

Ατομική Διπλωματική Εργασία

**ΕΦΑΡΜΟΓΗ ΓΙΑ ΕΞΥΠΝΑ ΤΗΛΕΦΩΝΑ
(ANDROID OR IOS APP) ΓΙΑ ΦΩΝΗΤΙΚΗ ΑΛΛΗΛΕΠΙΔΡΑΣΗ
ΜΕ ΣΥΣΤΗΜΑΤΑ ΕΞΥΠΝΟΥ ΣΠΙΤΙΟΥ**

Χαράλαμπος Χαραλάμπους

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ



ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μάιος 2016

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

**ΕΦΑΡΜΟΓΗ ΓΙΑ ΕΞΥΠΝΑ ΤΗΛΕΦΩΝΑ
(ANDROID OR IOS APP) ΓΙΑ ΦΩΝΗΤΙΚΗ
ΑΛΛΗΛΕΠΙΔΡΑΣΗ ΜΕ ΣΥΣΤΗΜΑΤΑ ΕΞΥΠΝΟΥ
ΣΠΙΤΙΟΥ**

Χαράλαμπος Χαραλάμους

Επιβλέπων Καθηγητής

Βάσος Βασιλείου

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

Μάιος 2016

Ευχαριστίες

Θα ήθελα να εκφράσω τις ευχαριστίες μου στον επιβλέποντα καθηγητή μου Δρ. Βάσο Βασιλείου για την εμπιστοσύνη που μου υπέδειξε για την εκπόνηση αυτής της διπλωματικής εργασίας καθώς επίσης και για τις κατευθυντήριες οδηγίες και συμβουλές που μου παρείχε τόσο κατά την εκπόνηση της διπλωματικής μου εργασίας όσο και κατά την διάρκεια των σπουδών μου.

Ακόμη θα ήθελα να ευχαριστήσω τον συμφοιτητή μου Κυριάκο Πέτρου για την βοήθεια που μου προσέφερε για την υλοποίηση της εφαρμογής.

Τέλος θα ήθελα να ευχαριστήσω τους φίλους μου και ιδιαίτερα την οικογένεια μου για την συμπαράσταση και καθοδήγηση που μου πρόσφεραν κατά την διάρκεια της διπλωματικής μου αλλά και καθ' όλη την διάρκεια των σπουδών μου.

Περίληψη

Με την ραγδαία ανάπτυξη της τεχνολογίας σε διάφορους τομείς, έχουμε καταφέρει να δημιουργήσουμε συσκευές που να εξυπηρετούν διάφορες καθημερινές ανάγκες του ανθρώπου, καθώς επίσης και να αλληλεπιδρούμε με υπολογιστικά συστήματα (Smart Phones, laptops, tablets κ.α.) μέσω της φωνής . Αυτή η διπλωματική εργασία έχει στόχο να εξηγήσει τον τρόπο με τον οποίο μπορούμε να δημιουργήσουμε μια εφαρμογή κινητού τηλεφώνου η οποία παρέχει την δυνατότητα να λαμβάνει φωνητικές εντολές από ένα χρήστη και να μεταβάλλει το περιβάλλον ενός «έξυπνου» σπιτιού. Πιο συγκεκριμένα θα γίνει μια εισαγωγή στα 2 βασικά συστατικά της εφαρμογής (Φωνητική Αλληλεπίδραση , «έξυπνο» σπίτι) μέσω επεξήγησης διάφορων ορισμών όπως για παράδειγμα: φωνητική αλληλεπίδραση, Smart, Internet of Things ,Smart Home και Smart phone έτσι ώστε να γίνει πιο κατανοητό το θέμα αυτής της διπλωματικής εργασίας. Επίσης γίνεται παρουσίαση του λειτουργικού συστήματος Android και του προτύπου Command και που θα χρησιμοποιηθούν έτσι ώστε να δημιουργήσουμε την εφαρμογή.

Τέλος αναφέρονται τα συμπεράσματα που προκύπτουν από αυτή την διπλωματική εργασία όπως επίσης και επισημάνσεις έτσι ώστε να γίνουν βελτιώσεις στην εφαρμογή που έχει υλοποιηθεί.

Περιεχόμενα

Ευχαριστίες	i
Περίληψη	ii
Περιεχόμενα	iii
Κεφάλαιο 1 Εισαγωγή	1
1.1 Περιγραφή και Σκοπός της διπλωματικής εργασίας.....	1
1.2 Δομή της διπλωματικής εργασίας	3
Κεφάλαιο 2 Αλληλεπίδραση ανθρώπου υπολογιστή	6
2.1 Ορισμός.....	6
2.2 Φωνητική αλληλεπίδραση	7
2.3 Αναγνώριση φωνής.....	8
2.4 Υλοποίηση φωνητικής αλληλεπίδρασης στην εφαρμογή.....	10
Κεφάλαιο 3 Έξυπνο *	13
3.1 Ορισμός.....	13
3.2 Internet of Things.....	17
Κεφάλαιο 4 Έξυπνο Σπίτι	21
4.1 «Έξυπνο» σπίτι	21
4.2 Πλεονεκτήματα «έξυπνου» σπιτιού.....	23
Κεφάλαιο 5 «Έξυπνο» τηλέφωνο	26
5.1 Ορισμός.....	26
5.2 Λειτουργικά συστήματα κινητών	28
5.3 Αρχιτεκτονική Android	30
Κεφάλαιο 6 Σχεδίαση Εφαρμογής	34
6.1 Πρότυπα.....	34
6.2 Εντολές – Σενάρια	38
6.3 Πρωτόκολλα επικοινωνίας –Αρχιτεκτονικές σχεδίασης	46
6.4 Συνδετικά συστατικά υποσυστήματα	58
6.5 Απαιτήσεις εφαρμογής	60
6.6 Λειτουργία εφαρμογής.....	61
6.7 Σύγκριση εφαρμογής με παρόμοιες εφαρμογές	68

Κεφάλαιο 7 Πειραματισμός - Αξιολόγηση	72
7.1 Εισαγωγή	72
7.2 Αξιολόγηση.....	75
Κεφάλαιο 8 Συμπεράσματα	76
8.1 Συμπεράσματα	76
8.2 Μελλοντικές επεκτάσεις.....	77
Βιβλιογραφία	79
Παράρτημα Α Smart Things Simulator	A-1
1.1 Πηγαίος Κώδικας Smart Things	A-1
Παράρτημα Β Πηγαίος Κώδικας Android	B - 1
1.1 Αρχείο Manifest	B - 1
1.2 Πηγαίος Κώδικας για Διεπαφές	B - 3
1.3 Πηγαίος Κώδικας για Activities.....	B - 5
1.4 Πηγαίος Κώδικας για Κλάσεις.....	B - 23
1.5 Πηγαίος Κώδικας για Layouts	B - 47

Κεφάλαιο 1

Εισαγωγή

1.1	Περιγραφή και Σκοπός της διπλωματικής εργασίας	1
1.2	Δομή της διπλωματικής εργασίας	3

1.1 Περιγραφή και Σκοπός της διπλωματικής εργασίας

Λόγω της συνεχούς ανάπτυξης της τεχνολογίας και των επικοινωνιών έχουν δημιουργηθεί τα «έξυπνα» τηλέφωνα, τα οποία παρέχουν διάφορες υπηρεσίες με στόχο την κάλυψη διαφόρων αναγκών του χρήστη. Για παράδειγμα με την χρήση ενός «έξυπνου» κινητού τηλεφώνου μπορούμε να κάνουμε λήψη φωτογραφιών, να κάνουμε πλοήγηση στο διαδίκτυο και ακόμη με την χρήση διάφορων ενσωματωμένων αισθητήρων να λαμβάνουμε διάφορες μετρήσεις σχετικά με την ταχύτητα, προσανατολισμό, θερμοκρασία, υγρασία κ.α.

Παράλληλα ένας αρκετά ενδιαφέρον τομέας, ο οποίος ασχολείται με την δημιουργία διαφόρων «έξυπνων» συσκευών έχει κεντρίσει το ενδιαφέρον διαφόρων εταιρειών. Αυτό έχει σαν αποτέλεσμα να κατασκευάζουν «έξυπνες» συσκευές, καθώς επίσης και εφαρμογές «έξυπνων» κινητών τηλεφώνων οι οποίες παρέχουν την δυνατότητα στον χρήστη να εκτελεί διάφορες ενέργειες οι οποίες μεταφράζονται σε εντολές που πρέπει να εκτελέσουν οι αντίστοιχες «έξυπνες» συσκευές έτσι ώστε να παραχθεί κάποιο επιθυμητό αποτέλεσμα.

Το πιο πάνω σενάριο μπορούμε να το παρομοιάσουμε με την καθημερινή συνήθεια που κάνουμε όταν επιθυμούμε να αλλάξουμε την κατάσταση μιας απομακρυσμένης συσκευής όπως για παράδειγμα την τηλεόραση, το ηχοσύστημα ή οποιαδήποτε άλλη συσκευή η οποία έχει τηλεχειριστήριο. Οι διαφορές που υπάρχουν μεταξύ τους είναι οι εξής:

1. Η απόσταση: Στην περίπτωση της εφαρμογής ο χρήστης μπορεί να βρίσκεται οπουδήποτε και με το πάτημα ενός κουμπιού να ενεργοποιεί την κατάλληλη συσκευή. Αυτό επιτυγχάνεται με πρόσβαση στο διαδίκτυο και σύνδεση με το δίκτυο του «έξυπνου» σπιτιού ενώ στην περίπτωση του τηλεχειριστηρίου δεν παρέχεται αυτή η δυνατότητα λόγω της περιορισμένης εμβέλειας του σήματος που εκπέμπει.
2. Το τηλεχειριστήριο: Στην εφαρμογή αντιστοιχεί στο «έξυπνο» κινητό τηλέφωνο το οποίο είναι υπεύθυνο για περισσότερες από μια «έξυπνες» συσκευές σε αντίθεση με το κανονικό τηλεχειριστήριο το οποίο περιορίζεται σε μια συσκευή.
3. Η συσκευή δέκτης (συσκευή η οποία εκτελεί την ενέργεια) όπου στην περίπτωση της εφαρμογής αντιστοιχεί σε κάθε συσκευή η οποία μπορεί να έχει πρόσβαση στο διαδίκτυο ενώ στην «παραδοσιακή» συσκευή αντιστοιχεί σε κάθε συσκευή η οποία περιέχει δέκτη υπερύθρου σήματος.

Η εξέλιξη της τεχνολογίας όμως δεν περιορίστηκε μόνο στο υλικό αλλά έχει διευρυνθεί και στο λογισμικό. Για παράδειγμα έχει καταφέρει να δημιουργήσει μια νέα προσέγγιση με την οποία ο χρήστης μπορεί να αλληλεπιδράσει με ένα σύστημα χωρίς να χρειάζεται να πληκτρολογήσει εντολή ή να αγγίξει κάποιο χειριστήριο έτσι ώστε να πραγματοποιήσει μια ενέργεια. Αυτή η νέα προσέγγιση επικεντρώνεται στην φωνή, δηλαδή με την ομιλία του ο χρήστης μπορεί να πραγματοποιήσει σχεδόν όλες τις ενέργειες που πραγματοποιούσε προηγουμένως χωρίς να χρειαστεί να έχει άμεση επαφή με το υλικό(πληκτρολόγιο, οθόνη αφής, ποντίκι).

Σκοπός της Διπλωματικής Εργασίας είναι αφενός η παροχή πληροφοριών σχετικά με το «έξυπνο» σπίτι, (δηλαδή μια εισαγωγή στις έννοιες Smart Things – Smart Home) και αφετέρου να εξηγήσει την μεθοδολογία με την οποία μπορούμε να δημιουργήσουμε μια εφαρμογή «έξυπνου τηλεφώνου» η οποία παρέχει στον χρήστη διάφορες δυνατότητες, έτσι ώστε να μπορεί να αλληλεπιδρά και να μεταβάλλει την κατάσταση του «έξυπνου» σπιτιού .

Επίσης ενδείκνυται και ένας διαφορετικός τρόπος με τον οποίο επιτυγχάνεται αλληλεπίδραση ανθρώπου και μηχανής (μέσω φωνητικής αλληλεπίδρασης) ο οποίος παρατηρείται σε μερικές παρόμοιες εφαρμογές που ήδη υπάρχουν και αρχίζει να γίνεται ολοένα και πιο δημοφιλής.

1.2 Δομή της διπλωματικής εργασίας

Πιο κάτω περιγράφεται πολύ συνοπτικά το περιεχόμενων των επόμενων κεφαλαίων.

Κεφάλαιο 2: Σε αυτό το κεφάλαιο αναλύεται το πρώτο βασικό στοιχείο της εφαρμογής που είναι η αλληλεπίδραση. Γίνεται μια αναφορά σχετικά με τον όρο αλληλεπίδραση ανθρώπου υπολογιστή , την φωνητική αλληλεπίδραση καθώς επίσης και σύγκριση δυνατοτήτων της εφαρμογής SAG με παρόμοιες εφαρμογές.

Κεφάλαιο 3: Σε αυτό το κεφάλαιο γίνεται μια περιγραφή σχετικά με την έννοια έξυπνο . Πιο συγκεκριμένα προσδιορίζεται ο όρος και παρουσιάζεται η μετέπειτα εξέλιξη του (Internet of Things – IoT).

Κεφάλαιο 4: Το κεφάλαιο 4 αναλύει το δεύτερο συστατικό στοιχείο της εφαρμογής που είναι το «έξυπνο» σπίτι το οποίο είναι ένα σύστημα που αποτελείται από IoT . Επίσης παρουσιάζονται τα πλεονεκτήματα που παρέχει ένα «έξυπνο» σπίτι.

Κεφάλαιο 5: Στο κεφάλαιο 5 αναλύεται το μέσο με το οποίο επιτυγχάνεται η επικοινωνία μεταξύ του χρήστη και του «έξυπνου» σπιτιού , δηλαδή το «έξυπνο» κινητό τηλέφωνο. Αναφέρονται διάφορα λειτουργικά συστήματα τα οποία

χρησιμοποιούν τα «έξυπνα» κινητά τηλέφωνα καθώς και η αρχιτεκτονική Android στην οποία στηρίζεται η εφαρμογή SAG.

Κεφάλαιο 6: Στο κεφάλαιο αυτό παρουσιάζεται ο τρόπος με τον οποίο θα υλοποιηθεί η εφαρμογή SAG, δηλαδή:

- το πρότυπο που θα χρησιμοποιηθεί (το πρότυπο Command το οποίο χρησιμοποιείται για την εκτέλεση των εντολών που θα δίνει ο χρήστης είτε μέσω φωνητικής εντολής είτε μέσω της χρήσης κουμπιών της εφαρμογής.)
- ο καθορισμός των εντολών και των σεναρίων που θα παρέχει η εφαρμογή (οι βασικές και σύνθετες εντολές)
- τα πρωτόκολλα επικοινωνίας που έχουν υλοποιηθεί (REST για την πλατφόρμα Smart Things, JAVA SDK για την πλατφόρμα Wunderbar).
- η χρήση των συνδετικών συστατικών υποσυστημάτων που παρέχει ευελιξία και απλότητα στην εφαρμογή
- οι απαιτήσεις της εφαρμογής όπως πλατφόρμα, μικρόφωνο, σύνδεση στο διαδίκτυο κ.α.
- ο τρόπος λειτουργίας της εφαρμογής μέσω εικόνων και οδηγιών.
- Σύγκριση της εφαρμογής με άλλες παρόμοιες εφαρμογές

Για την καλύτερη κατανόηση της διπλωματικής εργασίας παρουσιάζονται οι συντομογραφίες που χρησιμοποιούνται στα κεφάλαια και οι αντίστοιχες ερμηνείες τους:

SAG:	Smart Assistant Guardian
SP:	Smart Phone
SH:	Smart Home
IoT:	Internet of Things
GPS:	Global Position System
HSPA:	High Speed Packet Access

NFC:	Near Field Communication
LTE:	Long Term Evolution
API:	Application Programming Interface
MQTT:	Message Queuing Telemetry Trasport
IFTTT	If this then that

Κεφάλαιο 2

Αλληλεπίδραση ανθρώπου υπολογιστή

2.1 Ορισμός	7
2.2 Φωνητική αλληλεπίδραση	8
2.3 Αναγνώριση φωνής	9
2.4 Υλοποίηση φωνητικής αλληλεπίδρασης στην εφαρμογή	11

2.1 Ορισμός

Η αλληλεπίδραση μεταξύ ανθρώπου- υπολογιστή είναι κλάδος της πληροφορικής που όπως υποδηλώνει και το όνομα του ασχολείται με την αλληλεπίδραση ανθρώπου με υπολογιστή. Απαραίτητοι παράγοντες που καθιστούν εφικτή την κατανόηση και δημιουργία μιας επιθυμητής αλληλεπίδρασης μεταξύ αυτών των 2 οντοτήτων(άνθρωπος και μηχανή) είναι η πληροφορική, η γνωστική και κοινωνική ψυχολογία, η γλωσσολογία ,ο βιομηχανικός σχεδιασμός και άλλα γνωστικά πεδία. Η αλληλεπίδραση επιτυγχάνεται μέσω κάποιας διεπαφής χρήστη η οποία καθορίζεται μέσα από κάποιο λογισμικό και υλικό[12].Μια άλλη ερμηνεία της Αλληλεπίδρασης Ανθρώπου –Υπολογιστή είναι «η εύρεση διαφόρων τρόπων ούτως ώστε να κατασκευαστούν συστήματα, υπολογιστές και πολύπλοκες υπηρεσίες που παρέχουν πληροφορίες , τα οποία να είναι προσβάσιμα, εύχρηστα και αποδεκτά από ένα μη έμπειρο χρήστη»[4].

Για να αποδεχθεί ο χρήστης ένα σύστημα θα πρέπει να συμπεράνει κατά πόσο αυτό είναι εύχρηστο ή όχι. Δηλαδή αν μπορεί εύκολα να ενεργήσει με το σύστημα χωρίς να παρουσιάζεται δυσκολία στην εκμάθηση λειτουργίας του τότε αποδέχεται το σύστημα εφόσον εξυπηρετεί τις ανάγκες του εύκολα και γρήγορα αυξάνοντας έτσι την

αξιοπιστία και του συστήματος. Σε διαφορετική περίπτωση ο χρήστης θα απορρίψει το σύστημα και θα προτιμήσει διαφορετικό σύστημα που εξυπηρετεί τις ανάγκες του.



