστης είναι σε κατάσταση μη συνδεσιμότητας με την Android εφαρμογή του (δηλαδή έχει απενεργοποιημένη την υπηρεσία Wi-Fi στη συσκευή του), τότε του εμφανίζεται μήνυμα στη δραστηριότητα της συσκευής ότι είναι σε κατάσταση μη συνδεσιμότητας (The wifi is not connected).

Το Wi-Fi plug-in λειτουργεί σε ένα ξεχωριστό android project, όπως και έχει γίνει με το battery plug-in.Εγινε εγκατάσταση της εξωτερικής βιβλιοθήκης rscm_library.jar που περιέχει έτοιμες υλοποιημένες και βασικές συναρτήσεις του RSCM.

Αποτελείται από τη WifiConnectivitySensor κλάση, η οποία και αυτή προεκτείνεται από τη Sensor Service που βρίσκεται στην αντίστοιχη βιβλιοθήκη Μέσα στο σώμα της κλάσης αυτής που υλοποιεί το wifi plug in προϋπάρχει μέθοδος « (wifimanager) context.getSystemService (Context.WIFISERVICE) », η οποία αυτή η μέθοδος της επιστρέφει σαν παράμετρος τη υπηρεσία του Wi-Fi, μια υπηρεσία που εντοπίζει τη παροχή και διαθεσιμότητας διαδικτύου κάποιας χρονικής στιγμής.

Επιπλέον, υπάρχει και η συνάρτηση notify Listener (ContectValue.createContextValue (SCOPE_NETWORK_WIFI_CONNECTED, wifi_manager.isEnabled()). Στέλνοντας τις συγκεκριμένες παραμέτρους στην αντίστοιχη συνάρτηση που βρίσκεται μέσα στο Sensor Service της βιβλιοθήκης rscm_library.jar, παίρνουμε ως αποτέλεσμα το

request_scope του wifi plug in, επιστρέφοντας και μια Boolean τύπου μεταβλητή που ο λόγος δημιουργίας της είναι να μας ενημερώσει αν ο χρήστης έχει ενεργοποιημένη ή όχι την υπηρεσία Wi Fi στη συσκευή του .

6.4 Δυσκολίες κατά την υλοποίηση της εφαρμογής μας

Σε αυτό το σημείο αναφέρονται οι δυσκολίες που αντιμετωπίστηκαν κατά την επαναχρησιμοποίηση του κώδικα για τη χρήση του Wi Fi -plug-in.Ο κώδικας ήταν δοκιμασμένος και λειτουργήσιμος σε ποιο χαμηλού API Level android συσκευές, και έτσι χρησιμοποιώντας την τελευταία έκδοση android API 17 έπρεπε στο wifi.xml, manifest αρχείο να αλλάξουν κάποια permission για να ήταν εφικτή η λειτουργία του. Συγκεκριμένα το permission που προϋπήρχε για την χρησιμοποίηση σε προηγούμενα API Level για android συσκευές ήταν η αντίστοιχη :

<android.permission.ACCESS_INTERNET>

η οποία για να είναι εφικτή σε platform android API 17 LEVEL αντικαταστήθηκε με την πιο κάτω εντολή μέσα στο manifest.xml αρχείο του wifi_plug-in:

<αndroid.permission.ACCESS_WIFI_STATE>

Μια επιπλέον δυσκολία που αντιμετωπίστηκε κατά τη διάρκεια της ανάπτυξη της εφαρμογής ήταν στην εγκατάσταση ως εξωτερικής βιβλιοθήκης της rscm_library.jar.Η έκδοση της βιβλιοθήκης που ενσωματώνει έτοιμες υλοποιημένες κλάσεις υποστήριξης του RSCM δεν ταίριαζαν με APIs που είχε εγκατασταθεί. Έτσι ξαναδημιουργήθηκε η

βιβλιοθήκη του RSCM από τον κώδικα που υπάρχει στο https://code.google.com/p/rscm/ και την εγκατέστησα επιτυχώς σαν εξωτερική βιβλιοθήκη στο project.

Κεφάλαιο 7

Υλοποίηση της εφαρμογής μέσω διπροσωπίας για το χρήστη

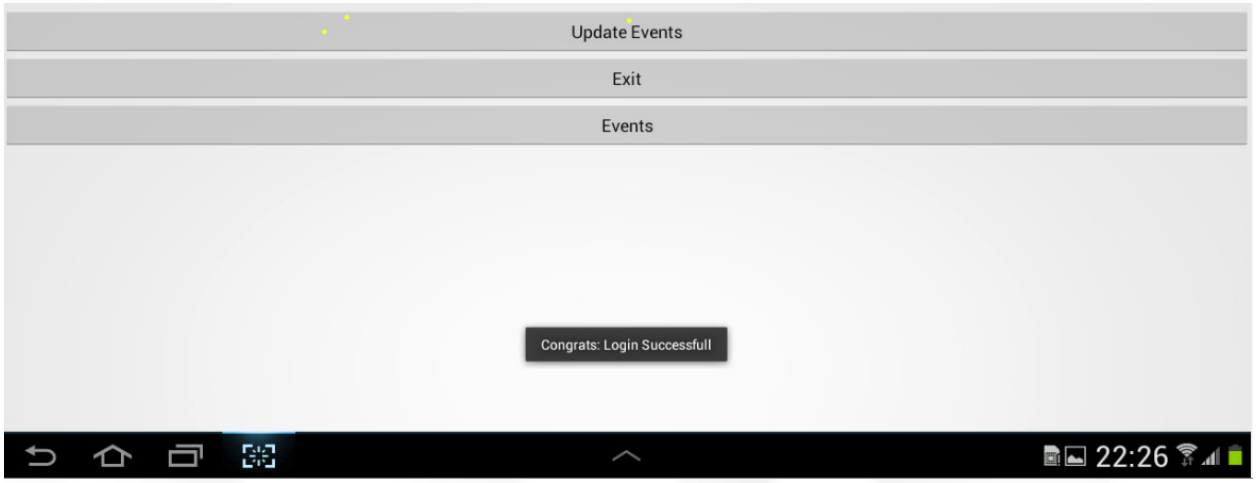
7.1 Διπροσωπία χρήστη μέσω android εφαρμογών