Σχήμα 2.1 Αλληλεπίδραση με οθόνη αφής

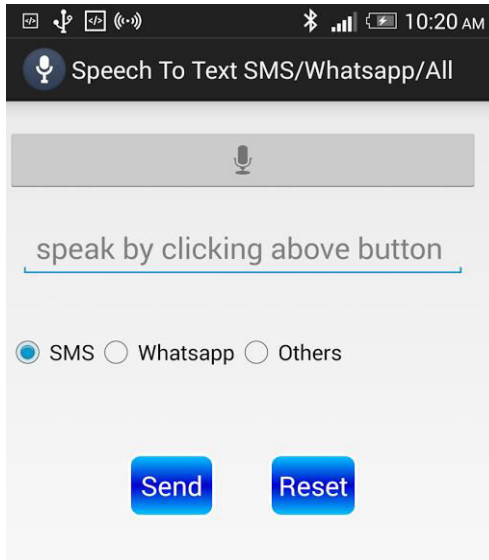


Σχήμα 2.2 Αλληλεπίδραση με γυαλιά εικονικής πραγματικότητας

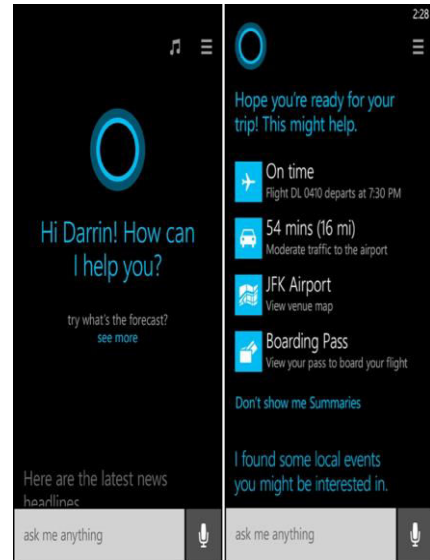
2.2 Φωνητική αλληλεπίδραση

Η φωνητική αλληλεπίδραση αποτελεί ένα διαφορετικό τρόπο με τον οποίο μπορούμε να επιτύχουμε αλληλεπίδραση με κάποιο σύστημα. Ενώ προηγουμένως η αλληλεπίδραση εκτελούνταν με αντίστοιχες ενέργειες αφής(πληκτρολόγιο, ποντίκι, οθόνη αφής) με την φωνητική αλληλεπίδραση το μέσο με το οποίο εκτελούνται διάφορες ενέργειες είναι η φωνή. Με την χρήση κατάλληλου κωδικοποιητή η μηχανή μπορεί να μεταφράσει το αναλογικό μας σήμα(φωνή) σε ψηφιακό και έπειτα από κάποια επεξεργασία να παράγεται κάποιο αποτέλεσμα ανάλογα με το πρόγραμμα που χρησιμοποιεί κάποιο πρόγραμμα αναγνώρισης φωνής. Ανάλογα με το πρόγραμμα αναγνώρισης φωνής μπορούμε να εκτελέσουμε απλές πράξεις όπως για παράδειγμα στην Σχήμα 2.3 βλέπουμε μια απλή εφαρμογή με την οποία μπορούμε να δώσουμε

δεδομένα(σε αναλογική μορφή δηλαδή ομιλία) και να δούμε το αντίστοιχο κείμενο ή ακόμη και να εκτελέσουμε πιο σύνθετες ενέργειες όπως άνοιγμα κάποιας εφαρμογής, κλήση ενός αριθμού, περιήγηση στο διαδίκτυο κ.α.



Σχήμα 2.3 Εφαρμογή speech to text



Σχήμα 2.4 Εφαρμογή Cordana Voice Assistant

2.3 Αναγνώριση φωνής

Η αναγνώριση φωνής μπορεί να γίνει με ή χωρίς πρόσβαση στο διαδίκτυο. Η εταιρεία Google έχει αρχίσει να δοκιμάζει ένα νέο σύστημα αναγνώρισης φωνής χωρίς να χρειάζεται σύνδεση στο διαδίκτυο. Υποστηρίζεται πως αυτό το σύστημα είναι αρκετά πιο γρήγορο σε σύγκριση με την αναγνώριση φωνής μέσω σύνδεσης(online) ενώ παράλληλα λόγω της ιδιότητας του εξαλείφει την εξάρτηση σύνδεσης για την παράξη αποτελεσμάτων. Δηλαδή αυτό το σύστημα μπορεί να χρησιμοποιηθεί χωρίς κάποια σύνδεση(offline) και έτσι ο χρήστης μπορεί να δίνει φωνητικές εντολές και να παράγεται το επιθυμητό αποτέλεσμα. Χρησιμοποιώντας όμως την τοπική αναγνώριση

φωνής (offline) έχουμε το μειονέκτημα πως το SP χρειάζεται να αναλάβει περισσότερο φόρτο εργασίας εφόσον τώρα χρειάζεται να εκτελέσει την διαδικασία αναγνώρισης κάτι που προηγουμένως γινόταν σε κάποιο εξυπηρετητή (datacenter).Επίσης χρειάζεται περισσότερη μνήμη λόγω της αποθήκευσης του συστήματος στην συσκευή έτσι ώστε να μπορεί να εκτελεί την διαδικασία αναγνώρισης φωνής.[17].

Στο λειτουργικό σύστημα Android έχει εγκατασταθεί και χρησιμοποιείται η αναγνώριση φωνής για περιήγηση στο διαδίκτυο.Για παράδειγμα στο σχήμα 2.5 βλέπουμε το εικονίδιο του μικροφώνου στην δεξιά πλευρά της μπάρας Google. Όταν ο χρήστης πατήσει αυτό το κουμπί τότε ενεργοποιείται η εγγραφή φωνής έτσι ώστε να αποθηκευτεί η φωνητική εντολή και να πραγματοποιηθεί αναζήτηση ιστοσελίδων όπως θα πραγματοποιούνταν σε περίπτωση που θα πληκτρολογούσαμε το κείμενο που έχουμε πει στην φωνητική εντολή.



Σχήμα 2.5 Μπάρα Google Search η οποία εμφανίζεται σε Android SP

Η αναγνώριση φωνής έχει γίνει γνωστή και μέσα από τα λειτουργικά συστήματα SP. Τα λειτουργικά συστήματα πλέον έχουν προεγκατεστημένη κάποια εφαρμογή αναγνώρισης φωνής από αποτελεί φωνητικό βοηθό του χρήστη.Για παράδειγμα ο φωνητικός βοηθός μπορεί να εκτελέσει άνοιγμα κάποιας εφαρμογής, φωνητικό μήνυμα για κάποια υπενθύμιση, ενημέρωση σχετικά με τον καιρό, τελευταίες ταινίες που έχουν κυκλοφορήσει κ.α. Στον πιο κάτω πίνακα παρουσιάζονται χαρακτηριστικά φωνητικών βοηθών για διάφορα λειτουργικά συστήματα έξυπνων κινητών τηλεφώνων. Με φωσφορούχο χρώμα φαίνονται οι φωνητικοί βοηθοί των λειτουργικών συστημάτων που αναφέρονται στο υποκεφάλαιο 5.2 . Πιο συγκεκριμένα το Blackberry 10 ανήκει στο OS Blackberry, η Cordana στο OS Windows 10 Mobile,ο Siri στο OS iOS και το Google Now στο OS Android . Σύμφωνα με τον πίνακα μέχρι τώρα το μόνο λειτουργικό σύστημα που υποστηρίζει ενσωμάτωση με IoT είναι το Android.

Intelligent personal assistant	Availability	Free software	Free and open-source hardware	HDMI out	External I/O	Retail price	Internet of things integration	Chromecast integration	Smart phone app	Always on	Unit to unit voice channel	Official website
Assistant	Play Store	No	N/A	N/A	N/A	—	No	No	Yes	No	N/A	Link
Aido	not released	No	N/A	N/A	N/A	US\$549	Yes	No	Yes	Yes	N/A	Link
BlackBerry Assistant	BlackBerry 10	No	N/A	N/A	N/A	—	No	No	Yes	No	N/A	Link
Braina	Windows	No	N/A	N/A	N/A	—	No	No	Yes	No	N/A	Link
Cortana	multiple	No	N/A	N/A	N/A	—	No	No	Yes	No	N/A	Link
Cubic	not released	No	N/A	N/A	N/A	US\$195	Yes	No	Yes	No	N/A	Link
Echo	shipping	No	No	No	No	US\$179	Yes	No	Yes	Yes	?	Link
Google Now	Play Store	No	N/A	N/A	N/A	—	Yes	Yes	Yes	Yes	N/A	Link
Hound	private beta	No	N/A	N/A	N/A	—	No	No	Yes	Yes	N/A	Link
Jibo	not released	No	N/A	N/A	N/A	US\$749	Yes	No	Yes	Yes	N/A	Link
Maluuba	not released	No	N/A	N/A	N/A	—	No	No	Yes	No	N/A	Link
Mycroft	not released	Yes	Yes	Yes	Yes	US\$154	Yes	Yes	Yes	Yes	Yes	Link
Nina	web-based	No	N/A	N/A	N/A	—	No	No	Yes	No	N/A	Link
SILVIA	API	No	N/A	N/A	N/A	—	No	No	Yes	No	N/A	Link
Siri	App Store	No	N/A	N/A	N/A	—	No	No	Yes	No	N/A	Link
Sirius	GitHub	Yes	N/A	N/A	N/A	—	No	No	Yes	No	N/A	Link
Ubi	API	No	N/A	N/A	N/A	US\$399	Yes	No	Yes	Yes	N/A	Link
Vlingo	multiple	No	N/A	N/A	N/A	—	No	No	Yes	No	N/A	Link
Your.MD	multiple	No	N/A	N/A	N/A	-	No	No	Yes	No	N/A	Link

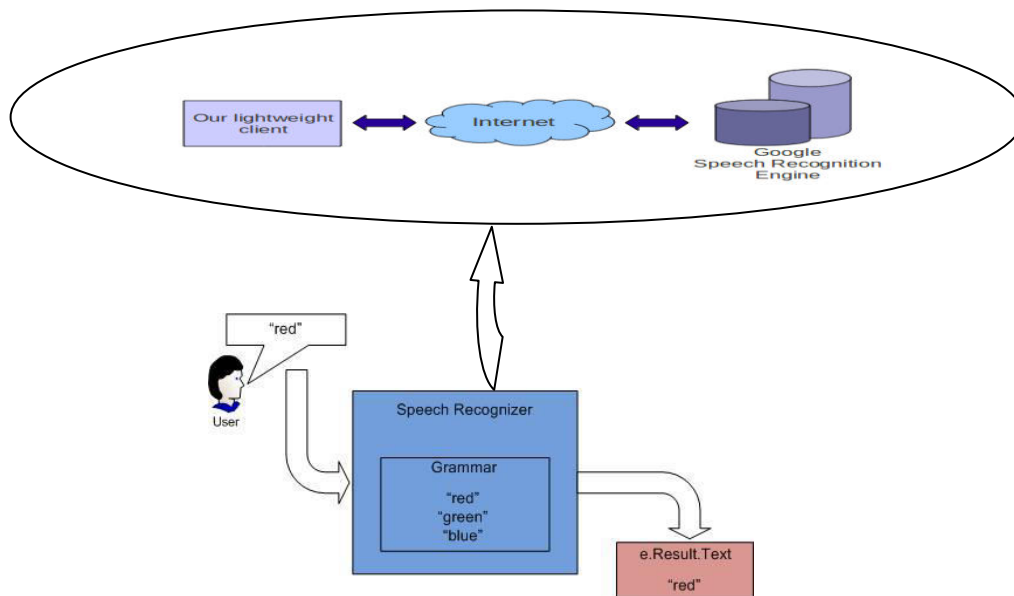
Πίνακας 2.6 Φωνητικοί βοηθοί (voice assistants) και χαρακτηριστικά τους

Η αναγνώριση φωνής, λοιπόν, μπορεί να εκτελεί διάφορες εντολές (εφόσον η εφαρμογή υποστηρίζει αυτή την δυνατότητα) που μπορεί να είναι απλές μεταφράσεις της ομιλίας σε κείμενο αλλά και ποιο σύνθετες εντολές όπως εύρεση ιστοσελίδων , υπενθύμιση για κάποιο γεγονός κ.α.

2.4 Υλοποίηση φωνητικής αλληλεπίδρασης στην εφαρμογή

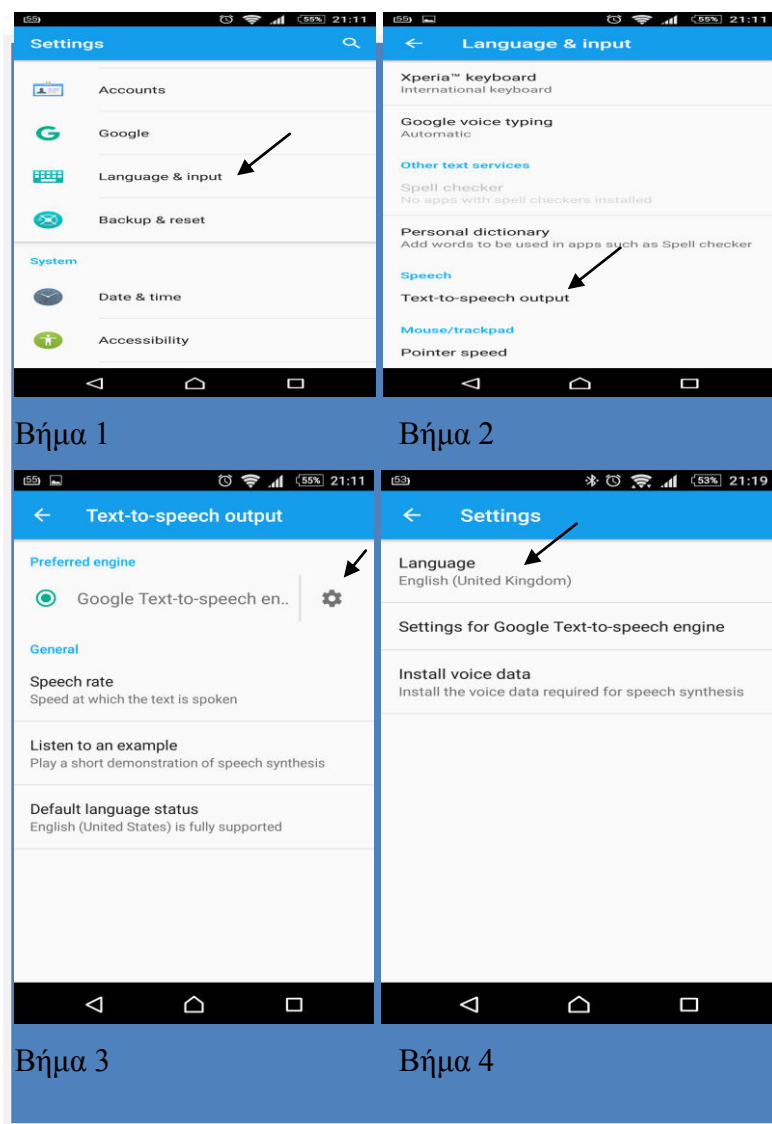
Στην εφαρμογή χρησιμοποιώ την κλάση SpeechRecognizer έτσι ώστε να επιτυγχάνεται αναγνώριση ομιλίας. Για να μπορεί να πραγματοποιηθεί η αναγνώριση θα πρέπει ο χρήστης να έχει πρόσβαση στο διαδίκτυο και να επιτρέψει στην εφαρμογή να χρησιμοποιήσει το μικρόφωνο του SP (record audio και internet permissions) εφόσον η ομιλία που έχει ηχογραφηθεί θα πρέπει να σταλεί σε κάποιους εξυπηρετητές που είναι υπεύθυνοι για αυτή την διαδικασία.[18] Ακόμη χρησιμοποιώ την διεπαφή RecognitionListener η οποία έχει εισαχθεί στο API level 8. Η διεπαφή επιτρέπει στο

πρόγραμμα να λαμβάνει ειδοποιήσεις σχετικά με την αναγνώριση φωνής. Επίσης έχω συνδυάσει την διεπαφή RecognitionListener με progress bar έτσι ώστε ο χρήστης να μπορεί να αντιλαμβάνεται ότι εκτελείται κάποια διαδικασία(αναγνώριση φωνής). [19]



Σχήμα 2.7 Τρόπος λειτουργίας του Speech Recognizer

Η εφαρμογή θα πρέπει επίσης να παράγει φωνητικά μηνύματα στην περίπτωση που ο χρήστης έχει δώσει κάποια εντολή έτσι ώστε να καλύπτεται και η βασική ανάγκη της εφαρμογής που στηρίζεται στην φωνητική αλληλεπίδραση. Για να το επιτύχουμε αυτό θα χρειαστεί να εγκαταστήσουμε στο SP την εφαρμογή Google Text to speech. Αν η έκδοση του SP είναι Jelly Bean θα πρέπει να εγκαταστήσουμε αυτή την εφαρμογή διαφορετικά σε οποιαδήποτε νεότερη έκδοση είναι ήδη εγκατεστημένη στο λειτουργικό σύστημα Android και θα πρέπει να την ενεργοποιήσουμε. Ο χρήστης θα πρέπει λοιπόν να εγκαταστήσει την εφαρμογή και έπειτα να καθορίσει την γλώσσα United Kingdom. Πιο κάτω βλέπουμε την διαδικασία που χρειάζεται να γίνει έτσι ώστε να ενεργοποιήσουμε την εφαρμογή Text to speech.



Σχήμα 2.8 Διαδικασία ρύθμισης Text To Speech

Κεφάλαιο 3

Έξυπνο *

3.1 Ορισμός	13
3.2 Internet of Things	17

3.1 Ορισμός

Με τον όρο smart εννοούμε την «έξυπνάδα» που μπορεί να χρησιμοποιηθεί σε μια συσκευή έτσι ώστε να μην περιορίζεται μόνο στην βασική λειτουργία της ,αλλά λόγω του μεγέθους και των χαρακτηριστικών της συσκευής να μπορεί να παρέχει επιπλέον δυνατότητες που να εξυπηρετούν περισσότερες ανάγκες του χρήστη. Αυτό επιτυγχάνεται με την ενσωμάτωση διαφόρων συστατικών στοιχείων όπως επεξεργαστές ,σχεδίαση κατάλληλου κυκλώματος , αισθητήρες και ό,τι άλλο είναι αναγκαίο για την λειτουργικότητα της συσκευής όπως πρωτόκολλα επικοινωνίας, αλγόριθμοι κ.α.