7.2 Σκοπός χρήσης battery plug in

7.3 Σκοπός χρήσης Wi Fi plug in

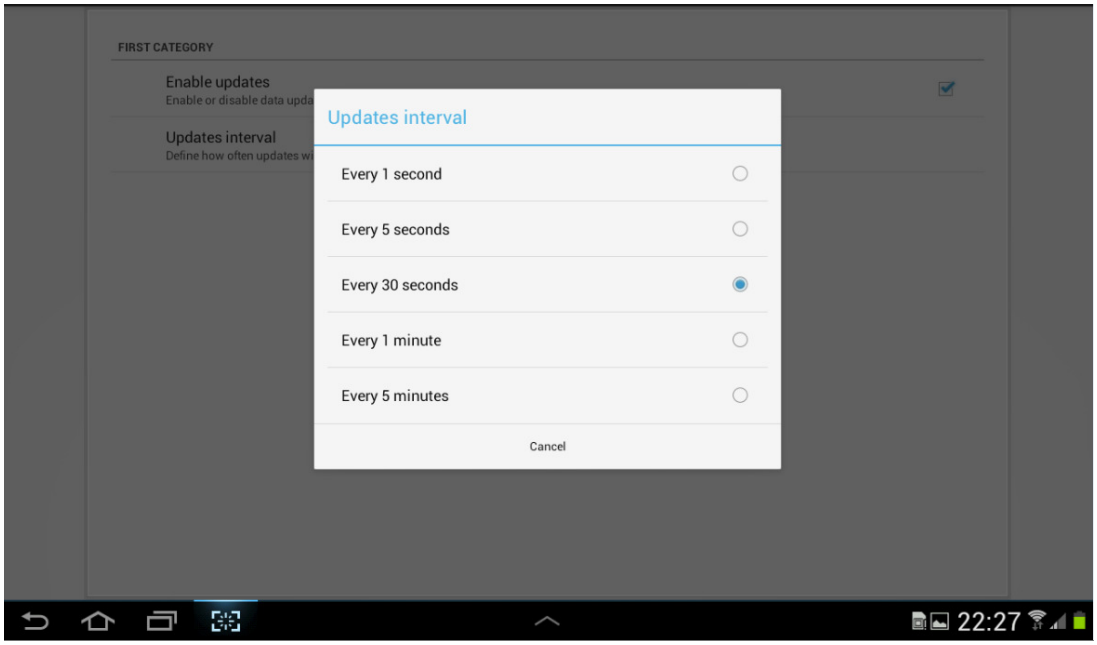
7.1 Διπροσωπία χρήστη μέσω android εφαρμογών

Ο κυρίως στόχος ήταν η λειτουργία και επαναχρησιμοποίηση του RSCM ως ένα είδος middleware καθώς και των δύο plug-ins.Επίσης χρησιμοποιήθηκαν και οι δυνατότητες που παρέχει το Android API 17 για υλοποίηση της εφαρμογής επίγνωσης πλαισίου(context aware) . Στη κύρια μου δραστηριότητα υπάρχουν 3 κουμπιά(buttons Βλ. εικόνα 7.1) τα οποία λειτουργούν ως εξής:



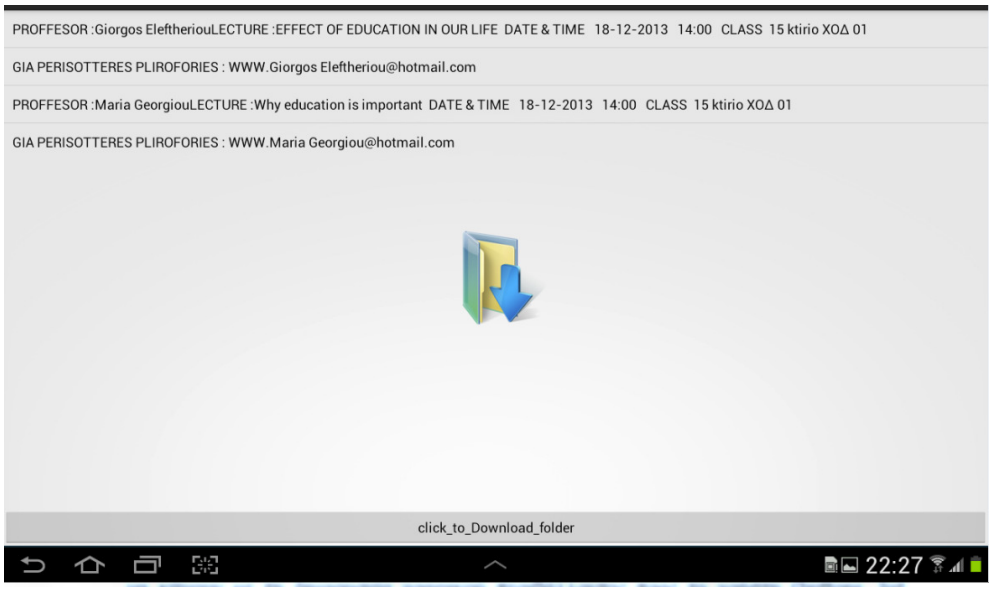
Εικόνα7.1 (Υλοποίηση κυρίως menu)