Τα τελευταία χρόνια παρατηρείται ενδιαφέρον και μεγάλη ανάπτυξη σε αυτό τον καινοτόμο τομέα ο οποίος ειδικεύεται στην δημιουργία διαφόρων συσκευών που έχει σαν στόχο να παρέχει διάφορες υπηρεσίες στον άνθρωπο .Οι επιστήμονες έχοντας κατανοήσει την λειτουργία των συσκευών έχουν επιτύχει να προχωρήσουν ακόμα ένα βήμα παραπέρα με αποτέλεσμα να κατασκευάσουν «έξυπνες» συσκευές οι οποίες χαρακτηρίζονται από «έξυπνάδα». Καθημερινά χρησιμοποιούμε διάφορες συσκευές οι οποίες διευκολύνουν και βελτιώνουν τρόπο ζωής . Ένα παράδειγμα μιας τέτοιας συσκευής είναι και το έξυπνο κινητό τηλέφωνο(smartphone). Ενώ προηγουμένως τα

κινητά τηλέφωνα είχαν ένα και μόνο σκοπό, την επικοινωνία 2 ανθρώπων που βρίσκονταν σε διαφορετική τοποθεσία, τώρα έχουν μετατραπεί σε συσκευές όπου με την βοήθεια διαφόρων αισθητήρων που είναι ενσωματωμένοι σε αυτές παρέχουν επιπλέον δυνατότητες όπως για παράδειγμα εύρεση τοποθεσίας του χρήστη, φωτογραφική μηχανή, υποστήριξη πρωτοκόλλων για ανταλλαγή δεδομένων (Bluetooth, NFC, infrared, LTE) αισθητήρες κίνησης κ.α.

Στις πιο κάτω εικόνες φαίνονται διάφορα SP και οι δυνατότητες τους:

Specs	Nexus 6P	Nexus 5X	iPhone 6s	Galaxy S6
Display	5.7-inch WQHD Super AMOLED	5.2 inch with FHD LCD	4.7-inch Retina Display	5.1-inches QHD Super AMOLED
Processor	Snapdragon 810 8-core	Snapdragon 808 6-core	Apple A9 chip	Exynos 64-bit Octa Core
RAM	3GB	2GB	2GB	3GB RAM
Storage Space	32,64,128 GB	16,32GB, no microSD	16,64,128GB	32 GB
Camera Rear	12.3 mp with f/2.0	12.3 mp with f/2.0	12 mp with f/2.2	16MP
Camera Front	8 mp with f.24	5 mp with f/2.0	5 mp with f/2.2	5MP
OS	Android 6.0 Marshmallow	Android 6.0 Marshmallow	iOS 9	Android 5.0 Lollipop
Battery	3,450 mAh	2,700 mAh	1715 mAh (14 hr on 3G)	2,550mAh
Fingerprint Sensor	Yes	Yes	Yes called Touch ID	Yes
Connectivity	Wi-Fi 802.11a/b/g/n/ac 2x2 MIMO	Wi-Fi 802.11a/b/g/n/ac 2x2 MIMO	Wi-Fi 802.11a/b/g/n/ac with MIMO	Wi-Fi: 802.11 a/b/g/n/ac with MIMO
LTE, NFC	NFC, LTE TDD and FDD	NFC, LTE TDD and FDD	NFC, LTE TDD	NFC, LTE FDD and TDD
Charging Port	Type-C USB	Type-C USB	Lightning Conductor	Micro-USB charging
Dimensions	159.3 X 77.8 X 7.3mm	147.0 x 72.6 x 7.9 mm	138.3 x 67.1 x 7.1mm	143.4 x 70.5 x 6.8
Price in India	Rs 39,999 for 32GB	Rs 31,900 for 16GB	TBA	Rs 41,900

Σχήμα 3.1 Χαρακτηριστικά διαφόρων «έξυπνων» κινητών τηλεφώνων

							
Operating System	Android 4.2.1 Jelly Bean	iOS 6	Android 4.2.2 Jelly Bean	Android 4.1 Jelly Bean	BlackBerry 10	Samsung Galaxy Note II	Android 4.1.2 Jelly Bean
Display	4.9" 1920 x 1080 (441 PPI)	4" 1136 x 640 Retina Display (326 PPI)	4.7" 1280 x 768 (316 PPI)	4.7" 1920 x 1080 (467 PPI)	4.2" 1280 x 768 (356 PPI)	5.5" 1280 x 720 (267 PPI)	5" 1920 x 1080 (443 PPI)
Processor	1.8 GHz eight-core Exynos 5	1.3GHz dual-core A6	1.5GHz quad-core Qualcomm Snapdragon S4 Pro	1.7GHz quad-core Snapdragon 600	1.5GHz dual-core Qualcomm Snapdragon S4 Plus	1.6GHz quad-core Exynos 4412 Quad	1.5GHz quad-core Qualcomm Snapdragon S4 Pro
RAM	2GB	1GB	2GB	2GB	2GB	2GB	2GB
Storage	16/32/64GB	16/32/64GB	8/16GB	32/64GB	16GB + microSD slot	16/32/64GB	16GB + microSD slot
Camera	13MP	8MP (rear) with f2.4 + 1.3MP (front)	8MP	4MP UltraPixel	8MP	8MP	13.1MP
Dimensions	140.1 x 71.8 x 7.7mm	123.83 x 58.57 x 7.6 mm	133.9 x 68.7 x 9.1 mm	134.9 x 68.1 x 9.14mm	130 x 66 x 9.3mm	151.1 x 80.5 x 9.4mm	139 x 71 x 7.9mm
Weight	138g	112g	139g	143g	136g	183g	146g
Battery	2,600mAh	1,440 mAh, 3.8V	2,100 mAh, 3.8V	2,300 mAh	1,800 mAh	3,100 mAh, 3.8V	2,400 mAh

Σχήμα 3.2 Χαρακτηριστικά «έξυπνων» κινητών τηλεφώνων

Αυτή η καινοτόμος ιδέα δεν περιορίζεται μόνο στην δημιουργία μιας συσκευής η οποία παρέχει διαφορετικές δυνατότητες για να καλύψει περισσότερες ανάγκες του χρήστη(εκτός από την βασική λειτουργία της συσκευής) αλλά μπορεί και να χρησιμοποιηθεί και ως μέσο ειδοποίησης για κάποια κατάσταση. Πιο συγκεκριμένα έχουν κατασκευαστεί smart watches τα οποία όχι μόνο δείχνουν την ώρα και ημερομηνία αλλά περιέχουν και διάφορους αισθητήρες οι οποίοι επιτρέπουν την μέτρηση των παλμών της καρδιάς, την μέτρηση των κινήσεων που εκτελεί ο χρήστης έτσι ώστε να υπενθυμίζουν(από το smart watch μέσω φωνητικής αλληλεπίδρασης) στο χρήστη να κινηθεί έτσι ώστε να βελτιώσει την φυσική του κατάσταση κ.α. Με αυτό τον τρόπο επιτυγχάνεται έγκαιρη αντιμετώπιση προβλημάτων υγείας π.χ. ένας άνθρωπος ο οποίος έχει κάποιο πρόβλημα με την καρδιά του μπορεί να δει τον ρυθμό παλμών της καρδιάς και να ενεργήσει ανάλογα προλαμβάνοντας καρδιακό επεισόδιο ή οποιοδήποτε άλλη ανεπιθύμητη κατάσταση.



Σχήμα 3.3 Smart watch που παρουσιάζει τις δραστηριότητες σχετικά με τα την άσκηση που έχει εκτελέσει ο χρήστης

Δύο πλεονεκτήματα που αποτελούν κίνητρα για την «εξέλιξη» του κινητού τηλεφώνου και του ρολογιού σε «έξυπνες» συσκευές είναι τα εξής:

1. το μικρό μέγεθος - μεταφερσιμότητα. Το μέγεθος ενός κινητού βοηθά στην εύκολη μεταφορά εφόσον μπορούμε να το φυλάξουμε στην τσέπη ,ενώ στην περίπτωση του ρολογιού το φορούμε στο χέρι χωρίς να μας εμποδίζει σε οποιαδήποτε κίνηση και αν κάνουμε. Επίσης έχουν και τα 2 μικρό βάρος το οποίο βοηθά επίσης στην καλύτερη μεταφορά.

2. Συνδεσιμότητα. Λόγω της βασικής ιδιαιτερότητας του κινητού τηλεφώνου να λαμβάνει λήψη σημάτων και την ανάπτυξη νέων τεχνολογιών (4G) καθιστά εφικτή την συνεχή σύνδεση στο διαδίκτυο με αποτέλεσμα να παρέχει επιπλέον δυνατότητες στην συσκευή όπως εύρεση τοποθεσίας, χρήση διαδικτύου(online συναλλαγές, παρακολούθηση βίντεο ,χρήση μέσων κοινωνικής δικτύωσης κ.α.).

Η δημιουργία «έξυπνων» συσκευών δεν περιορίζεται μόνο σε ατομικές συσκευές όπως τις 2 προηγούμενες που έχουν αναφερθεί άλλα και σε συσκευές που χρησιμοποιούν πολλοί άνθρωποι. Μερικά παραδείγματα είναι:

- «έξυπνη» τηλεόραση. Μερικές εταιρείες άρχισαν να κατασκευάζουν τηλεοράσεις οι οποίες υποστηρίζουν εκτός από την μετάδοση τηλεοπτικών καναλιών και σύνδεση στο διαδίκτυο μετατρέποντας την απλή τηλεόραση σε ένα είδος υπολογιστή με τον οποίο μπορείς δώσεις εντολές(πληκτρολόγηση κειμένου , κίνηση κέρσορα) μέσω του τηλεχειριστηρίου. Ακόμη παρέχεται η δυνατότητα της ταυτόχρονης θέασης όπου ο χρήστης μπορεί να βλέπει δύο διαφορετικά πράγματα ταυτόχρονα π.χ. η τηλεόραση μπορεί να διαχωριστεί σε δύο μέρη όπου στο πρώτο ο χρήστης μπορεί να παρακολουθεί μια ταινία ενώ στο δεύτερο μέρος κάποιος άλλος χρήστης να βλέπει μια ιστοσελίδα.



Σχήμα 3.4 «Έξυπνη» τηλεόραση

- «έξυπνο» ψυγείο. Στο εσωτερικό του ψυγείου υπάρχουν κάμερες με τις οποίες ο χρήστης μπορεί να δει τι περιέχει το ψυγείο οποιαδήποτε στιγμή όπου και αν βρίσκεται με την βοήθεια εφαρμογής που υπάρχει για τα smartphones. Επίσης στην πόρτα του ψυγείου υπάρχει εγκατεστημένη τηλεόραση η οποία προσφέρει

ψυχαγωγία στην κουζίνα καθώς επίσης και εξοικονόμηση χώρου εφόσον δεν χρειάζεται να υπάρχει τηλεόραση σε κάποιο άλλο σημείο της κουζίνας.



Σχήμα 3.5 «Έξυπνο» ψυγείο

3.2 Internet of Things

Στο προηγούμενο υποκεφάλαιο υπήρξε αναφορά σχετικά με τον τρόπο με τον οποίο μπορούμε να δημιουργήσουμε «έξυπνες» συσκευές. Λαμβάνοντας υπόψη διάφορα χαρακτηριστικά όπως μέγεθος, λειτουργικότητα, την ενσωμάτωση διαφόρων συστατικών στοιχείων όπως επεξεργαστών καθώς επίσης διάφορα πρωτόκολλα και αλγορίθμους επιτυγχάνεται η κατασκευή αυτών των καινοτόμων συσκευών. Μπορεί να μην είναι εμφανές όμως οι «έξυπνες» συσκευές είναι στην πραγματικότητα διαφορετικές υπολογιστικές μηχανές οι οποίες ανάλογα με κάποια είσοδο επεξεργάζονται δεδομένα και παράγουν στην έξοδο ένα αποτέλεσμα:

- Το «έξυπνο» τηλέφωνο μέσω του GPS receiver εμφανίζει στον χρήστη την τοποθεσία στην οποία βρίσκεται
- Το «έξυπνο» ρολόι διαθέτει αισθητήρες που μετρούν τους χτύπους της καρδιάς και εμφανίζει στην οθόνη το αποτέλεσμα ή ακόμη και την πραγματοποίηση online συναλλαγών.

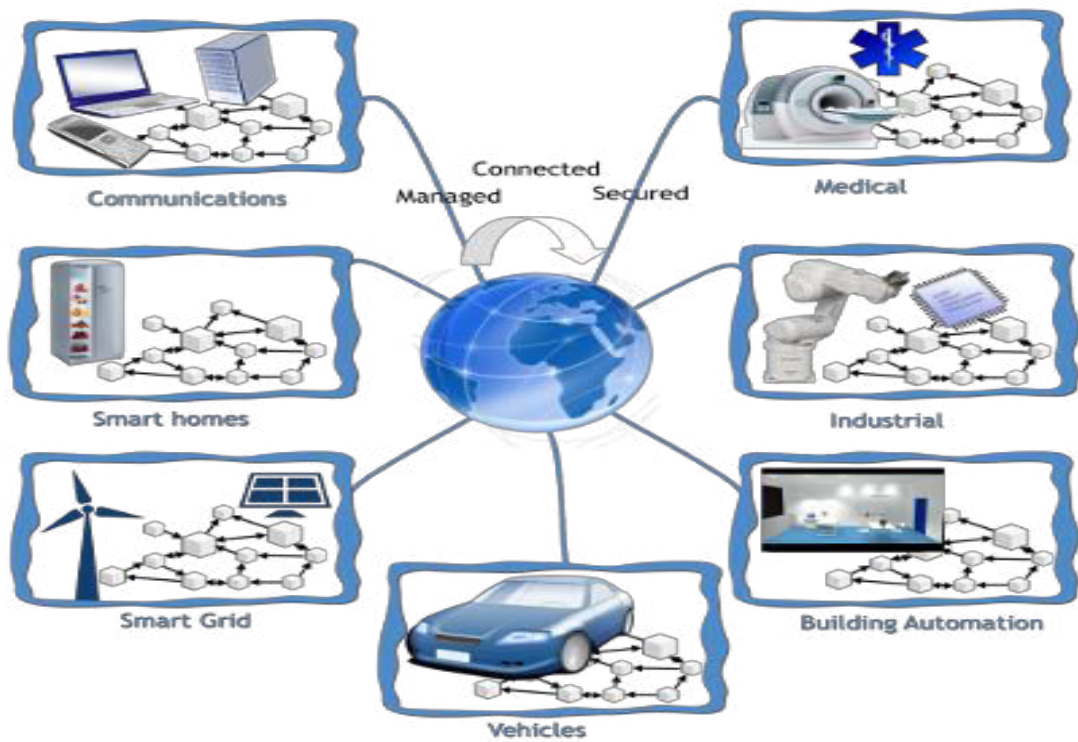
- Το «έξυπνο ψυγείο» «στέλλει» μέσω του διαδικτύου την εικόνα, που απεικονίζει τι περιέχει το ψυγείο, όταν ο χρήστης χρησιμοποιήσει την κινητή εφαρμογή που έχει δημιουργηθεί για να μπορεί να επικοινωνεί με το «έξυπνο» ψυγείο.
- Η «έξυπνη» τηλεόραση θα πρέπει να στείλει την εντολή, που έχει εκτελέσει ο χρήστης , στο μόντεμ έτσι ώστε να βρεθεί και να επιστραφεί το περιεχόμενο της ιστοσελίδας που έχει ζητηθεί.

Το πιο σημαντικό κοινό στοιχείο που κατέχουν οι συσκευές είναι η επικοινωνία μεταξύ τους. Τα πιο πάνω παραδείγματα δεν διαφέρουν και πολύ από το τον τρόπο επικοινωνίας που χρησιμοποιούν οι ηλεκτρονικοί υπολογιστές όταν χρειάζονται κάποιες πληροφορίες. Όταν χρειαζόμαστε κάποια πληροφορία τότε χρησιμοποιούμε ένα φυλλομετρητή και με την βοήθεια του Διαδικτύου βρίσκουμε την πληροφορία που μας ενδιαφέρει. Με παρόμοιο τρόπο οι «έξυπνες» συσκευές συνδέονται μεταξύ τους έτσι ώστε να ανταλλάζουν πληροφορίες και να παράγουν το κατάλληλο αποτέλεσμα το οποίο μπορεί να είναι μια ενέργεια , μια ειδοποίηση ή μια εικόνα. Αυτή η δυνατότητα επικοινωνίας είναι πηγή έμπνευσης για μια καινούρια έννοια, που με τον καιρό γίνεται όλο και πιο δημοφιλής , γνωστή ως Internet of Things.

Με τον όρο IoT εννοούμε τις «έξυπνες» συσκευές που μπορούν να συνδέονται στο διαδίκτυο έτσι ώστε να επικοινωνούν με άλλες «έξυπνες» συσκευές με στόχο την ανταλλαγή δεδομένων για να δημιουργία ενός αποτελέσματος. Γύρω στο 1982 δημιουργήθηκε η πρώτη μηχανή που είχε στόχο την καταγραφή αναψυκτικών και τον έλεγχο θερμοκρασίας έτσι ώστε να μπορούσε κάποιος να συμπεράνει αν τα τελευταία αναψυκτικά που έχουν εισαχθεί σε αυτό είναι παγωμένα. [13] Αυτή η τροποποιημένη μηχανή αναψυκτικών (Coke Machine) σηματοδότησε την έναρξη για την συσχέτιση μηχανή- συσκευή και διαδίκτυο.

Έχοντας ως πρότυπο αυτή την ιδέα , να υπάρχουν συσκευές οι οποίες να δημιουργούν δικό τους δίκτυο επικοινωνίας, όλο και περισσότεροι τομείς αρχίζουν να εφαρμόζουν αυτή την ιδέα :

- Αυτοματοποίηση σπιτιού:
χρησιμοποιώντας «έξυπνες» οικιακές συσκευές οι οποίες «συμπεριφέρονται» ανάλογα με το περιβάλλον ή με εντολές που δίνει ένας χρήστης «έξυπνου» σπιτιού.
- Αυτοματοποίηση αυτοκινήτου:
Χρησιμοποιώντας διάφορους «έξυπνους» αισθητήρες που μπορούν να ειδοποιήσουν τον οδηγό για κάποιο τεχνικό πρόβλημα ή ακόμη και να διευκολύνει ανθρώπους που δεν μπορούν να οδηγήσουν (google car).
- Υγεία:
Όπως έχει αναφερθεί προηγουμένως μπορούμε να έχουμε άμεση αντιμετώπιση σοβαρών προβλημάτων με την χρήση του έξυπνου ρολογιού. Επίσης απομακρυσμένη παρακολούθηση ασθενών και αισθητήρες που ανιχνεύουν κάποιο πρόβλημα στον ανθρώπινο οργανισμό είναι 2 τρόποι με τους οποίους επιτυγχάνεται άμεση ενημέρωση για την κατάσταση ενός ασθενή έτσι ώστε να ασκηθεί η κατάλληλη ιατρική/φαρμακευτική περίθαλψη.
- Παρακολούθηση περιβάλλοντος - «έξυπνες» πόλεις:
Χρησιμοποιώντας αισθητήρες που ανιχνεύουν την κυκλοφορία στους δρόμους, τον μέσο όρο διαφόρων αερίων στο περιβάλλον επιτυγχάνεται η πρόληψη ή και αντιμετώπιση σε περίπτωση πυκνής κίνησης στους δρόμους ή αυξημένη ποσότητα επικίνδυνων αερίων σε μια περιοχή.
- Αυτοματοποίηση βιομηχανίας :
χρησιμοποιώντας ένα «έξυπνο » συνδυασμό εργαλείων ,λογισμικού και υλικού επιτυγχάνεται καλύτερη και γρηγορότερη σχεδίαση μιας νέας βιομηχανικής εφαρμογής.



Σχήμα 3.6 Διάφοροι τομείς που «εκμεταλλεύονται» την δυνατότητα του IoT

Κεφάλαιο 4

Έξυπνο Σπίτι

4.1 «Έξυπνο» σπίτι	21
4.2 Πλεονεκτήματα «έξυπνου» σπιτιού	23

4.1 «Έξυπνο» σπίτι

Το προηγούμενο κεφάλαιο (Smart - IoT) αποτελεί βάση για την δημιουργία ενός «έξυπνου» σπιτιού. Έξυπνο σπίτι ονομάζουμε το σπίτι στο οποίο έχουν εγκατασταθεί διάφορες «έξυπνες» συσκευές που λειτουργούν με τέτοια μεθοδολογία έτσι ώστε να προσφέρουν διάφορες διευκολύνσεις και υπηρεσίες στους ανθρώπους που ζουν σε αυτό. [1]. Μπορούμε να πούμε πως το «έξυπνο» σπίτι, είναι ένα σύνολο από IoT το οποίο με βάση κάποιο πρωτόκολλο επικοινωνίας μπορεί να εκτελέσει μια εντολή ή και μια σειρά από καθορισμένες εντολές, για να δημιουργήσει μια νέα κατάσταση που επιθυμεί να υπάρχει σε μια συγκεκριμένη στιγμή ένας χρήστης. Στην προκειμένη περίπτωση χρήστη θεωρείται οποιοσδήποτε άνθρωπος ζει σε ένα έξυπνο σπίτι και έχει εξουσιοδότηση να δίνει εντολές (να αλληλεπιδρά) με τις «έξυπνες » συσκευές που έχουν εγκατασταθεί σε ένα SP.

Πιο κάτω ακολουθούν «έξυπνες» συσκευές – δυνατότητες που θα διαθέτει το SH :

1. Έλεγχο φωτισμού όλων των δωματίων: σύστημα με το οποίο μπορούμε να διαχειριστούμε τον φωτισμό οποιουδήποτε δωματίου
2. Έλεγχος θερμοκρασίας όλων των δωματίων-εξωτερικού χώρου: μέσω της χρήσης αισθητήρων θερμότητας επιτυγχάνεται καλύτερη διαχείριση του κεντρικού συστήματος κλιματισμού - θέρμανσης
3. Ηλιακά πλαίσια: με την χρήση ηλιακών πλαισίων μπορούμε να έχουμε καλύτερη διαχείριση ηλεκτρικής ενέργειας

4. Έξυπνο πλυντήριο: πλυντήριο το οποίο μπορεί ο χρήστης να ενεργοποιήσει με κατάλληλη εντολή , οποιαδήποτε χρονική στιγμή, έτσι ώστε να ξεκινήσει την λειτουργία του (δεν περιορίζεται μόνο από την δυνατότητα ενός χρονοδιακόπτη για εκκίνηση) .
5. Έξυπνο σύστημα home cinema: σύστημα ψυχαγωγίας το οποίο μπορεί να ενεργοποιήσει μέσω εντολής ο χρήστης (π.χ. ενεργοποίηση ηχοσυστήματος και έξυπνης τηλεόρασης).
6. Έξυπνο ρολόι : ρολόι το οποίο ρυθμίζεται ανάλογα με τις προτιμήσεις του εκάστοτε χρήστη.
7. Αυτόματο σύστημα γκαράζ: Αυτοματοποιημένο σύστημα γκαράζ το οποίο μπορεί να ανοίγει και να κλείνει την θύρα βασισμένο στην ώρα(π.χ. ο χρήστης Α καθημερινά επιστρέφει στις 16 άρα θα πρέπει στις 15:58 να ανοίξει η θύρα και να κλείσει στις 16:03).
8. Κάμερες ασφαλείας: Οι «έξυπνες» κάμερες ασφαλείας οι οποίες θα ειδοποιούν τον χρήστη ότι κάποιο μέλος της οικογένειας μόλις έχει φθάσει σπίτι ή ακόμη να «προκαλούν» ήχο σε περίπτωση που παρατηρηθεί ύποπτη κίνηση κάποιου ατόμου.
9. Σύστημα ασφαλείας για μεταφορικά μέσα (αυτοκίνητο, ποδήλατο)
10. Έξυπνος φούρνος: φούρνος ο οποίος μπορεί να λειτουργήσει «έξυπνα» και να «θυμάται» τον τρόπο ψησίματος φαγητών. [5]
11. Έξυπνο pop corn maker: «έξυπνη» συσκευή που «αντιλαμβάνεται» πότε είναι η στιγμή για παρακολούθηση ταινίας έτσι ώστε να ξεκινήσει την λειτουργία της.
12. Έξυπνο coffee maker: «έξυπνη» συσκευή που για παρασκευή καφέ είτε μέσω καθορισμένης ρουτίνας, είτε με την φωνητική εντολή που δίνει ο χρήστης.
13. Έξυπνο Toast maker: «έξυπνη» συσκευή η οποία φτιάχνει φρυγανιές είτε μέσω καθορισμένης ρουτίνας, είτε με την φωνητική εντολή που δίνει ο χρήστης.
14. Αισθητήρες κίνησης: αισθητήρες οι οποίοι ανιχνεύουν κίνηση και μπορούν να ενεργοποιήσουν κάμερες ασφαλείας ή να ειδοποιήσουν τον χρήστη για να ενεργήσει ανάλογα.

15. Κεντρικό σύστημα θέρμανσης : σύστημα το οποίο «λαμβάνει» μηνύματα από τους αισθητήρες θερμότητας ή από τον χρήστη και «ενεργεί» με καθορισμένο τρόπο έτσι ώστε να έχει όσο το δυνατό την καλύτερη δυνατή απόδοση.
16. Αισθητήρες φωτός για παρακολούθηση ηλιοφάνειας και κατάλληλη προσαρμογή στο εσωτερικό του έξυπνου σπιτιού.
17. Έξυπνο Σύστημα θέρμανσης νερού: σύστημα το οποίο «γνωρίζει» τις προτιμήσεις στην θερμοκρασία κάθε ατόμου και με «έξυπνο» τρόπο λειτουργεί με τέτοιο τρόπο έτσι ώστε να καταναλώσει όσο το δυνατό λιγότερη ενέργεια αλλά να ικανοποιεί όλους τους χρήστες.
18. Σύστημα ελέγχου και κίνησης για παράθυρα ,πόρτες και κουρτίνες : αισθητήρες και μηχανισμοί κίνησης οι οποίοι είναι ενσωματωμένοι στα παράθυρα και στις πόρτες έτσι ώστε να γίνεται έλεγχος για τυχόν παραβίαση ή για να ανοίγουν και να κλείνουν οι πόρτες , τα παράθυρα ακόμη και οι κουρτίνες ανάλογα με τα μηνύματα που λαμβάνουν από άλλες έξυπνες συσκευές ή από εντολή που έχει δώσει ο χρήστης.
19. Έξυπνη σκούπα: σκούπα η οποία «καταλαβαίνει» πως δεν υπάρχουν άτομα στο σπίτι και ξεκινά σκούπισμα του σπιτιού.

4.2 Πλεονεκτήματα «έξυπνου» σπιτιού

Με την βοήθεια των διαφόρων υπηρεσιών και διευκολύνσεων που παρέχονται από την χρήση ενός έξυπνου σπιτιού ο χρήστης επωφελείται σε 3 σημαντικούς τομείς. Πιο συγκεκριμένα οι 3 τομείς είναι:

- Αυτοματοποίηση
- Ασφάλεια
- Εξοικονόμηση ενέργειας

Αυτοματοποίηση:

Με την χρήση διάφορων «έξυπνων» συσκευών ο χρήστης μπορεί να δώσει εντολή με την χρήση του έξυπνου κινητού του τηλεφώνου ή με τον πίνακα ελέγχου των συσκευών και να μην χρειάζεται να εκτελεί ο ίδιος (χειροκίνητα με τον παραδοσιακό τρόπο) οποιαδήποτε ενέργεια.

Παραδείγματα:

1. Χρήση «έξυπνης» καφετιέρας . Ο χρήστης πρέπει μόνο να τοποθετήσει νερό και συστατικά που επιθυμεί στον καφέ του και να δώσει εντολή οποιαδήποτε στιγμή μέσω του κινητού του τηλεφώνου, οπουδήποτε και αν βρίσκεται (π.χ. έχει τελειώσει από την δουλειά και βρίσκεται στο δρόμο για το σπίτι), έτσι ώστε να ενεργοποιήσει την «έξυπνη» καφετιέρα .
2. Χρήση «έξυπνης» σκούπας. Όπως και στο προηγούμενο παράδειγμα ο χρήστης μπορεί να δώσει εντολή και να ξεκινήσει αυτόματο καθαρισμό . Έτσι παρέχεται η δυνατότητα να μπορεί να επιλέξει οποιαδήποτε ώρα επιθυμεί (ώρα στην οποία δεν θα υπάρχουν άτομα στο σπίτι) για να ξεκινήσει το καθαρισμό του σπιτιού από την «έξυπνη» σκούπα.

Τα 2 πιο πάνω παραδείγματα δεν παρέχουν μόνο αυτοματοποίηση διαφόρων ενεργειών στον χρήστη αλλά και περισσότερο ελεύθερο χρόνο εφόσον δεν χρειάζεται πλέον να εκτελεί ο ίδιος αυτές τις ενέργειες και έτσι να ασχολείται με άλλες δραστηριότητες.

Ασφάλεια:

Ένας αρκετά σημαντικός παράγοντας ο οποίος μπορεί αν εξασφαλιστεί με την χρήση «έξυπνων» αισθητήρων .

Παραδείγματα:

1. Χρήση «έξυπνων» αισθητήρων κίνησης: Οι αισθητήρες κίνησης ανιχνεύουν κίνηση σε μια καθορισμένη εμβέλεια και μπορούν να «ειδοποιούν» τον χρήστη σε περίπτωση που ο ίδιος επιθυμεί να ελέγχει μια περιοχή γύρω από το «έξυπνο» σπίτι του.
2. Χρήση «έξυπνων» καμερών ασφαλείας: Με τις κάμερες ασφαλείας που είναι ένας συνηθισμένος τρόπος για να ελέγχουμε το σπίτι, μπορούν επιπλέον να

αποτελέσουν και τρόπο πρόληψης διαρρήξεων. Αν για παράδειγμα η κάμερα ασφαλείας εντοπίσει, με την λειτουργία αναγνώρισης προσώπου , κάποιο άτομο το οποίο βρίσκεται σε μια «περίεργη» απόσταση από το «έξυπνο σπίτι» τότε θα ενεργοποιούνται διάφορες συσκευές έτσι ώστε να δίνεται η ψευδαίσθηση ότι κάποιος βρίσκεται στο σπίτι και να αποτρέψει μια πιθανή διάρρηξη. Πιθανές ενέργειες που μπορούν να δημιουργήσουν παρουσία κάποιου ατόμου στο σπίτι είναι η ενεργοποίηση ηχοσυστήματος/τηλεόρασης, αναπαραγωγή ηχογραφημένης συνομιλίας ατόμων κ.α.

Εξοικονόμηση ενέργειας:

Ένας ακόμη σημαντικός παράγοντας είναι και το κόστος ενέργειας που δαπανάται λόγω της χρήσης διάφορων ηλεκτρικών συσκευών. Με τις σωστές ρυθμίσεις των «έξυπνων» συσκευών μπορούμε να μετατρέψουμε το «έξυπνο» σπίτι σε πράσινο (οικολογικό) «έξυπνο» σπίτι.

Παραδείγματα:

1. Εγκαθιστώντας ένα «έξυπνο» μετρητή ενέργειας μπορούμε να αναλύσουμε την κατανάλωση ηλεκτρικής ενέργειας διαφόρων «έξυπνων» ηλεκτρικών συσκευών και να αποφασίσουμε πως μπορούμε να έχουμε ένα καλύτερο τρόπο διαχείρισης αυτών των συσκευών με σκοπό τόσο την εξοικονόμηση ενέργειας η οποία θα έχει σαν αποτέλεσμα την μείωση κόστους.[9]
2. Με την χρήση αισθητήρων φωτός μπορούμε να ελέγχουμε την φωτεινότητα και ανάλογα να ανοίγουν ή να κλείνουν τα παράθυρα έτσι ώστε να εξοικονομείται ενέργεια εφόσον ο φωτισμός θα προέρχεται από φυσικό φως και όχι από τους λαμπτήρες του «έξυπνου» σπιτιού.

Κεφάλαιο 5

«Έξυπνο» τηλέφωνο

5.1 Ορισμός	26
5.2 Λειτουργικά συστήματα κινητών	28
5.3 Αρχιτεκτονική Android	30

5.1 Ορισμός

Όπως έχει αναφερθεί και στο κεφάλαιο Smart το SP είναι μια έξυπνη συσκευή που δεν περιορίζεται μόνο στην τηλεφωνική επικοινωνία αλλά με την χρήση ενσωματωμένων αισθητήρων παρέχει επιπλέον δυνατότητες. Τώρα έχει μετατραπεί σε μια καθημερινή και αναπόσπαστη συσκευή εφόσον λόγω των πολλαπλών υπηρεσιών και δυνατοτήτων που παρέχει καλύπτει διάφορες καθημερινές ανάγκες του χρήστη.

Πιο κάτω επισημαίνονται μερικές δυνατότητες που παρέχει ένα SP :

Συνεχής πρόσβαση στο διαδίκτυο:

μέσω της ανάπτυξης της ασύρματης επικοινωνίας έχει επιτευχθεί η ασύρματη πρόσβαση στο διαδίκτυο μέσω διαφόρων πρωτοκόλλων όπως 3G,4G, HSPA+ . Έτσι ο χρήστης μπορεί πλέον να κάνει αναζήτηση στο παγκόσμιο ιστό, να κατεβάσει αρχεία , να μιλήσει μέσω skype, να συνδεθεί σε κάποιο κοινωνικό δίκτυο(facebook, twitter), να δει την ηλεκτρονική του αλληλογραφία κ.α. Δηλαδή μπορεί να πραγματοποιήσει οποιαδήποτε λειτουργία σχετίζεται με το διαδίκτυο όπως θα μπορούσε να το πράξει με την χρήση ενός φορητού υπολογιστή ή και μιας ηλεκτρονικής ταμπλέτας.

Πλοήγηση με την χρήση κάποιας εφαρμογής πλοήγησης:

Με την ενσωμάτωση GPS receiver μπορεί να βρει την ακριβή τοποθεσία στην οποία βρίσκεται ή ακόμη και να καθορίσει κάποιο προορισμό έτσι ώστε να ανακαλύψει το δρομολόγιο που πρέπει να ακολουθήσει για να φτάσει στον επιθυμητό προορισμό του χωρίς να σπαταλά άσκοπα χρόνο και κόστος.

Φωτογραφική μηχανή-Βίντεο:

Κάθε SP παρέχει την δυνατότητα λήψης φωτογραφίας και βίντεο. Σε περίπτωση που ο χρήστης θέλει να βγάλει φωτογραφία ή βίντεο το μόνο που έχει να κάνει είναι να ενεργοποιήσει την κατάλληλη λειτουργία και να αρχίσει την λήψη όπως θα έκανε με μια φωτογραφική μηχανή ή με μια κάμερα. Επίσης υπάρχουν μερικοί κατασκευαστές τηλεφώνων οι οποίοι έχουν δημιουργήσει SP τα οποία είναι ανθεκτικά στο νερό και έτσι ο χρήστης μπορεί να φωτογραφίζει κάτω από το νερό (για καθορισμένη διάρκεια και καθορισμένο βάθος).

Ραδιόφωνο:

Όλα τα SP μπορούν να μετατραπούν σε κινητά ραδιόφωνα . Με την χρήση ακουστικών τα οποία συνδέονται στην υποδοχή του SP ο χρήστης μπορεί να ακούσει τους τοπικούς ραδιοφωνικούς σταθμούς που επιθυμεί.

Ανταλλαγή δεδομένων :

Με την χρήση των πρωτοκόλλων όπως Bluetooth και NFC ένας χρήστης μπορεί να ανταλλάξει πληροφορίες με κάποιο άλλο χρήστη ο οποίος έχει SP που υποστηρίζει το ίδιο πρωτόκολλο.

Επίσης ανάλογα με τον κατασκευαστή ενός SP υπάρχουν και διάφοροι αισθητήρες όπως για παράδειγμα accelerometer, gyroscope, magnetometer, proximity sensor, light sensor , barometer, thermometer όπου καθένας από αυτούς προσφέρει ανάλογη πληροφορία που αντιστοιχεί σε μετρήσεις που έχει πραγματοποιήσει έτσι ώστε να παράξει το κατάλληλο αποτέλεσμα.[8]

5.2 Λειτουργικά συστήματα κινητών

Τα SP παρέχουν τις δυνατότητες που έχουν αναφερθεί και ακόμα περισσότερες λόγω της δημιουργίας κατάλληλων πλατφόρμων που λειτουργούν ως λειτουργικά συστήματα. Τα πιο γνωστά λειτουργικά συστήματα είναι τα εξής:

iOS: αυτό το OS έχει δημιουργηθεί από την εταιρεία Apple και το χρησιμοποιεί σε κάθε κινητή συσκευή που κατασκευάζει όπως iPhone, iPad και iPod touch.