Πρώτο Button (Enable Updates) : Πατώντας αυτό το κουμπί(button) ο χρήστης είναι σε θέση να πραγματοποιεί αναβαθμίσεις στη βάση του, ανάλογα με τη χρονική διάρκεια που επιθυμεί να πραγματοποιούνται. Τη δημιουργία του τη στήριξα στο ότι όταν ένας χρήστης μπορούσε να έχει την εφαρμογή του σε λειτουργία στο παρασκήνιο και ενώ θα ήταν συνδεδεμένος, να μπορεί να κοιτάξει αν έχει δημιουργηθεί ένα νέο event .Αυτό πραγματοποιείται με αναβάθμιση της βάσεις δεδομένων και συγκεκριμένα του πίνακα που περιέχονται τα events και ανάλογα με τη χρονική διάρκεια που θα γίνονται αναβαθμίσεις στη βάση (ανάλογα με την επιλογή του χρήστη) ,θα του παρουσιάζεται αν τυχόν δημιουργήθηκε ένα νέο event,(Βλ.εικόνα 7.2).



Εικόνα 7.2 (Υλοποίηση για αναβάθμιση βάσης)

Δεύτερο Button(Events): Κάνωντας αυτή την επιλογή ο χρήστης, θα του παρουσιάζεται μια list view από μαθήματα στα οποία πραγματοποιούνται events(π.χ. Μαθηματικά, Πληροφορική).Ο χρήστης, ανάλογα με τις προτιμήσεις του και τα ενδιαφέροντα του θα κάνει την επιλογή που θέλει. Κάνοντας αυτή την επιλογή του παρουσιάζονται κάποια στοιχεία για το συγκεκριμένο event.Για παράδειγμα το όνομα του Event, ο καθηγητής που το παρουσιάζει ημερομηνία και ώρα διεξαγωγής καθώς και το αντίστοιχο κτίριο και αίθουσα που θα παρουσιαστεί το συγκεκριμένο event, (Βλ. εικόνα 7.3).



Εικόνα 7.3(Παρουσίαση events ανάλογα με ην επιλογή του χρήστη)

Τρίτο button(Exit) : Πατώντας το συγκεκριμένο κουμπί(button) ο χρήστης θα βγαίνει από την εφαρμογή μας.

Οι Android υπηρεσίες που χρησιμοποίησα για την υλοποίηση της εφαρμογής μου, είχαν να κάνουν με τη δημιουργία γραφικού περιβάλλοντος προς το χρήστη (buttons, list views, image, edit text, text view) και την απαραίτητη δήλωση τους μέσα στα αντίστοιχα layout.xml αρχεία. Τα αρχεία αυτά, τα δημιουργούσα, ανά δραστηριότητα(activity), είναι αρχεία τύπου xml, στα οποία μέσα δηλώνουμε τις αντίστοιχες εντολές για τη δημιουργία γραφικού περιβάλλοντος(user interface)π.χ.(Buttons, list views, image, edit text, text view), όπου την ονομασία της κάθε μίας κατηγορίας αλληλεπίδρασης με το χρήστη την ονόμαζα ξεχωριστά μέσα στο strings.xml αρχείο.

Επιπλέον η εναλλαγή των δραστηριοτήτων και η μετάβαση πολλών τιμών(του επιπέδου της μπαταρίας και τη κατάσταση συνδεσιμότητας του wi fi) από μια

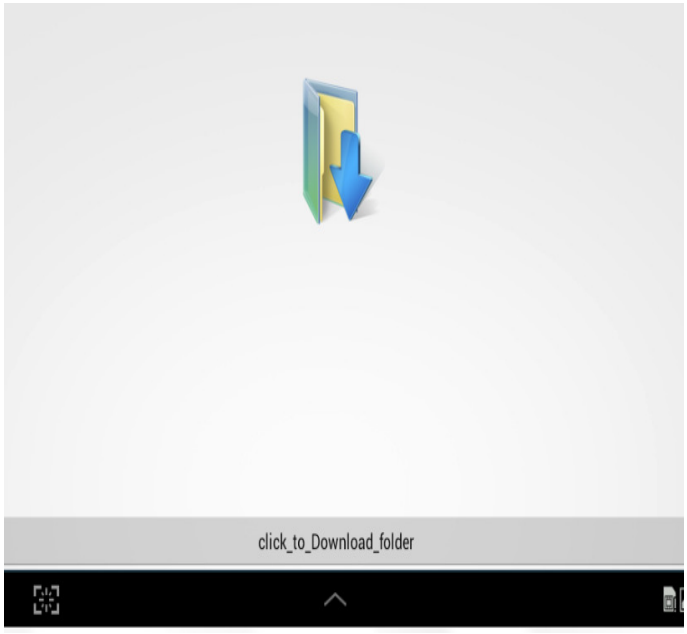
δραστηριότητα στην άλλη ήταν μια υπηρεσία που χρησιμοποίησα και τέλος η σύνδεση της βάσης δεδομένων μου (sqlite) πραγματοποιήθηκε μέσο της android εφαρμογής.

7.2 Σκοπός χρήσης battery plug in

Όπως έχουμε ήδη προαναφέρει έχουμε κάνε χρήση δύο plug-ins για την ανάπτυξη και υλοποίηση της εφαρμογής μας. Τα συγκεκριμένα plug-ins, το κάθε ένα από αυτά έχει το δικό του ρόλο και χρησιμότητα στη εφαρμογή μας.