Windows 10 Mobile: το OS δημιουργήθηκε από την εταιρεία Microsoft. Εκτός από την δημιουργία λειτουργικών συστημάτων και εφαρμογών η Microsoft έχει επεκταθεί (όπως και η εταιρεία Apple) στον τομέα των λειτουργικών συστημάτων για κινητά τηλέφωνα.

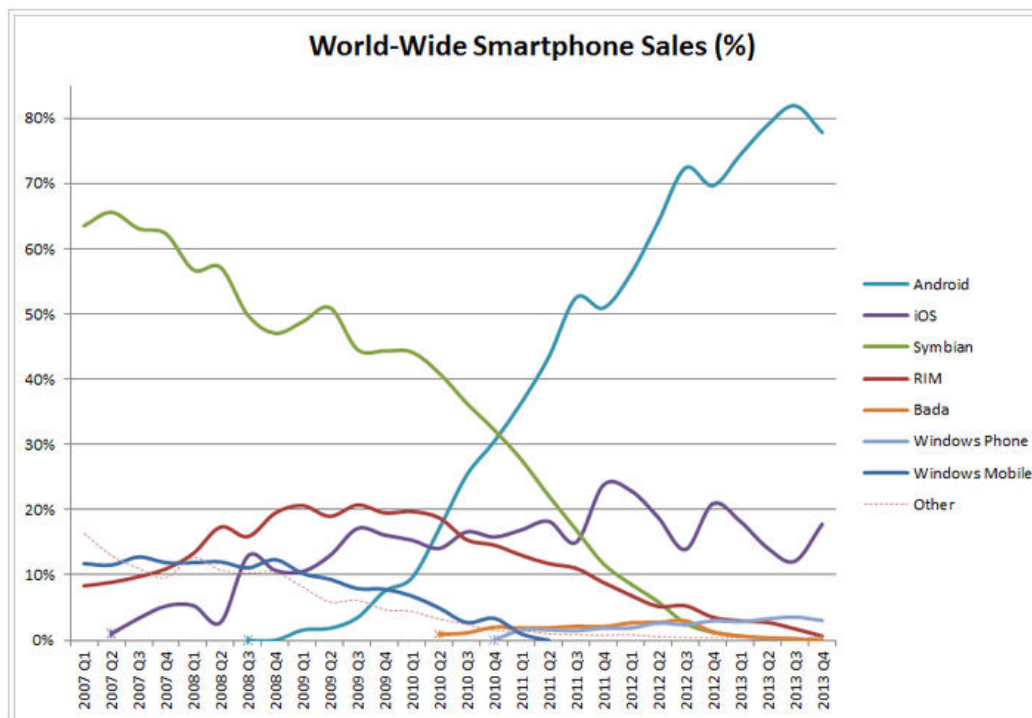
BlackBerry 10: η εταιρεία Blackberry έχει δημιουργήσει αυτό το OS το οποίο τρέχει μόνο σε προϊόντα που έχει κατασκευάσει η ίδια η εταιρεία

Android: δημιουργήθηκε από την εταιρεία Google. Σύμφωνα με διάφορες στατιστικές έρευνες είναι το OS που χρησιμοποιείται περισσότερο από τα άλλα OS που έχουν αναφερθεί πιο πάνω .

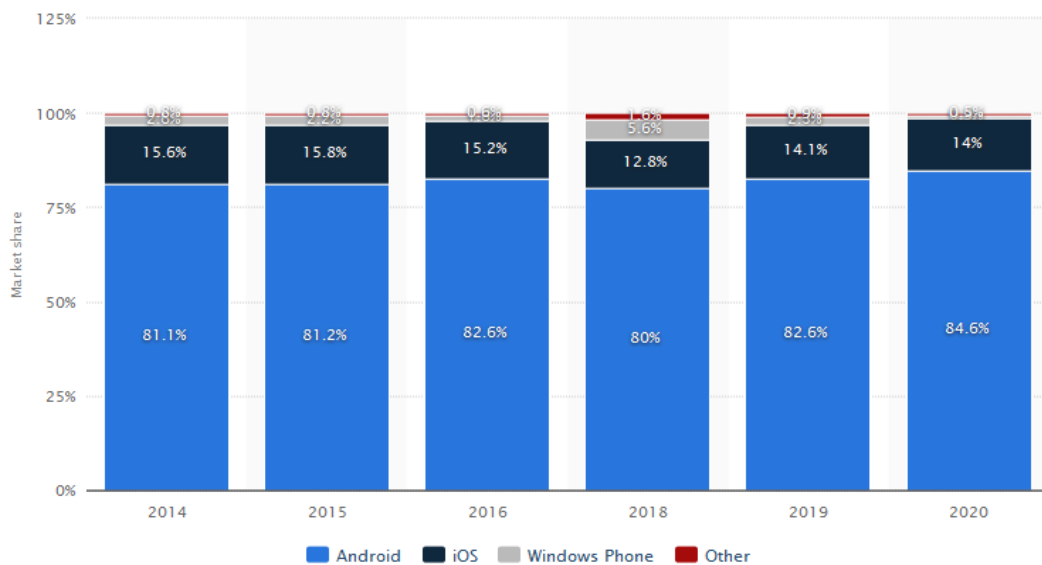
Στον πιο κάτω σύνδεσμο μπορούμε να δούμε την κατάσταση ,σχετικά με το λειτουργικό σύστημα έξυπνων κινητών τηλεφώνων, μιας χώρας.

<http://www.kantarworldpanel.com/global/smartphone-os-market-share/>

Στην επόμενη σελίδα παρουσιάζονται οι παγκόσμιες αγορές που έχουν γίνει ή πρόκειται να γίνουν για αγορά έξυπνων κινητών τηλεφώνων.



Σχήμα 5.1 Παγκόσμιες πωλήσεις «έξυπνων» κινητών τηλεφώνων [14]



Σχήμα 5.2 Ποσοστά αγορών «έξυπνων» κινητών τηλεφώνων [10]

5.3 Αρχιτεκτονική Android

Η πλατφόρμα Android έχει δημιουργηθεί από την εταιρεία Google και στηρίζεται στον πυρήνα του Linux. Αυτό το λειτουργικό σύστημα δεν περιορίζεται μόνο στην λειτουργία «έξυπνων» κινητών τηλεφώνων εφόσον ήδη χρησιμοποιείται και σε άλλες συσκευές που έχουν ενσωματωμένη οθόνη αφής όπως για παράδειγμα τηλεοράσεις , αυτοκίνητα , ρολόγια χεριού κ.α.[11].

Τα κύρια χαρακτηριστικά του λειτουργικού συστήματος είναι τα εξής:

- Διεπαφή:

Αποτελεί την βασική επικοινωνία μεταξύ του χρήστη και του λειτουργικού συστήματος. Η γραφική διεπαφή χρήστη επιτρέπει στον χρήστη να αλληλεπιδρά με διάφορα εικονίδια-αντικείμενα για να μπορεί να εκτελεί τις επιθυμητές ενέργειες έτσι ώστε να παραχθεί μια διαφορετική κατάσταση που να ανταποκρίνεται στις ανάγκες του χρήστη.

- Εφαρμογές:

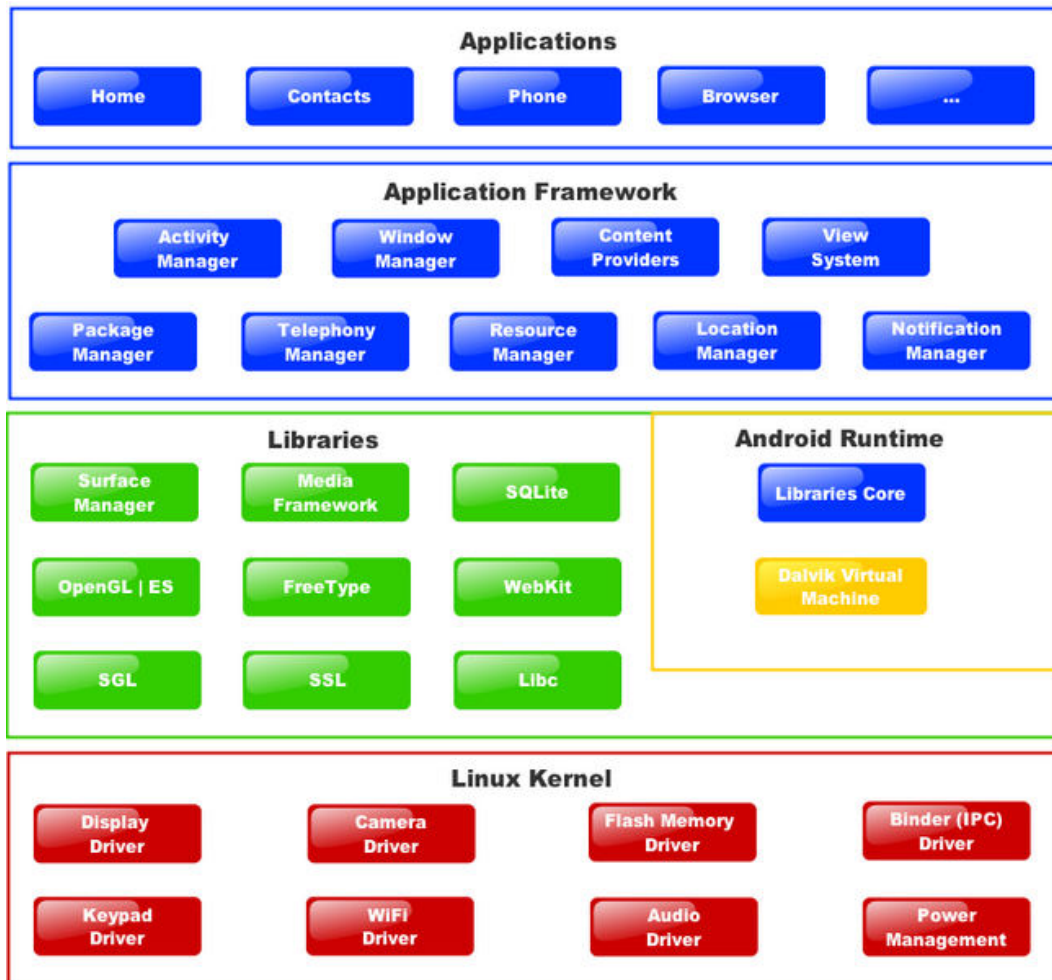
Οι εφαρμογές Android είναι προγράμματα που παρέχουν διάφορες δυνατότητες στο «έξυπνο» κινητό τηλέφωνο προσφέροντας με αυτό τον τρόπο επεκτασιμότητα στο λειτουργικό σύστημα. Οι εφαρμογές είναι κυρίως υλοποιημένες στην γλώσσα προγραμματισμού Java όμως μπορούν να υλοποιηθούν και σε συνδυασμό και με άλλες γλώσσες προγραμματισμού όπως C, C++ ,Go. Δύο τρόποι με τους οποίους μπορεί κάποιος να εγκαταστήσει εφαρμογές είναι είτε μέσω εγκατάστασης του εκτελέσιμου αρχείου της εφαρμογής είτε μέσω κάποιου καταστήματος εφαρμογών το οποίο παρέχει Android εφαρμογές όπως Google Play Store, F-Droid κ.α.

- Διαχείριση μνήμης:

Λόγω της ιδιαιτερότητας των κινητών συσκευών Android (λειτουργούν μέσω μπαταρίας) χρειάζεται να γίνεται κατάλληλη διαχείριση των διεργασιών έτσι ώστε επιτυγχάνεται περισσότερη διάρκεια χρήσης. Έτσι το λειτουργικό

σύστημα Android ελέγχει τις ενεργές διεργασίες έτσι ώστε να καταναλώνεται όσο το δυνατό λιγότερη ενέργεια χωρίς να επηρεάζεται η απόδοση του συστήματός. Αυτό επιτυγχάνεται με την συνεχή παρακολούθηση της ελεύθερης μνήμης. Σε περίπτωση που οι διεργασίες καταναλώνουν αρκετή μνήμη τότε το λειτουργικό σύστημα τερματίζει τις διεργασίες που ήταν ανενεργές για το μεγαλύτερο χρονικό διάστημα.

Το λειτουργικό σύστημα Android αποτελείται από 5 επίπεδα (τα οποία παρέχουν διαφορετικές υπηρεσίες)που επικοινωνούν μεταξύ τους έτσι ώστε να παρέχουν υποστήριξη τόσο του συστήματος όσο και των εφαρμογών που έχουν εγκατασταθεί στο σύστημα[6]. Τα 5 συστατικά στοιχεία του λειτουργικού συστήματος Android όπως φαίνονται και στο Σχήμα 5.4 είναι : Linux Kernel, Libraries, Android Run time ,Application Framework και Android app.



Σχήμα 5.4 Αρχιτεκτονική Android

Πιο κάτω ακολουθεί περιγραφή των 5 επιπέδων του λειτουργικού συστήματος ξεκινώντας από το ψηλότερο επίπεδο.

1. Applications:

Οι εφαρμογές αποτελούν το ψηλότερο επίπεδο της αρχιτεκτονικής Android. Σε αυτό το επίπεδο ανήκουν όλα τα προγράμματα που έχουν εγκατασταθεί στο λειτουργικό σύστημα είτε αυτά είναι προεπιλεγμένα (built in) είτε τα έχει εγκαταστήσει ο χρήστης .

2. Application Framework:

Σε αυτό το επίπεδο υπάρχουν προγράμματα τα οποία είναι υπεύθυνα να διαχειρίζονται τις εντολές που λαμβάνουν από τις εφαρμογές π.χ. διαχείριση πόρων , διαχείριση κλήσης, διαχείριση διάφορων ειδοποιήσεων κτλ .

3. Android Run Time:

Αυτό το επίπεδο αποτελείται από τις βιβλιοθήκες και το Dalvik Virtual Machine. Τα πλεονεκτήματα του Dalvik Virtual Machine καθιστούν το λειτουργικό σύστημα αρκετά αποδοτικό εφόσον έχει σχεδιαστεί έτσι ώστε να παρέχει χαμηλά επίπεδα τόσο σε μνήμη όσο και σε ενέργεια. Ακόμη λόγω της ιδιότητας του να υποστηρίζει ταυτόχρονη χρήση πολλαπλών στιγμιότυπων εικονικής μηχανής (virtual machine) παρέχει ασφάλεια, διαχείριση μνήμης και νημάτωση(threading).

4. Libraries:

Είναι βιβλιοθήκες οι οποίες είναι γραμμένες στην γλώσσα προγραμματισμού c /c++ και διαφέρουν από συσκευή σε συσκευή. Οι βιβλιοθήκες είναι υπεύθυνες για την διαχείριση διαφορετικών τύπων δεδομένων που πρέπει να χρησιμοποιήσει μια συσκευή. Για παράδειγμα η βιβλιοθήκη Open GL χρησιμοποιείται για την απεικόνιση δυσδιάστατων ή τρισδιάστατων γραφικών.

5. Linux kernel:

Αποτελεί το επίπεδο στο οποίο στηρίζεται ολόκληρο το λειτουργικό σύστημα Android. Αυτό το επίπεδο είναι υπεύθυνο για την επικοινωνία του υλικού με το λογισμικό. Δηλαδή περιέχει όλους τους οδηγούς του υλικού και είναι υπεύθυνο να μεταφράζει τις εντολές των προηγούμενων επιπέδων σε ενέργειες που πρέπει να εκτελέσει το αντίστοιχο υλικό π.χ. αν ο χρήστης ενεργοποιήσει την εφαρμογή της φωτογραφικής μηχανής τότε στον kernel πρέπει να εκτελεστούν οι οδηγίες που αφορούν το υλικό της φωτογραφική του «έξυπνου» κινητού τηλεφώνου , το φλας που πιθανό να είναι ενεργοποιημένο κτλ.

Κεφάλαιο 6

Σχεδίαση Εφαρμογής

6.1 Πρότυπα	34
6.2 Εντολές – Σενάρια	38
6.3 Πρωτόκολλα επικοινωνίας –Αρχιτεκτονικές σχεδίασης	46
6.4 Συνδετικά συστατικά υποσυστήματα	58
6.5 Απαιτήσεις εφαρμογής	60
6.6 Λειτουργία εφαρμογής	61
6.7 Σύγκριση εφαρμογής με παρόμοιες εφαρμογές	68

6.1 Πρότυπα

Σχεδιαστικά Πρότυπα, σύμφωνα με τον Christopher Alexander [3], περιγράφουν ένα πρόβλημα το οποίο επαναλαμβάνεται συνεχώς καθώς και μια λύση στο πρόβλημα που το επιλύει με αποδοτικό τρόπο ούτως ώστε να μπορεί να επαναχρησιμοποιηθεί πολλές φορές χωρίς να υπάρχει ανάγκη να ξαναδημιουργήσουμε την ίδια λύση. Δηλαδή αν μπορούμε να κατανοήσουμε σωστά τις διάφορες ανάγκες και απαιτήσεις του προβλήματος τότε αν γνωρίζουμε και το κατάλληλο σχεδιαστικό πρότυπο μπορούμε να το χρησιμοποιήσουμε ή να το προσαρμόσουμε ανάλογα χωρίς να χρειαστεί να δημιουργήσουμε από την αρχή όλη την διαδικασία.

Χαρακτηριστικά σχεδιαστικού προτύπου

Μπορούμε να παρομοιάσουμε το σχεδιαστικό πρότυπο σαν ένα ιστορικό γεγονός, το οποίο έχει γίνει στο παρελθόν κατά το οποίο γνωρίζουμε το πρόβλημα που υπήρχε την λύση που είχε δοθεί για την αντιμετώπιση του, καθώς επίσης και τις συνέπειες που προέκυψαν[2].

Πιο συγκεκριμένα το σχεδιαστικό πρότυπο αποτελείται από τα 4 πιο κάτω χαρακτηριστικά:

- Όνομα του σχεδιαστικού προτύπου.

Αποτελεί λέξη κλειδί η οποία περιγράφει περιληπτικά το πρόβλημα, την λύση και τις συνέπειες ενός σχεδιαστικού προτύπου. Με αυτό επιτυγχάνεται ο προσδιορισμός μιας πιο αφαιρετικής έννοιας με αποτέλεσμα οι σχεδιαστές και οι προγραμματιστές να μπορούν με περισσότερη ευκολία να αποφασίσουν για την χρήση και υλοποίηση του πιο κατάλληλου σχεδιαστικού μοντέλου. Μια αντιστοιχία που μπορούμε να σκεφτούμε είναι βασικές έννοιες που χρησιμοποιούνται για να καταλάβουμε τι ακριβώς προσπαθούμε να επιτύχουμε π.χ. αν αναφερθούμε σε ένα πρωτόκολλο όπως TCP τότε γνωρίζουμε κάποιες ιδιότητες που παρουσιάζει, τα προβλήματα – συμβιβασμούς που υπάρχουν κ.α. Καταλαβαίνουμε δηλαδή μια γενική εικόνα για το είδος του προβλήματος που έχουμε να αντιμετωπίσουμε και τον τρόπο επίλυσης του.

Στην περίπτωση του προβλήματος που θέλουμε να λύσουμε θα χρησιμοποιήσουμε το πρότυπο Command. Το πρότυπο αυτό χρησιμοποιείται όταν έχουμε πολλές συσκευές στις οποίες χρειάζεται να δώσουμε κάποιες εντολές και ανάλογα να εκτελεστεί μια λειτουργία που να εξυπηρετεί την εντολή και να παράγει το επιθυμητό αποτέλεσμα. Ένα τέτοιο παράδειγμα είναι και το τηλεχειριστήριο που χρησιμοποιούμε στις τηλεοράσεις ή και στα ηχοσυστήματα με την διαφορά πως με το πάτημα ενός κουμπιού ελέγχουμε και αλλάζουμε την κατάσταση της αντίστοιχης συσκευής ενώ στην περίπτωση περισσότερων συνδεδεμένων συσκευών επιτυγχάνεται αντίστοιχη αλλαγή κατάσταση της συσκευής της οποίας επιθυμούμε.

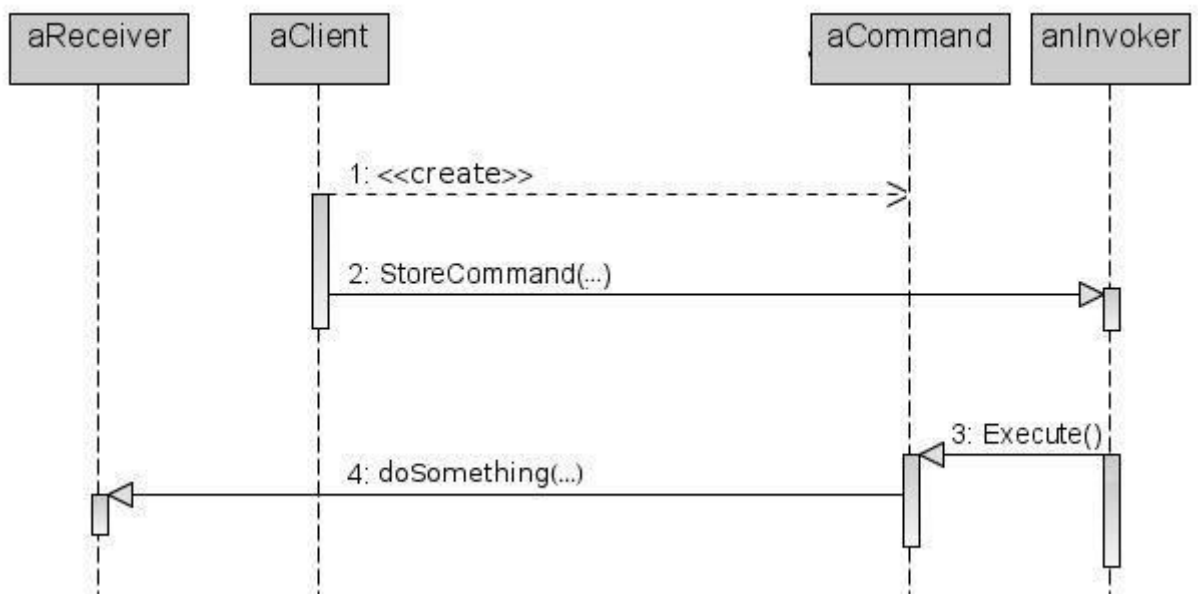
- Πρόβλημα

Αποτελεί μια περιγραφή στην οποία παρουσιάζεται το πρόβλημα που πρέπει να επιλυθεί. Επίσης μπορεί να περιέχει και διάφορες καταστάσεις οι οποίες αποτελούν παραδείγματα τα οποία χρησιμοποιούν το σχεδιαστικό πρότυπο έτσι ώστε οι προγραμματιστές- σχεδιαστές να μπορούν να κατανοήσουν καλύτερα τον τρόπο με τον οποίο μπορούν να χρησιμοποιήσουν το σχεδιαστικό πρότυπο.

Το πρόβλημα που αντιμετωπίζουμε είναι η επικοινωνία μεταξύ του χρήστη και των Smart things . Θέλουμε να δίνουμε φωνητικές εντολές οι οποίες να μεταφράζονται σε ενέργειες που πρέπει να εκτελέσουν τα Smart Things έτσι ώστε να ολοκληρώσουν μια εργασία που αποτελεί και την απαίτηση – επιθυμητό αποτέλεσμα του χρήστη. Δηλαδή χρειάζεται να βρούμε ένα τρόπο έτσι ώστε να γίνει μια επεξεργασία της φωνής έτσι ώστε να ενεργοποιηθεί η κατάλληλη εντολή και να εκτελεστεί με την βοήθεια του μηχανισμού που παρέχει την δυνατότητα διεκπεραίωσης της. Οι μηχανισμοί σε αυτή την περίπτωση αναπαριστούν τις 19 «έξυπνες συσκευές» που έχουν αναφερθεί στο κεφάλαιο Smart Home.

- Λύση

Αυτό το χαρακτηριστικό παρουσιάζει μια γενική περιγραφή για την σχεδίαση λύσης του προβλήματος με την χρήση των συστατικών στοιχείων και τις σχέσεις μεταξύ τους. Πιο κάτω βλέπουμε τον τρόπο λειτουργίας του πρότυπου Command.[16]



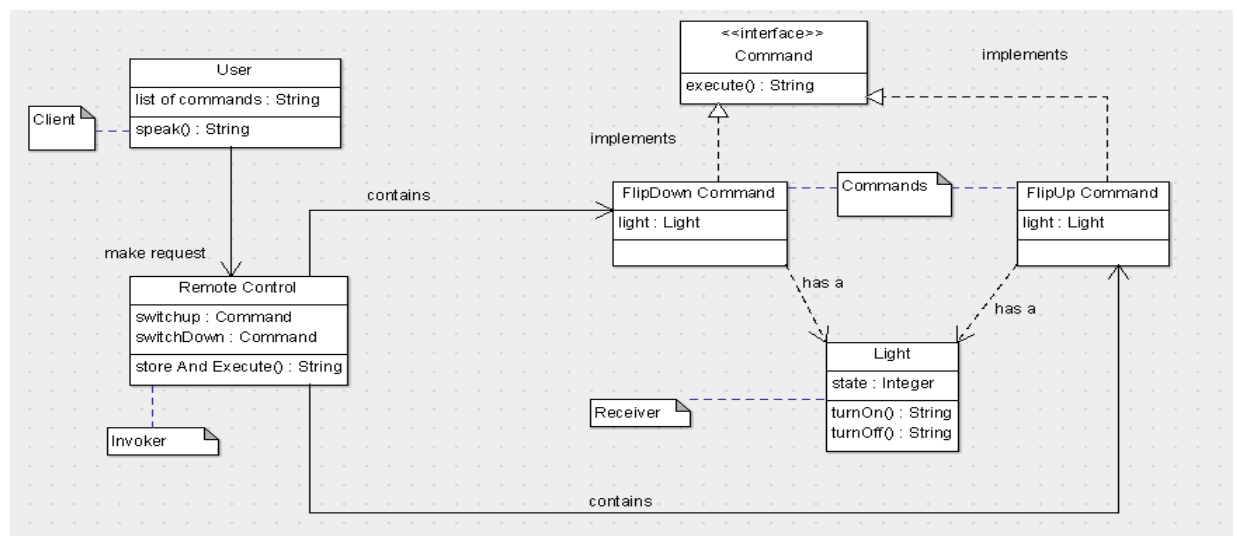
Σχήμα 6.1 Ακολουθιακό Διάγραμμα πρότυπου Command

Επεξήγηση προτύπου:

Στο πιο πάνω διάγραμμα αναπαριστώνται με βέλη οι ενέργειες που πραγματοποιούνται

μεταξύ των αντικειμένων έτσι ώστε πραγματοποιηθεί μια εντολή. Ο χρήστης δημιουργεί μια εντολή και έπειτα στέλλει την εντολή στον διαμεσολαβητή οποίος είναι υπεύθυνος για την διεκπεραίωση αυτής της εντολής με την βοήθεια του κατάλληλου παραλήπτη. Ο χρήστης γνωρίζει τις διαθέσιμες εντολές που υπάρχουν στο σύστημα(δηλαδή τις δυνατότητες που παρέχει το σύστημα χωρίς οποιαδήποτε γνώση για το πως εκτελούνται οι εντολές) και ανάλογα με την τρέχουσα εντολή που δίνει κάθε φορά απευθύνεται στον διαμεσολαβητή ο οποίος την αναγνωρίζει και την ενεργοποιεί έτσι ώστε να εκτελεστούν οι κατάλληλες ενέργειες στον αντίστοιχο παραλήπτη.

Πιο κάτω ακολουθεί το διάγραμμα κλάσεων που αναπαριστά την λύση στο πρόβλημα που έχουμε να λύσουμε .Πιο συγκεκριμένα ο χρήστης (Client) γνωρίζει τις φωνητικές εντολές που μπορεί να χρησιμοποιήσει και εφόσον πραγματοποιήσει φωνητική αλληλεπίδραση που σχετίζεται με κάποια εντολή τότε η κλάση Remote Control(Invoker) προσδιορίζει την εντολή που πρέπει να πραγματοποιηθεί. Κάθε εντολή(Command) υλοποιεί μια συγκεκριμένη λειτουργία που συσχετίζεται με ένα αντίστοιχο παραλήπτη(Receiver).Εφόσον ολοκληρωθεί η εντολή τότε επιστρέφεται ανάλογο μήνυμα το οποίο υποδηλώνει μια κατάσταση(η εντολή έχει πραγματοποιηθεί με επιτυχία/δεν πραγματοποιήθηκε με επιτυχία ή διάφορες μετρήσεις) καθώς επίσης και εικόνα – ήχος (ανάλογα με την εντολή).



Συσχέτιση ρόλων μεταξύ του διαγράμματος κλάσεων και του ακολουθιακού διαγράμματος

Σχήμα 6.2 Διάγραμμα κλάσεων για το Command Pattern

- Συνέπειες

Χαρακτηριστικό ενός πρότυπου στο οποίο παρουσιάζονται τα πλεονεκτήματα – μειονεκτήματα σε περίπτωση που υλοποιείται ένα πρότυπο.

Οι συνέπειες του Command pattern είναι οι εξής:

1. Μπορούμε να προσθέσουμε περισσότερες εντολές χωρίς να επηρεάζονται οι υπάρχουσες κλάσεις. Με αυτό τον τρόπο μπορούμε να έχουμε περισσότερες κλάσεις οι οποίες να εκτελούν διαφορετικές εντολές διατηρώντας μια απλότητα στην σχεδίαση και καλύτερο έλεγχο .
2. Μπορούμε να δημιουργήσουμε πιο σύνθετες εντολές. Αυτό έχει σαν αποτέλεσμα να χρησιμοποιούμε υπάρχουσες εντολές χωρίς να χρειάζεται να ξαναδημιουργηθούν (διευκολύνουν την περίπτωση σεναρίων).
3. Μπορούμε να επεκτείνουμε πιο εύκολα τις υπάρχουσες κλάσεις που υλοποιούν την διεπαφή Command
4. Με την χρήση του Command pattern επιτυγχάνεται έλεγχος μέσω του μεσολαβητή (invoker) και διάκριση του από τον παραλήπτη(receiver).
5. Λόγω της απλότητας των εντολών επιτυγχάνεται καλύτερη αποσφαλμάτωση σε περίπτωση που παρουσιαστούν προβλήματα.

6.2 Εντολές – Σενάριο

Οι εντολές αποτελούν τον συνδετικό κρίκο μεταξύ του χρήστη, που χρησιμοποιεί την εφαρμογή, και του συστήματος τους έξυπνου σπιτιού. Ο χρήστης λαμβάνοντας υπόψη τις διάφορες εντολές που παρέχονται από την εφαρμογή μπορεί να αλληλεπιδράσει με την «έξυπνη» συσκευή που αντιστοιχεί στην εντολή την οποία έχει προσδιορίσει προηγουμένως. Με βάση τις 19 «έξυπνες» συσκευές οι οποίες έχουν αναφερθεί προηγουμένως έχουν δημιουργηθεί κατηγορίες εντολών με κριτήριο τις ιδιότητες-χαρακτηριστικά των «έξυπνων» συσκευών. Πιο κάτω φαίνεται ο πίνακας με τις βασικές εντολές της εφαρμογής για τις υποστηριζόμενες «έξυπνες» συσκευές.

Εντολή	Περιγραφή	«Έξυπνες» συσκευές - συστήματα
Turn on/off	Αλλαγή κατάστασης της «έξυπνης» συσκευής	• Lighting system • pop corn maker • toast maker • coffee corn maker • home cinema • glass break sensor • Sensor(humidity, light etc.)
Activate/Deactivate	Ενεργοποίηση/Απενεργοποίηση ενός συστήματος	• Security camera system • Security vehicle system • Central heating system • Personal alarm • Washing machine • Boiler system • Solar system • Kitchen oven
Notify	Ειδοποίηση για κάποιο γεγονός	• Family notification system • Baby watch system
Check	Παρακολούθηση μιας κατάστασης (ένδειξη μετρήσεων)	• Temperature • Water temperature • Lightness • humidity
Open /Close	Αλλαγή κατάστασης της «έξυπνης» συσκευής	• Doors • Windows • Garage gate

Πίνακας 6.3 Βασικές εντολές της εφαρμογής

Με τον καθορισμό των εντολών μπορούμε να πραγματοποιήσουμε σύνθετες εντολές οι οποίες αποτελούν ένα σενάριο. Η λέξη σενάριο προσδιορίζει παρόμοια σημασία όπως την γνωρίζουμε με την διαφορά ότι υπάρχει μια σειρά με την οποία εκτελούνται συγκεκριμένες ενέργειες με αποτέλεσμα να δημιουργείται μια επιθυμητή κατάσταση που ικανοποιεί ορισμένες ανάγκες του χρήστη.

Ανάλογα με τις καθημερινές δραστηριότητες του κάθε χρήστη, έχει δημιουργηθεί και ένα συγκεκριμένο σενάριο με σκοπό να αυτοματοποιήσει διάφορες ενέργειες και να παρέχει εξοικονόμηση χρόνου στον χρήστη. Παρακάτω ακολουθούν σενάρια που καθορίζουν την «συμπεριφορά» των «έξυπνων» συσκευών για μια οικογένεια. Πιο συγκεκριμένα αφορούν μια 3μελή οικογένεια, όπου παρουσιάζεται η καθημερινότητα του πατέρα, της μητέρας και ενός παιδιού, κατά την διάρκεια μιας καθημερινής μέρας.

Σενάριο	Περιγραφή
1	Σενάριο αφύπνισης για τον πατέρα
2	Σενάριο αφύπνισης για την μητέρα
3	Σενάριο αφύπνισης για το παιδί
4	Σενάριο για επιστροφή της μητέρας από δουλειά
5	Σενάριο για επιστροφή του πατέρα από δουλειά
6	Σενάριο για επιστροφή του παιδιού από δουλειά
7	Σενάριο διασκέδασης(για παρακολούθηση κινηματογραφικής ταινίας)
8	Σενάριο εξοικονόμησης ενέργειας
9	Σενάριο ασφάλειας του έξυπνου σπιτιού
10	Σενάριο παραπλάνησης
11	Σενάριο αντιμετώπισης σε περίπτωση κλοπής

Πίνακας 6.4 Σενάρια της εφαρμογής

1. Wake up scenario

Voice Command: Smart Home wake me up

Actions:

- Check temperature of water at 6:30
- Activate
 - a. boiler if water temperature below 30 °C
 - b. alarm at 7 o' clock .
- Turn on
 - a. bedroom's lights at 7:00.
 - b. bathroom's lights at 7:00.
 - c. coffee maker at 7:05
 - d. toast maker at 7:10
 - e. kitchen lights on 7:10

- Deactivate
 - Security for back door at 7:25
- Open garage gate at 7:28
- Close garage gate at 7:31
- Activate
 - a. security guard at 7:48
 - b. family notifier

2. Wake up scenario(Mother)

Voice Command: Smart Home wake me up

Actions:

- Check temperature of water at 6:30
- Activate
 - boiler if water temperature below 30 °C
- Activate
 - a. washing machine remote controller *
 - b. kitchen oven remote controller *
- Open garage gate at 7:45
- Close garage gate at 7:48

3. Wake up scenario(Child)

Voice Command: Smart Home wake me up

Actions:

- Check temperature of water at 6:50
- Activate
 - a. boiler if water temperature below 30 °C
 - b. alarm at 7 o' clock .