Σκοπός χρήσης battery plug-in : Με την εγκατάσταση αυτού του plug –in και με την ιδιότητα του , όπως έχουμε προαναφέρει κάνουμε την εφαρμογή του, μέσε στο android project μας με το εξής τρόπο :

Όταν ο χρήστης έχει ποσοστό επιπέδου μπαταρίας μεγαλύτερο του 20%, τότε θα του επιτρέπεται να κατεβάζει ένα φάκελο που θα περιέχει πληροφορίες ανάλογα (Βλ. εικόνα 7.4) με το event που το ενδιαφέρει. Σε αντίθετη περίπτωση θα του παρουσιάζεται το εξής μήνυμα : “LOW BATTERY “, αν το ποσοστό της μπαταρίας του είναι χαμηλότερο ή ίσο με 20% και δεν θα τον επιτρέπει να κατεβάζει το συγκεκριμένο φάκελο (folder).



Εικόνα 7.4(Παρουσίαση φακέλου προς το χρήστη αν η μπαταρία του είναι πάνω από τα επιτρεπτά όρια)

7.3 Σκοπός χρήσης wifi plug in

Με την εγκατάσταση του wifi plug-in, και με την αντίστοιχη χρήση του μέσα στην εφαρμογή μας όπως έχουμε ήδη προαναφέρει κάνουμε την εφαρμογή του με τον εξής τρόπο : Αν ο χρήστης είναι σε κατάσταση συνδεσιμότητας (connected) τότε με την επιλογή του στο κουμπί events θα του παρουσιάζονται όλες οι σχετικές πληροφορίες για το event π.χ.(το email του καθηγητή που θα κάνει το event για να μπορεί να έρθει σε επικοινωνία μαζί του) αν δεν είναι σε κατάσταση συνδεσιμότητας (is not connected), τότε η context aware εφαρμογή μας θα συνεχίζει κανονικά τη λειτουργίας της, εκτελώντας τις οποιεσδήποτε επιλογές του χρήστη.(Βλ. εικόνα 7.5)

Εικόνα 7.5
(Χρήση Wi Fi plug in όταν είναι σε κατάσταση συνδεσιμότητας)

Κεφάλαιο 8

Συμπεράσματα και Μελλοντικές Επεκτάσεις

8.1 Συμπεράσματα

8.2 Μελλοντικές προεκτάσεις

8.1 Συμπεράσματα

Στόχος της διπλωματικής μας, ήταν η ανάπτυξη μίας context aware εφαρμογής, χρησιμοποιώντας το ενδιάμεσο λογισμικό (RSCM), που έχει αναλυθεί στα προηγούμενα κεφάλαια, όπου οι προγραμματιστές των context aware εφαρμογών, έχουν στη διάθεση τους τη χρήση ευέλικτου ενδιάμεσου λογισμικού, το οποίο προσφέρει δυνατότητα επαναχρησιμοποίησης κώδικα μέθοδος επαναχρησιμοποίησης κώδικα που παρέχεται από τη χρήση του RSCM παρέχει τη δυνατότητα απλούστευσης της υλοποίησης της εφαρμογής επίγνωσης πλαισίου καθώς και την ευκολία κατανόησης της εφαρμογής.

Τα έτοιμα context reasoners και context sensors plug ins που προσφέρει το RSCM, προσδίδουν στην υλοποίηση της εφαρμογής μας, ένα είδος απλότητας και την αποφυγή από μέρους ενός προγραμματιστή να εμπλακεί σε ατελείωτα μη κατανοητά κομμάτια

κώδικα, τα οποία έχουν να κάνουν με πολύπλοκες λειτουργίες όπως π.χ. λήψη πληροφοριών από αισθητήρες.

Μέσα, από την υλοποίηση του RSCM συστήματος είναι σαφείς τα ωφελήματα που προσφέρει, όσο αφορά την απλότητα και την ευκαμψία και παράλληλα τη συμβατότητα με το λειτουργικό σύστημα Android.

Ανακεφαλαιώνοντας, μέσα από τους τρόπους και μεθόδους υλοποίησης της εφαρμογής, έχουμε καταλήξει στο συμπέρασμα, ότι έχοντας στη διάθεση μας, μία context aware εφαρμογή, η οποία χρησιμοποιεί υλοποιημένα plug ins, κάνει τα πράγματα πιο εύκολα στην υλοποίηση της context εφαρμογής. Επιπρόσθετα, είναι πολύ πιο εύκολο για τους προγραμματιστές να επαναχρησιμοποιήσουν αυτά τα συστατικά και να κάνουν αλλαγές σε αυτά σύμφωνα με τις ανάγκες τους. Τέλος υπάρχει η άμεση δυνατότητα επέκτασης της εφαρμογής με την συμπερίληψη υφιστάμενων plug-ins ή την δημιουργία καινούριων plug-ins τα οποία μπορούν να χρησιμοποιηθούν άμεσα από την εφαρμογή ή και πιθανές μελλοντικές εφαρμογές επίγνωσης πλαισίου.

8.2 Μελλοντικές προεκτάσεις

Βλέποντας τις προεκτάσεις και εφαρμογές που παραθέτει η εφαρμογή μας στη διάθεση του χρήστη θα ήθελα να αναφέρω τις ιδέες που έχουμε όσο αφορά την επέκταση της εφαρμογής μας στο μέλλον. Καταρχάς, μία υλοποίηση του location plug-in, θα αποτελούσε μία καλή σκέψη σε συνδυασμό με τα ήδη 2 plug-ins που προϋπάρχουν στην εφαρμογή μας και εξηγήσαμε αναλυτικά στα προηγούμενα κεφάλαια .Την εφικτή δημιουργία του location plug-ins θα μπορούσαμε να εντοπίζουμε τη τοποθεσία του χρήστη, εντός πανεπιστημιακού χώρου και να τον ειδοποιούμε ανάλογα για το πού παίρνει δράση το event που τον ενδιαφέρει, και συγκεκριμένα μία υπόδειξη του κτιρίου και της αίθουσας που βρίσκεται.