- Turn on
 - a. bedroom's lights at 7:00.
 - b. bathroom's lights at 7:00.
- Deactivate
 - a. Security of bicycle at 7:45
- Open garage gate at 7:45
- Close garage gate at 7:48

4. Ordinary day scenario(Mother)

Voice Command: Smart Home I' m coming home

Actions:

- Open garage gate at 13:30
- Close garage gate at 13:33
- Open bedrooms' windows at 13:45
- Deactivate security for the back door at 13:35
- Open back door at 13: 35
- Open garage gate at 15:20 (go for shopping)
- Close garage gate at 15:23
- Open garage gate at 16:30
- Close garage gate at 16:33
- Check
 - temperature of water at 17:00
- Activate
 - boiler if water temperature below 32 °C

5. Arrival scenario(Father)

Voice Command: Smart Home I' m coming home

Actions:

- Open garage gate in 16:20
- Close garage gate at 16:23
- Deactivate security for front door at 16:23

- Check
 - temperature of water at 16:00
 - temperature of the house at 16:00

- Activate
 - boiler if water temperature below 35 °C
 - central heater if temperature below 30 °C
 - solar energy system
- Turn on living room lights at 16:23
- Open living room's door if user open car's rear door
- Turn on TV at 16:45

6. Arrival scenario(Child)

Voice Command: Smart Home I' m coming home

Actions:

- Activate bicycle security at 14:05
- Turn on TV at 14:30
- Notify child at 15:05 for homework
- Turn off TV at 15:10
- Open curtains of child's study room at 15:10
- Close window if temperature below 26 °C
- Notify child for dog's walk at 17:00
- Notify child for coming home at 18:30

- Close curtains of child's study room at 17:00
- Check
 - temperature of water at 18:20
- Activate
 - boiler if water temperature below 35 °C
 - solar energy system
- Turn on motion sensor at 18:30 (for lighting when child comes home)
- Deactivate
 - Security of back door at 18:35
- Open back door at 18:35

7. Entertainment scenario

Voice Command: Smart Home enable movie hour at ____ (movie hour)

Actions:

- Check light state
- Turn off lights if it's night / Close curtains if it's day
- Activate home cinema system
- Turn on pop corn maker

8. Save energy scenario

Voice Command: Smart Home activate save energy mode

Actions:

- Check room leaks at 12:00 every day
- Turn on motion sensor when someone is in the house (lights control, device state)
- Activate
 - climate watcher when someone is in the house(house lighting)
 - boiler saver (water use)

9. Security guard scenario

Voice Command: Security guard

Actions:

- Turn on
 - Glass break sensor
 - Outdoor motion sensors
- Activate
 - Doors' security system
 - Outdoor cameras' system

10. Feint scenario

Voice Command: Smart Home create illusion scenes (*Scenes take place different hours every day*)

Actions:

Activate security guard

- Turn on
 - lights on rooms at
 - TV at....
 - radio at...
 - player at (play recorded discussions)
- Turn off
 - lights on rooms at
 - TV at....
 - radio at...
 - player at....

11. Theft prevention/ authentication scenario

Voice command: Block out

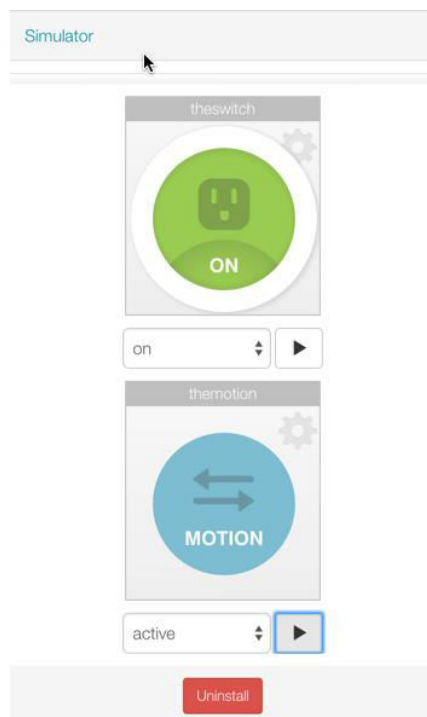
Actions:

- Activate
 - a. Security guard.
 - b. Security vehicle guard
 - c. Voice authenticator system
 - d. SAG user gps position

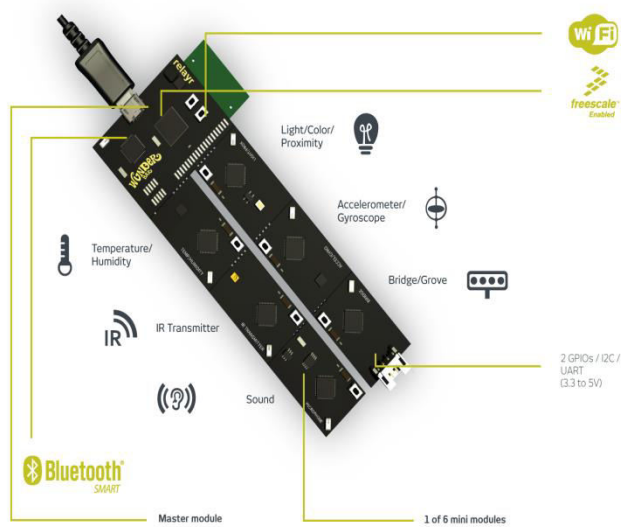
6.3 Πρωτόκολλα επικοινωνίας –Αρχιτεκτονικές σχεδίασης

Τα πρωτόκολλα αποτελούν την βάση στην οποία στηρίζεται η επικοινωνία συστημάτων. Παρέχουν την δομή και την σειρά των μηνυμάτων που πραγματοποιεί οποιαδήποτε συσκευή του δικτύου ,καθώς επίσης και τις ενέργειες που πρέπει να πραγματοποιηθούν κατά την αποστολή – παραλαβή τους.

Στην εφαρμογή SAG χρησιμοποιούνται 2 τρόποι με τους οποίους επιτυγχάνεται σύνδεση με πλατφόρμες που παρέχουν «έξυπνους» αισθητήρες και «έξυπνες» συσκευές. Η πρώτη πλατφόρμα ονομάζεται Smart Things και στηρίζεται σε ένα εξυπηρετητή ο οποίος παρέχει προσομοιωτή έτσι ώστε να μπορούμε να ελέγξουμε την συμπεριφορά των «έξυπνων» συσκευών όταν πρέπει να εκτελέσουν μια ενέργεια πριν πραγματοποιηθεί σε φυσική συσκευή. Η 2^η πλατφόρμα είναι το Wunderbar η οποία αποτελείται από 6 αισθητήρες και ένα εξυπηρετητή (relayr cloud). Μπορεί επίσης να επεκταθεί με σύνδεση περισσότερων αισθητήρων με την χρήση ενός μικροελεγκτή (Arduino).



Σχήμα 6.5 Smart Things Simulator



Σχήμα 6.6 Wunderbar

Πλατφόρμα Smart Things

Αρχιτεκτονική πλατφόρμας Smart Things

Η πλατφόρμα Smart Things υποστηρίζει τα πρωτόκολλα Zigbee, Z-Wave καθώς επίσης και IP/Wifi έτσι ώστε να παρέχει την δυνατότητα να συνδέονται όλο και περισσότερες «έξυπνες» συσκευές. Ακολουθούν σύντομες περιγραφές των 3 αυτών πρωτοκόλλων.

Πρωτόκολλο zigbee:

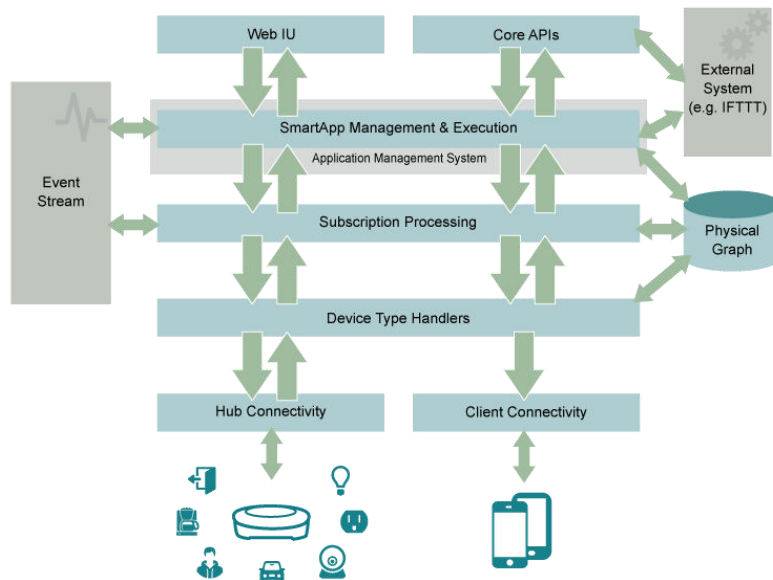
Αυτό το πρωτόκολλο ανήκει στην κατηγορία των ασύρματων δικτύων προσωπικού χώρου (WPAN) και στηρίζεται στο πρότυπο IEEE802.15.4. Επίσης λειτουργεί σε συχνότητες που δεν χρειάζονται άδεια για χρήση και χρησιμοποιείται συνήθως για παρακολούθηση, συλλογή και ανάλυση δεδομένων. Ένα αρκετό σημαντικό πλεονέκτημα που έχει είναι η χαμηλή κατανάλωση ενέργειας με αποτέλεσμα οι συσκευές που χρησιμοποιούν αυτό το πρωτόκολλο να έχουν μεγάλη αντοχή στον χρόνο(σε περίπτωση που χρησιμοποιούν μπαταρίες).

Πρωτόκολλο z-wave:

Αυτό το πρωτόκολλο έχει σχεδιαστεί για να παρέχει ψηλά data rates και low latency transmission. Χρησιμοποιεί διαφορετικές συχνότητες μετάδοσης ανάλογα με την περιοχή και υλοποιείται με την διαμόρφωση Manchester. Ακόμη χρησιμοποιεί την τοπολογία source routed mesh network έτσι ώστε να μπορεί να μεταφέρει μηνύματα-πληροφορίες από μια «έξυπνη» συσκευή σε μία άλλη.

Πρωτόκολλο wifi

Το πρότυπο που χρησιμοποιεί το πρωτόκολλο wifi είναι το 802.11. Αυτό το πρωτόκολλο χρησιμοποιείται για να παρέχει ασύρματη επικοινωνία μεταξύ συσκευών ενός τοπικού ασύρματου δικτύου με την βοήθεια του IP. Επίσης υποδιαιρείται σε περισσότερες κατηγορίες οι οποίες διαφέρουν ως προς την συχνότητα μετάδοσης καθώς επίσης και το data rate.



Σχήμα 6.7 Αρχιτεκτονική πλατφόρμας Smart Things

Στο πιο πάνω σχήμα φαίνεται η αρχιτεκτονική της πλατφόρμας Smart Things η οποία αποτελείται από τα εξής συστατικά στοιχεία[20]:

Devices:

Αποτελούν τις «έξυπνες» συσκευές που υπάρχουν στο δίκτυο οι οποίες θα εκτελέσουν τις λειτουργίες για τις οποίες έχουν σχεδιαστεί όταν παραλάβουν μήνυμα από το smart things hub.

Smart things hub:

Χρησιμοποιείται στην σύνδεση των διαφόρων «έξυπνων» συσκευών με το διαδίκτυο .Το hub συνδέεται με τον broadband router έτσι ώστε να καταστεί εφικτή η σύνδεση τόσο με το cloud όσο και με την εφαρμογή του SP.

Device type handlers:

Αυτό το συστατικό στοιχείο καθορίζει την εικονική αναπαράσταση μιας «έξυπνης» συσκευής. Ανάλογα με την συσκευή καθορίζεται κάποιος τύπος λειτουργίας ο οποίος περιέχει διαφορετικές ικανότητες με αποτέλεσμα να πραγματοποιούνται οι αντίστοιχες εντολές.

Στην εφαρμογή χρησιμοποιούνται handlers για συσκευές που υποστηρίζουν τις ικανότητες :

Switch : για να πραγματοποιούνται οι εντολές on/off

Alarm: για την εκτέλεση των εντολών strobe/off

Lock: για την εκτέλεση των εντολών lock/unlock

Subscription processing:

Η πλατφόρμα smart things στηρίζεται στην ύπαρξη γεγονότων. Τα γεγονότα δημιουργούνται είτε από εντολές που προέρχονται από μια εφαρμογή είτε από κάποια αλλαγή κατάστασης κάποιας συνδεδεμένης «έξυπνης» συσκευής. Το subscription processing παρέχει την δυνατότητα συσχέτισης ενός γεγονότος με τον κατάλληλο device handler .

Smart Application management & execution:

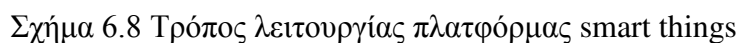
Αυτό το στοιχείο αποτελεί την εφαρμογή που χρησιμοποιεί την πλατφόρμα smart things έτσι ώστε να αλληλεπιδρά με τις «έξυπνες» συσκευές που είναι συνδεδεμένες. Ακόμη επικοινωνεί με το προηγούμενο στοιχείο, σε περίπτωση που υπάρχει ένα νέο γεγονός που προέρχεται από μια «έξυπνη» συσκευή και δημιουργεί ένα γεγονός το οποίο είτε προέρχεται από κάποια εντολή του χρήστη είτε από κάποια συγκεκριμένη ρουτίνα.

Web UI

Το Web UI αποτελεί την διαδικτυακή εφαρμογή μέσω της οποίας μπορούμε να προσδιορίσουμε τις ενέργειες τις οποίες μπορούν να εκτελέσουν οι «έξυπνες» συσκευές. Δηλαδή μπορούμε να καθορίσουμε ποιες «έξυπνες» συσκευές υπάρχουν στο «έξυπνο» σπίτι , ποιες εντολές μπορούν να εκτελούν κτλ. Σε αυτό το συστατικό στοιχείο συμπεριλαμβάνεται και ο προσομοιωτής που παρέχει την δυνατότητα να καθορίσουμε την συμπεριφορά των «έξυπνων» συσκευών προτού να εκτελέσουμε τις εντολές σε πραγματικές «έξυπνες» συσκευές.

Client connectivity:

Αυτό το στοιχείο παρέχει την δυνατότητα στις διάφορες συσκευές που χρησιμοποιεί ο χρήστης (SP, tablet κ.α.) να μπορούν να συνδεθούν στο cloud της πλατφόρμας.



Βήμα 1:

Βήματα 2-4:

Βήμα 5: Εφόσον ο χρήστης έχει συνδεθεί επιτυχώς με το σύστημα μπορεί να αποκτήσει πρόσβαση σε «έξυπνες» συσκευές μέσω των κωδικών που του έχουν καθοριστεί από το Σύστημα(authorization code, OAuth ID και client ID)με την χρήση του “API Access ” SmartApps.

Βήμα 6:

Έπειτα εφόσον πιστοποιηθούν οι κωδικοί πρόσβασης μπορεί να δημιουργηθεί ένα Access Token το οποίο θα πρέπει να χρησιμοποιηθεί για να έχουμε πρόσβαση στις «έξυπνες» συσκευές.

Βήματα 7-8:

Με την ολοκλήρωση του βήματος 6 ο χρήστης μπορεί να γνωρίζει τις διευθύνσεις πρόσβασης των «έξυπνων» συσκευών και έτσι να μπορεί να διενεργεί «κλήσεις» σε αυτές τις διευθύνσεις μέσω καθορισμένων API.



Σχήμα 6.9 Επικοινωνία – Αρχιτεκτονική πλατφόρμας Smart Things

Στην πιο πάνω εικόνα βλέπουμε την επικοινωνία των διαφόρων συστατικών στοιχείων της πλατφόρμας smart things. Πιο συγκεκριμένα παρατηρούμε το «έξυπνο» σπίτι στο οποίο βρίσκονται οι «έξυπνες» συσκευές οι οποίες ενώνονται μεταξύ τους μέσω κάποιας εξειδικευμένης πλατφόρμας που έχει δημιουργήσει η εταιρεία Samsung(όπως έχουμε πει προηγουμένως οι «έξυπνες» συσκευές συνδέονται στο smart hub). Έπειτα εφόσον ο χρήστης έχει εγγραφεί στην πλατφόρμα μέσω κάποιου εξυπηρετητή ,ο οποίος λειτουργεί ως υπολογιστικό νέφος, μπορεί να δίνει εντολές μέσω κάποιας «έξυπνης » συσκευής (SP, tablet κ.α). Εφόσον προηγουμένως έχει πραγματοποιηθεί σύνδεση και γνωστοποίηση των «έξυπνων» συσκευών που υπάρχουν στο «έξυπνο» σπίτι, οι εντολές που στέλλει ο χρήστης αποστέλλονται πρώτα στο υπολογιστικό νέφος το οποίο με την σειρά που στέλλει την εντολή στην εξειδικευμένη πλατφόρμα Samsung και τέλος η πλατφόρμα είναι υπεύθυνη έτσι ώστε να καθορίσει ποια «έξυπνη» συσκευή πρέπει να εκτελέσει κάποια ενέργεια. Αυτή η διαδικασία εκτελείται κάθε φορά που ο χρήστης επιθυμεί να αλλάξει μια συγκεκριμένη κατάσταση μιας «έξυπνης» συσκευής ή και για να λάβει κάποιες πληροφορίες όπως για παράδειγμα λήψη βίντεο για παρακολούθηση κάποιου δωματίου ή για να ενημερωθεί σχετικά με μετρήσεις πχ θερμοκρασία , υγρασία κ.α. Στην περίπτωση που δημιουργηθεί κάποιο γεγονός από κάποια «έξυπνη» συσκευή / αισθητήρα τότε πραγματοποιείται η αντίστροφη επικοινωνία όπως φαίνεται στο Σχήμα 6.9 .

Αρχιτεκτονική REST

Η πλατφόρμα Smart Things μπορεί να χρησιμοποιηθεί με την χρήση της αρχιτεκτονικής REST(**R**epresentational **S**tate **T**ransfer). Το REST παρέχει την δυνατότητα να δημιουργήσουμε διαδικτυακή εφαρμογή με την χρήση του πρωτοκόλλου HTTP. Τα 3 βασικά στοιχεία που αποτελούν αυτή την αρχιτεκτονική είναι τα εξής:

Πρόσβαση σε πόρους μέσω κάποιας διεύθυνσης:

Αποτελούν τα στοιχεία που είναι διαθέσιμα σε ένα εξυπηρετητή και είναι διαθέσιμα στον χρήστη του συστήματος(client). Σε αυτή την περίπτωση , με τον όρο στοιχεία

εννοούμε τις νοητές «έξυπνες» συσκευές και «έξυπνους» αισθητήρες που παρέχει η πλατφόρμα Smart Things. Για να μπορεί ο χρήστης να έχει πρόσβαση σε αυτά τα δεδομένα χρειάζεται να έχει εξουσιοδότηση και πιστοποίηση η οποία παρέχεται μέσω κλειδιού που του παρέχεται όταν εγγραφεί στην πλατφόρμα Smart Things. Όταν ολοκληρωθεί επιτυχώς η προηγούμενη διαδικασία, ο χρήστης λαμβάνει μια συγκεκριμένη διεύθυνση η οποία πρέπει να χρησιμοποιείται (σε συνδυασμό με μέθοδο HTTP και όνομα του πόρου) .