Μία άλλη σκέψη, θα ήταν η προέκταση του ήδη υπάρχον Wi-Fi plug-in δημιουργώντας το reasoner wifi plug-in.Το συγκεκριμένο reasoner plug-in, θα παίρνει τις κατάλληλες και απαραίτητες πληροφορίες από το wifi sensor plugin, και στην συνέχεια επεξεργάζοντας τα δεδομένα που θα του παρέχει, θα δημιουργηθεί το αντίστοιχο Wi-Fi reasoner plug-in, που θα τον ενημερώνει για το ακριβώς είδος του wifi που είναι ενωμένη η εν λόγο συσκευή του χρήστη.

Με την μελλοντική εφαρμογή και υλοποίηση των δύο αυτών κομματιών που έχω προαναφέρει, θεωρώ ότι εφαρμογή θα προσφέρει όλες τις κατάλληλες πληροφορίες που θα θελήσει να έχει ο χρήστης στην διάθεση του.

Βιβλιογραφία

[1] http://developer.android.com/about/index.html

[2] http://android-app-tutorial.blogspot.com/2012/08/architecture-system-application-stack.html#.UZ18FqIa6So

[3] http://www.angroid.gr

[4] http://developer.android.com/training/basics/firstapp/index.html

[5] http://www.vogella.com/articles/OSGi/article.html

[6] http://searchsoa.techtarget.com/definition/middleware

[7] https://code.google.com/p/rscm/

[8] https://code.google.com/p/rscm/downloads/list

[9] https://code.google.com/p/rscm/source/checkout

ΠΑΡΑΡΤΗΜΑ Α

Σε αυτό το παράρτημα παραθέτουμε τις απαραίτητες συναρτήσεις για να δουλέψει η εφαρμογή .

-

/*ΚΛΑΣΗ ΓΙΑ ΥΛΟΠΟΙΗΣΗ ΤΗΣ ΒΑΣΗΣ ΜΑΣ*/

package com.example.rscm_context;

```
import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteDatabase.CursorFactory;
import android.database.sqlite.SQLiteOpenHelper;
import android.util.Log;

public class DataBaseHelper extends SQLiteOpenHelper
{
        public DataBaseHelper(Context context, String name,CursorFactory factory, int version)
   {
                super(context, name, factory, version);
        }
        // Called when no database exists in disk and the helper class needs
        // to create a new one.
        @Override
        public void onCreate(SQLiteDatabase _db)
        {
                        _db.execSQL(LoginDataBaseAdapter.DATABASE_CREATE);

        }
        // Called when there is a database version mismatch meaning that the version
        // of the database on disk needs to be upgraded to the current version.
        @Override
        public void onUpgrade(SQLiteDatabase _db, int _oldVersion, int _newVersion)
        {
                        // Log the version upgrade.
                        Log.w("TaskDBAdapter", "Upgrading from version " +_oldVersion + "
to " +_newVersion + ", which will destroy all old data");


                        // Upgrade the existing database to conform to the new version. Multiple
                        // previous versions can be handled by comparing _oldVersion and
_newVersion
                        // values.
                        // The simplest case is to drop the old table and create a new one.
                        _db.execSQL("DROP TABLE IF EXISTS " + "TEMPLATE");
                        // Create a new one.
```

```
                onCreate(_db);
        }


}
package com.example.rscm_context;



import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class SignUPActivity extends Activity
{
        EditText editTextUserName,editTextPassword,editTextConfirmPassword;
        Button btnCreateAccount;

        LoginDataBaseAdapter loginDataBaseAdapter;
        @Override
        protected void onCreate(Bundle savedInstanceState)
        {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.signup);

                // get Instance  of Database Adapter
                loginDataBaseAdapter=new LoginDataBaseAdapter(this);
                loginDataBaseAdapter=loginDataBaseAdapter.open();

                // Get Refferences of Views
                editTextUserName=(EditText)findViewById(R.id.editTextUserName);
                editTextPassword=(EditText)findViewById(R.id.editTextPassword);

        editTextConfirmPassword=(EditText)findViewById(R.id.editTextConfirmPassword);

                btnCreateAccount=(Button)findViewById(R.id.buttonCreateAccount);
                btnCreateAccount.setOnClickListener(new View.OnClickListener() {

                public void onClick(View v) {
                        // TODO Auto-generated method stub

                        String userName=editTextUserName.getText().toString();
                        String password=editTextPassword.getText().toString();
                        String confirmPassword=editTextConfirmPassword.getText().toString();

                        // check if any of the fields are vaccant

        if(userName.equals("")||password.equals("")||confirmPassword.equals(""))
                        {
                                        Toast.makeText(getApplicationContext(), "Field
Vaccant", Toast.LENGTH_LONG).show();
                                        return;
                        }
                        // check if both password matches
                        if(!password.equals(confirmPassword))
                        {
```

```
                                        Toast.makeText(getApplicationContext(), "Password does not
match", Toast.LENGTH_LONG).show();
                                        return;
                                }
                                else
                                {
                                   // Save the Data in Database
                                   loginDataBaseAdapter.insertEntry(userName, password);
                                   Toast.makeText(getApplicationContext(), "Account Successfully
Created ", Toast.LENGTH_LONG).show();
                                }
                        }
        });
}
        @Override
        protected void onDestroy() {
                // TODO Auto-generated method stub
                super.onDestroy();

                loginDataBaseAdapter.close();
        }
}
```

/*ΔΗΜΙΟΡΓΙΑ ΤΗΣ ΚΥΡΙΑΣ ΔΡΑΣΤΗΡΙΟΤΗΤΑΣ ΤΗΣ ΕΦΑΡΜΟΓΗΣ ΜΑΣ*/

```
package com.example.rscm_context;


import android.content.ContentValues;
im package com.example.rscm_context;




import android.app.Activity;

import android.content.SharedPreferences;

import android.os.Bundle;

import android.preference.PreferenceManager;
```

```
import android.widget.TextView;


public class ShowSettingsActivity extends Activity {


 @Override
 protected void onCreate(Bundle savedInstanceState) {


  super.onCreate(savedInstanceState);

  setContentView(R.layout.c);


  SharedPreferences sharedPrefs = PreferenceManager.getDefaultSharedPreferences(this);


  StringBuilder builder = new StringBuilder();


  builder.append("\n" + sharedPrefs.getBoolean("perform_updates", false));

  builder.append("\n" + sharedPrefs.getString("updates_interval", "-1"));

  builder.append("\n" + sharedPrefs.getString("welcome_message", "NULL"));


  TextView settingsTextView = (TextView) findViewById(R.id.settings_text_view);

  settingsTextView.setText(builder.toString());


 }


} port android.content.Context;
import android.database.Cursor;
import android.database.SQLException;
import android.database.sqlite.SQLiteDatabase;

public class LoginDataBaseAdapter
{
                    static final String DATABASE_NAME = "sascxc";
                    static final int DATABASE_VERSION = 1;
                    public static final int NAME_COLUMN = 1;
                    // TODO: Create public field for each column in your table.
                    // SQL Statement to create a new database.
                    static final String DATABASE_CREATE = "create table "+"LOGIN"+
                                        "( " +"ID"+" integer primary key autoincrement,"+
"USERNAME  text,PASSWORD text); ";
```

```
                    // Variable to hold the database instance
                    public  SQLiteDatabase db;
                    // Context of the application using the database.
                    private final Context context;
                    // Database open/upgrade helper
                    private DataBaseHelper dbHelper;
                    public  LoginDataBaseAdapter(Context _context)
                    {
                            context = _context;
                            dbHelper = new DataBaseHelper(context, DATABASE_NAME, null,
DATABASE_VERSION);
                    }
                    public  LoginDataBaseAdapter open() throws SQLException
                    {
                            db = dbHelper.getWritableDatabase();
                            return this;
                    }
                    public void close()
                    {
                            db.close();
                    }

                    public  SQLiteDatabase getDatabaseInstance()
                    {
                            return db;
                    }

                    public void insertEntry(String userName,String password)
                    {
                ContentValues newValues = new ContentValues();
                            // Assign values for each row.
                            newValues.put("USERNAME", userName);
                            newValues.put("PASSWORD",password);

                            // Insert the row into your table
                            db.insert("LOGIN", null, newValues);
                            ///Toast.makeText(context, "Reminder Is Successfully Saved",
Toast.LENGTH_LONG).show();
                    }
                    public int deleteEntry(String UserName)
                    {
                            //String id=String.valueOf(ID);
                       String where="USERNAME=?";
                       int numberOFEntriesDeleted= db.delete("LOGIN", where, new
String[]{UserName}) ;
                // Toast.makeText(context, "Number fo Entry Deleted Successfully :
"+numberOFEntriesDeleted, Toast.LENGTH_LONG).show();
                 return numberOFEntriesDeleted;
                    }
                    public String getSinlgeEntry(String userName)
                    {
                            Cursor cursor=db.query("LOGIN", null, " USERNAME=?", new
String[]{userName}, null, null, null);
                if(cursor.getCount()<1) // UserName Not Exist
                {
                    cursor.close();
                    return "NOT EXIST";
                }
                       cursor.moveToFirst();
```

```
                        String password=
cursor.getString(cursor.getColumnIndex("PASSWORD"));
                        cursor.close();
                        return password;
                }
                public void  updateEntry(String userName,String password)
                {
                        // Define the updated row content.
                        ContentValues updatedValues = new ContentValues();
                        // Assign values for each row.
                        updatedValues.put("USERNAME", userName);
                        updatedValues.put("PASSWORD",password);

            String where="USERNAME = ?";
                    db.update("LOGIN",updatedValues, where, new String[]{userName});

                }
}

package com.example.rscm_context;



import android.app.Activity;
import android.app.Dialog;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class HomeActivity extends Activity
{
        Button btnSignIn,btnSignUp;
        LoginDataBaseAdapter loginDataBaseAdapter;

        @Override
        protected void onCreate(Bundle savedInstanceState)
        {

                super.onCreate(savedInstanceState);
           setContentView(R.layout.main);

           // create a instance of SQLite Database
           loginDataBaseAdapter=new LoginDataBaseAdapter(this);
           loginDataBaseAdapter=loginDataBaseAdapter.open();

           // Get The Refference Of Buttons
           btnSignIn=(Button)findViewById(R.id.buttonSignIN);
           btnSignUp=(Button)findViewById(R.id.buttonSignUP);

          // Set OnClick Listener on SignUp button
          btnSignUp.setOnClickListener(new View.OnClickListener() {
                public void onClick(View v) {
                        // TODO Auto-generated method stub

                        /// Create Intent for SignUpActivity  and Start The Activity
                        Intent intentSignUP=new
Intent(getApplicationContext(),SignUPActivity.class);
```

```
                        startActivity(intentSignUP);
                        }
                });
        }
        // Methos to handleClick Event of Sign In Button
        public void signIn(View V)
          {
                        final Dialog dialog = new Dialog(HomeActivity.this);
                        dialog.setContentView(R.layout.login);
                    dialog.setTitle("Login");

                    // get the Refferences of views
                    final  EditText
editTextUserName=(EditText)dialog.findViewById(R.id.editTextUserNameToLogin);
                    final  EditText
editTextPassword=(EditText)dialog.findViewById(R.id.editTextPasswordToLogin);

                        Button btnSignIn=(Button)dialog.findViewById(R.id.buttonSignIn);

                        // Set On ClickListener
                        btnSignIn.setOnClickListener(new View.OnClickListener() {

                                public void onClick(View v) {
                                        // get The User name and Password
                                        String
userName=editTextUserName.getText().toString();
                                        String password=editTextPassword.getText().toString();
                                         //Intent intent = getIntent();
                                   //String value = intent.getStringExtra("key");
                                        // fetch the Password form database for respective user
name
                                        String
storedPassword=loginDataBaseAdapter.getSinlgeEntry(userName);

                                        // check if the Stored password matches with  Password
entered by user
                                        if(password.equals(storedPassword))
                                        {
                                                Toast.makeText(HomeActivity.this, "Congrats:
Login Successfull", Toast.LENGTH_LONG).show();
                                                dialog.dismiss();

                                                  Intent intent = new Intent(HomeActivity.this,
MainActivity.class);
                                                  String value = null;
                                                intent.putExtra("userName1", userName);
                                                  startActivity(intent);


                                        }
                                        else
                                        {
                                                Toast.makeText(HomeActivity.this, "User Name
or Password does not match", Toast.LENGTH_LONG).show();
                                        }
                                }
                        });

                        dialog.show();
        }
```

```
	@Override
	protected void onDestroy() {
		super.onDestroy();
	   // Close The Database
		loginDataBaseAdapter.close();
	}
}
```

```
package com.example.rscm_context;



import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.ProgressBar;
import android.widget.TextView;
import android.widget.Toast;
import org.aspectsense.rscm.ContextValue;
import org.aspectsense.rscm.context.client.ContextListenerActivity;
import org.json.JSONException;



import java.util.Date;

public class MainActivity extends ContextListenerActivity {
	public static final String WIFI_CONNECTED = "wifi.connected";
	@Override public String[] getRequestedScopes()
   {
      return new String[] { "battery.level", "network.wifi_connected","location.fine"};
   }

	private TextView batteryProgressLabel;
   private ProgressBar batteryProgressBar;
   private CheckBox connectedCheckBox;
   private TextView logMessageTextView;
	private TextView messageTextView;
int batteryLevel=0;
```

```
 boolean wifi_connected="g" != null;

//  private TextView batteryProgressLabel;
  //  private ProgressBar batteryProgressBar;
  //  private CheckBox connectedCheckBox;
  //  private TextView logMessageTextView;


    @Override protected void onCreate(Bundle savedInstanceState)
    {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.ss);
       batteryProgressLabel = (TextView) findViewById(R.id.battery_progress_label);
       batteryProgressBar = (ProgressBar) findViewById(R.id.battery_progress_bar);
       connectedCheckBox = (CheckBox) findViewById(R.id.connected_check_box);
       logMessageTextView = (TextView) findViewById(R.id.log_message);


       //  messageTextView = new TextView(this);

    // setContentView(messageTextView);
       Intent intent = getIntent();
                String value1 = intent.getStringExtra("userName1");

       // setContentView(R.layout.ss);

    //  batteryProgressLabel = (TextView) findViewById(R.id.battery_progress_labe);
     //  batteryProgressBar = (ProgressBar) findViewById(R.id.battery_progress_br);
       //connectedCheckBox = (CheckBox) findViewById(R.id.connected_check_box);
       //logMessageTextView = (TextView) findViewById(R.id.log_message);

                appendMessage("WELCOME "+value1 +"\n");

       appendMessage("Activity created");

       Button btn2 = (Button) findViewById(R.id.btn2);
       btn2.setOnClickListener(new OnClickListener() {
             public void onClick(View v) {
              // TODO Auto-generated method stub
              finish();
              System.exit(0);
            }
         });

       Button btn3 = (Button) findViewById(R.id.btn3);
       btn3.setOnClickListener(new OnClickListener() {
             public void onClick(View v) {


                    Intent intent = new Intent(MainActivity.this, ddf.class);
                Bundle bundle = new Bundle();
                bundle.putInt ("battery", batteryLevel);
                bundle.putBoolean("wifi", wifi_connected);
                   intent.putExtras(bundle);
                startActivity(intent);
```

```
                /*  Intent intent = new Intent(MainActivity.this, ddf.class);
                        intent.putExtra("battery", batteryLevel);
                        intent.putExtra("wifi", wifi_connected);*/

                        System.out.println(batteryLevel);
                        System.out.println(wifi_connected);
                        startActivity(intent);


          }
        });




    Button btn1 = (Button) findViewById(R.id.btn1);
    btn1.setOnClickListener(new View.OnClickListener() {

                        public void onClick(View v) {

                                Intent intent = new Intent(MainActivity.this,
QuickPrefsActivity.class);

                                 startActivity(intent);

                        }

                });


  }

  private void appendMessage(final String message)
  {
        {
      final String currentMessage = logMessageTextView.getText().toString();
    logMessageTextView.setText(currentMessage + "\n" + message);
  }
}


  public  void onContextValueChanged(ContextValue contextValue)
  {
    try
    {

        appendMessage("The context value scope is: " +contextValue.getScope());


       if("battery.level".equals(contextValue.getScope()))
      {


         batteryLevel = contextValue.getValueAsInteger();

        batteryProgressLabel.setText(batteryLevel + "%");
        batteryProgressBar.setProgress(batteryLevel);
```

```
            appendMessage(new Date() + ": The battery level is " + batteryLevel + "%");


          }


      else if ("network.wifi_connected".equals(contextValue.getScope())){
           wifi_connected = contextValue.getValueAsBoolean();
             appendMessage(new Date() + ": The Wifi is " + (wifi_connected ? "connected" :
"disconnected"));

          }



   }
       catch (JSONException jsone)
       {
          Toast.makeText(this, "Error while displaying context event: " + contextValue,
Toast.LENGTH_SHORT).show();
       }

  }
```

```
 package com.example.rscm_context;


import android.content.Intent;

import android.os.Build;
import android.os.Bundle;

import android.preference.PreferenceActivity;

import android.view.Menu;

import android.view.MenuItem;

import com.example.rscm_context.R;

public class QuickPrefsActivity extends PreferenceActivity {



    @Override

    public void onCreate(Bundle savedInstanceState) {

       super.onCreate(savedInstanceState);


          addPreferencesFromResource(R.xml.cc);
```

```
        }


    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        menu.add(Menu.NONE, 0, 0, "Show current settings");
        return super.onCreateOptionsMenu(menu);
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case 0:
                startActivity(new Intent(this, ShowSettingsActivity.class));
                return true;
        }
        return false;
    }

}


package com.example.rscm_context;


import android.app.Activity;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.preference.PreferenceManager;
import android.widget.TextView;
```

```
public class ShowSettingsActivity extends Activity {


 @Override
 protected void onCreate(Bundle savedInstanceState) {


  super.onCreate(savedInstanceState);

  setContentView(R.layout.c);


  SharedPreferences sharedPrefs = PreferenceManager.getDefaultSharedPreferences(this);

  StringBuilder builder = new StringBuilder();


  builder.append("\n" + sharedPrefs.getBoolean("perform_updates", false));

  builder.append("\n" + sharedPrefs.getString("updates_interval", "-1"));

  builder.append("\n" + sharedPrefs.getString("welcome_message", "NULL"));


  TextView settingsTextView = (TextView) findViewById(R.id.settings_text_view);

  settingsTextView.setText(builder.toString());


 }


}
```

ΠΑΡΑΡΤΗΜΑ Β

Σε αυτό το παράρτημα παραθέτουμε το κώδικα του battery plug in

```
package org.aspectsense.rscm.sensor.battery;



import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.os.BatteryManager;
import android.os.IBinder;
import org.aspectsense.rscm.ContextValue;
import org.aspectsense.rscm.context.plugin.SensorService;

public class BatterySensor extends SensorService
{
   public static final String TAG =
"org.aspectsense.rscm.sensor.battery.BatterySensor";

   public static final String SCOPE_BATTERY_LEVEL = "battery.level";

   public static final IntentFilter filter = new
IntentFilter(Intent.ACTION_BATTERY_CHANGED);

   @Override public IBinder onBind(Intent intent)
   {
      registerReceiver(batteryReceiver, filter);

      return super.onBind(intent);
   }

   @Override public boolean onUnbind(Intent intent)
   {
```

```
        unregisterReceiver(batteryReceiver);

        return super.onUnbind(intent);
    }

    final BroadcastReceiver batteryReceiver = new BroadcastReceiver()
    {
        int level = -1;

        @Override public void onReceive(Context context, Intent intent)
        {
            level = intent.getIntExtra(BatteryManager.EXTRA_LEVEL, -1);

            final ContextValue lastContextValueBatteryLevel =
ContextValue.createContextValue(SCOPE_BATTERY_LEVEL, level);

            notifyListener(lastContextValueBatteryLevel);
        }
    };
}
```

ΠΑΡΑΤΗΜΑ Γ

Σε αυτό το παράρτημα παραθέτουμε το κώδικα του wifi plug in

```
package org.aspectsense.rscm.sensor.wifi_connectivity;

import org.aspectsense.rscm.context.plugin.SensorService;


import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.net.wifi.WifiManager;
import android.os.IBinder;
import android.util.Log;
import org.aspectsense.rscm.ContextValue;


public class WifiConnectivitySensor extends SensorService
{
   public static final String TAG =
"org.aspectsense.rscm.sensor.wifi_connectivity.WifiConnectivitySensor";

   public static final String ACTION_NETWORK_CONNECTIVITY_CHANGE
= "android.net.conn.CONNECTIVITY_CHANGE";

   public static final IntentFilter filter = new
IntentFilter(ACTION_NETWORK_CONNECTIVITY_CHANGE);

   @Override public IBinder onBind(Intent intent)
   {
      Log.d(TAG, "WifiConnectivitySensor: registerReceiver");
      registerReceiver(wifiConnectivityReceiver, filter);

      return super.onBind(intent);
   }
```

```
   @Override public boolean onUnbind(Intent intent)
   {
      Log.d(TAG, "WifiConnectivitySensor: unregisterReceiver");
      unregisterReceiver(wifiConnectivityReceiver);

      return super.onUnbind(intent);
   }

   public static final String SCOPE_NETWORK_WIFI_CONNECTED =
"network.wifi_connected";

   final BroadcastReceiver wifiConnectivityReceiver = new BroadcastReceiver()
   {
      @Override public void onReceive(Context context, Intent intent)
      {
         Log.d(TAG, "intent is " + intent);


if(ACTION_NETWORK_CONNECTIVITY_CHANGE.equals(intent.getAction())
)
         {
            final WifiManager wifiManager = (WifiManager)
context.getSystemService(Context.WIFI_SERVICE);

notifyListener(ContextValue.createContextValue(SCOPE_NETWORK_WIFI_CO
NNECTED, wifiManager.isWifiEnabled()));
         }
      }
   };
}
```