Χρήση HTTP μεθόδων :

Το πρωτόκολλο HTTP παρέχει τις μεθόδους GET, POST, PUT και DELETE τις οποίες μπορούμε να τις χρησιμοποιήσουμε έτσι ώστε να εκτελούμε διαφορετικές ενέργειες στους πόρους του συστήματος. Δηλαδή αν γνωρίζουμε το όνομα του πόρου που θέλουμε να διαχειριστούμε τότε εφαρμόζοντας την κατάλληλη μέθοδο μπορούμε να επιτύχουμε το επιθυμητό αποτέλεσμα. Πιο κάτω ακολουθούν μερικά παραδείγματα χρήσης των μεθόδων για το φωτισμό.

Παράδειγμα 1

HTTP GET: <https://graph.api.smarthings.com/api/smartapps/installations/045e1ab6-9c82-4e1b-9362-dee29c29558c/switches> . Με αυτή την εντολή εξασφαλίζεται η πρόσβαση στους πόρους switches(φωτισμός δωματίων) έτσι ώστε να μπορούμε στην συνέχεια να εκτελέσουμε την εντολή turn on/off.

Παράδειγμα 2

HTTP PUT: <https://graph.api.smarthings.com/api/smartapps/installations/045e1ab6-9c82-4e1b-9362-dee29c29558c/switches/on/kitchen> . Με αυτή τη εντολή μπορούμε να αλλάξουμε την κατάσταση του φωτισμού. Σε αυτό το παράδειγμα εκτελείται η εντολή turn on. Αν επιθυμούμε να εκτελέσουμε την εντολή turn off απλά πρέπει να αντικαταστήσουμε την λέξη on με την λέξη off.

Γενικά για να πραγματοποιείται αλληλεπίδραση του χρήστη με τις «έξυπνες» συσκευές χρησιμοποιείται η ακόλουθη γενική μορφή διεύθυνσης:

DEFAULT_URL/ TOKEN/ HANDLER/OPERATION/ DEVICE

Ακολουθεί επεξήγηση της γενικής μορφής διεύθυνσης για το παράδειγμα 2:

DEFAULT_URL: <https://graph.api.smartthings.com/api/smartapps/installations/> .

Αποτελεί την προκαθορισμένη διεύθυνση του παροχέα υπηρεσιών.

TOKEN: Αποτελεί μια γραμματοσειρά συμβόλων, γραμμάτων και αριθμών που είναι μοναδική για κάθε χρήστη που χρησιμοποιεί τις διαδικτυακές υπηρεσίες του παροχέα. Η προκαθορισμένη διεύθυνση ακολουθημένη από το token αποτελούν το API Endpoint το οποίο είναι μοναδικό για κάθε χρήστη. Αντιστοιχεί στην γραμματοσειρά 045e1ab6-9c82-4e1b-9362-dee29c29558c.

HANDLER: Αποτελεί το στοιχείο που υπάρχει ήδη στον εξυπηρετητή και αντιστοιχεί σε μια ομάδα «έξυπνων» συσκευών. Στην περίπτωση του παραδείγματος αντιστοιχεί στην λέξη switches.

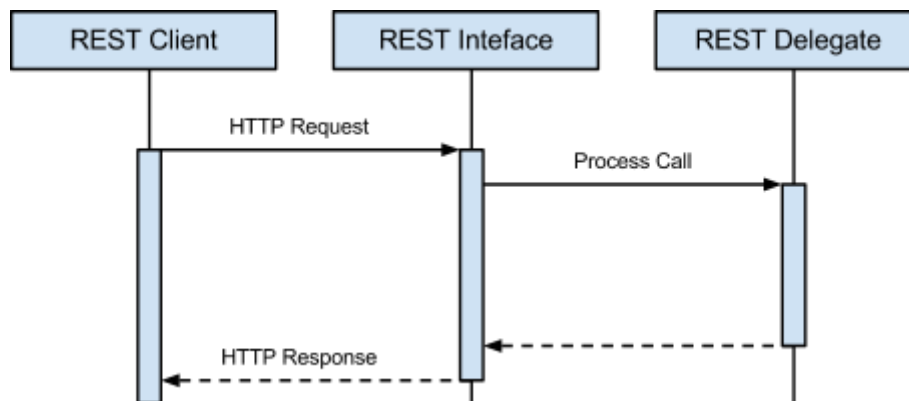
OPERATION: Κάθε «έξυπνη» συσκευή παρέχει διαφορετικές λειτουργίες, και έτσι σε αυτό το πεδίο θα πρέπει να προσδιοριστεί η εντολή που πρέπει να εκτελεστεί. Στην περίπτωση του φωτισμού οι λειτουργίες που μπορούν να χρησιμοποιηθούν είναι on και off. Στην περίπτωση του παραδείγματος αντιστοιχεί στην λέξη on.

DEVICE: Αποτελεί την «έξυπνη» συσκευή ή το δωμάτιο στο οποίο βρίσκεται η «έξυπνη» συσκευή. Στην περίπτωση που ο HANDLER αποτελείται από 1 στοιχείο («έξυπνη» συσκευή) τότε παραλείπεται αλλιώς καθορίζεται μια λέξη που αντιστοιχεί στον χώρο που βρίσκεται η «έξυπνη» συσκευή. Στην περίπτωση του παραδείγματος 2 επειδή υπάρχει φωτισμός σε διάφορα δωμάτια έχει καθοριστεί η λέξη kitchen έτσι ώστε να εκτελεστεί η εντολή στο φωτισμό της κουζίνας.

Παρουσίαση στοιχείων:

Η παρουσίαση στοιχείων αποτελεί το 3^ο βασικό στοιχείο της αρχιτεκτονικής REST. Αυτό το χαρακτηριστικό παρέχει την δυνατότητας διαφορετικής αναπαράστασης των δεδομένων που θα παρουσιαστούν στον χρήστη (HTML, XML κ.α.). Στην περίπτωση της εφαρμογής SAG δεν παρουσιάζονται στοιχεία εφόσον εκτελούνται οι 2 πιο πάνω μέθοδοι για να αλλάξουμε την κατάσταση «έξυπνων» συσκευών ή για να λάβουμε

διάφορες μετρήσεις από αισθητήρες οι οποίες και εμφανίζονται στην οθόνη του smart phone.

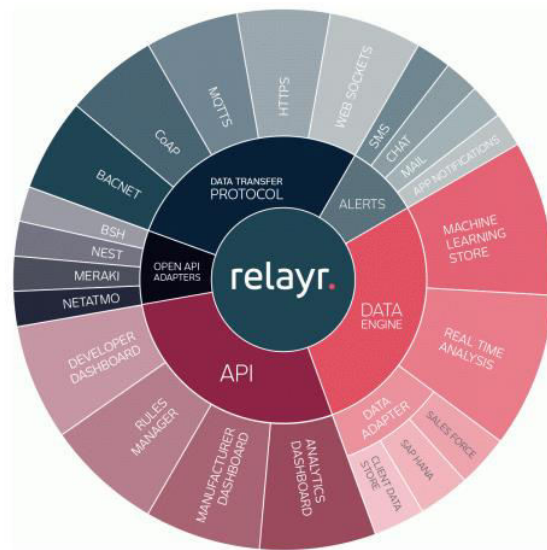


Σχήμα 6.9 Τρόπος λειτουργίας της αρχιτεκτονικής REST

Πλατφόρμα wunderbar

Σε αυτή την πλατφόρμα χρησιμοποιείται το πρωτόκολλο MQTT. Το πρωτόκολλο αυτό είναι υπεύθυνο για την ανάγνωση των μετρήσεων των διάφορων αισθητήρων που είναι συνδεδεμένοι στο wunderbar και την αποστολή αυτών των μετρήσεων σε ένα κεντρικό σύστημα το οποίο με την σειρά του θα στείλει αυτές τις πληροφορίες στο relayr cloud μέσω του τοπικού δικτύου. Δηλαδή το MQTT αποτελεί το πρωτόκολλο εγκατάστασης και επικοινωνίας των αισθητήρων έτσι ώστε να μπορούμε να λαμβάνουμε τις διαθέσιμες μετρήσεις [15]. Επίσης το relayr έχει διαθέσιμα πολλά πρωτόκολλα τα οποία παρέχουν διαφορετικές λειτουργίες όπως φαίνεται στην Σχήμα 6.10. Για την εφαρμογή SAG έχει χρησιμοποιηθεί το relayr JAVA SDK το οποίο παρέχει την δυνατότητα διαχείρισης συσκευών και αισθητήρων που είναι συνδεδεμένοι στο wunderbar. Πιο συγκεκριμένα χρησιμοποιεί πιστοποίηση μέσω κάποιου κλειδιού (token) έτσι ώστε να καταστεί εφικτή η πρόσβαση στο relayr Cloud και έπειτα για να έχουμε πρόσβαση στις συσκευές που έχουμε εγκαταστήσει στο wunderbar θα πρέπει να εισάξουμε τα κλειδιά (id) τους. Τα κλειδιά λειτουργούν ως access points με τα οποία μπορούμε να καλέσουμε διάφορες μεθόδους που έχουν καθοριστεί από το relayr

API για να παραλάβουμε μετρήσεις των αισθητήρων ή για να εκτελέσουμε μια ενέργεια σε μια συγκεκριμένη συσκευή.



Σχήμα 6.10 : Πρωτόκολλα που υποστηρίζει το relayr Cloud

Αρχιτεκτονική πλατφόρμας wunderbar

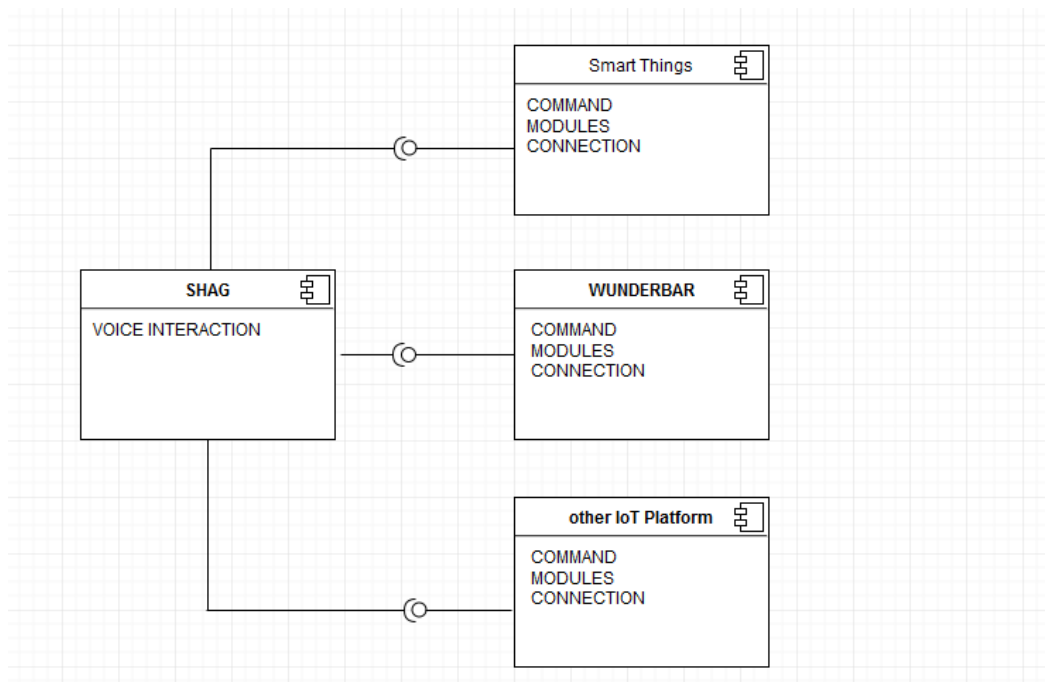
Στο σχήμα 6.11 παρουσιάζεται η αρχιτεκτονική της πλατφόρμας wunderbar. Ο χρήστης θα πρέπει πρώτα να εγκαταστήσει τους αισθητήρες στο «έξυπνο» σπίτι και έπειτα να κάνει εγγραφή στο υπολογιστικό νέφος relayr μέσω του οποίου θα μπορεί να αλληλεπιδρά με τους διαθέσιμους αισθητήρες ή με τον μικροελεγκτή Arduino ο οποίος μπορεί να εκτελεί διάφορες ενέργειες στις συσκευές που είναι συνδεδεμένες σε αυτό. Η επικοινωνία των συστατικών στοιχείων που γίνεται στην πλατφόρμα παρουσιάζει ομοιότητες με την πλατφόρμα Smart things όπως για παράδειγμα η σύνδεση σε κάποιο υπολογιστικό νέφος καθώς επίσης και η σύνδεση του υπολογιστικού νέφους με ένα κεντρικό εξυπηρετητή (στην περίπτωση του smart things το smart hub) που σε αυτή την περίπτωση είναι το main module. Όπως βλέπουμε και στην εικόνα η επικοινωνία μεταξύ εφαρμογής και πλατφόρμας wunderbar επιτυγχάνεται μέσω της σύνδεσης αυτών των 2 οντοτήτων με το υπολογιστικό νέφος relayr.



Σχήμα 6.11 Αρχιτεκτονική wunderbar

6.4 Συνδεδετικά συστατικά υποσυστήματα

Τα συνδεδετικά συστατικά υποσυστήματα αποτελούν ένα ανεξάρτητο μέρος ενός προγράμματος το οποίο εξυπηρετεί ένα συγκεκριμένο στόχο ,δηλαδή παρέχει υπηρεσίες σε ένα πρόγραμμα έτσι ώστε να πραγματοποιούνται οι κατάλληλες ενέργειες για να παραχθεί το επιθυμητό αποτέλεσμα. Στην περίπτωση της εφαρμογής SAG ένα συστατικό υποσύστημα αποτελείται από το πρότυπο Command καθώς επίσης και μια συγκεκριμένη πλατφόρμα IoT. Εφόσον έχουν καθοριστεί οι έξυπνες συσκευές καθώς επίσης και τα σενάρια που πρόκειται να υλοποιηθούν στην εφαρμογή SAG μπορούμε εύκολα να επαναχρησιμοποιήσουμε τις κλάσεις των συσκευών καθώς και την διεπαφή Command και να υλοποιήσουμε μια διαφορετική διεπαφή που να παρέχει την δυνατότητα σύνδεσης και επικοινωνίας με την νέα πλατφόρμα που θέλουμε να προσθέσουμε.



Σχήμα 6.12 Η εφαρμογή SAG σε μορφή υποσυστημάτων

Πλεονεκτήματα χρήσης συστατικών υποσυστημάτων:

- Παρέχει καλύτερη αποσφαλμάτωση λαθών: Κάθε πλατφόρμα αποτελεί ένα ξεχωριστό συστατικό του συστήματος (module) και έτσι ανάλογα με ποια πλατφόρμα παρουσιάζει κάποιο πρόβλημα μπορούμε να έχουμε πιο γρήγορη αντιμετώπιση.
- Ευελιξία: Η εφαρμογή που χρησιμοποιεί συστατικά υποσυστήματα παρέχει την δυνατότητα να χρησιμοποιήσουμε περισσότερες πλατφόρμες IoT χωρίς να χρειάζεται να εγκαταστήσουμε άλλες εφαρμογές .
- Επαναχρησιμοποίηση: Λόγω της ανεξαρτησίας του συστατικών υποσυστημάτων μπορεί να γίνει επαναχρησιμοποίηση κάποιου συστατικού σε παρόμοια εφαρμογή.[7]
- Αντικατάσταση : Σε περίπτωση μη υποστήριξης (κατάργησης) κάποιας πλατφόρμας IoT μπορεί εύκολα να αφαιρεθεί το αντίστοιχο συστατικό χωρίς να επηρεάζεται η λειτουργικότητα /απόδοση του υπόλοιπου συστήματος.[7]

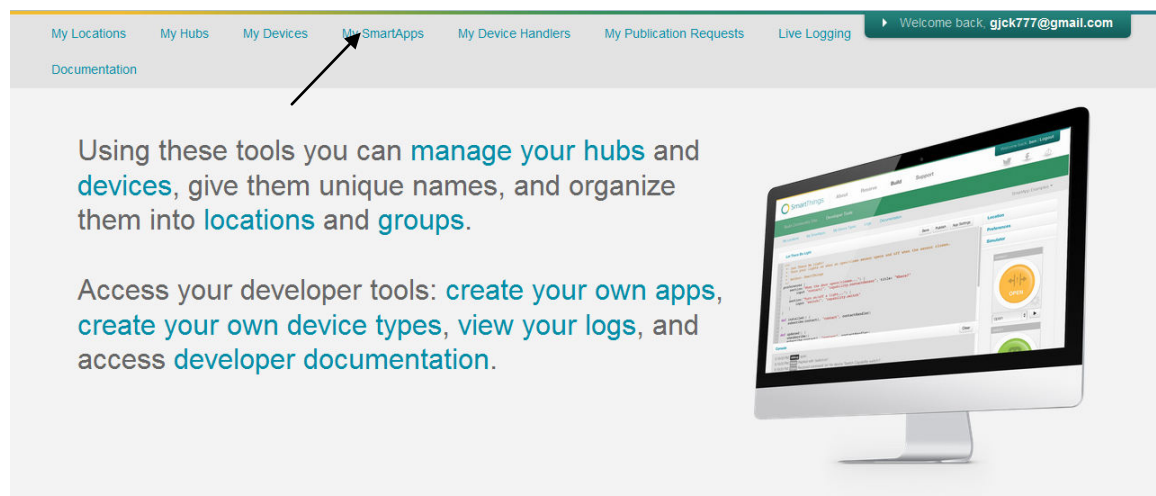
6.5 Απαιτήσεις εφαρμογής

- Για την επιτυχή λειτουργία της εφαρμογής έχουν γίνει οι παρακάτω προϋποθέσεις – απαιτήσεις:
- Δημιουργία λογαριασμού στην πλατφόρμα «έξυπνου» σπιτιού Smart Things. Για να μπορούμε να αλληλεπιδρούμε με τον προσομοιωτή που διατίθεται από την πλατφόρμα Smart Things πρέπει να γίνει εγγραφή του χρήστη έτσι ώστε να δίνει φωνητικές εντολές στις «έξυπνες» συσκευές που διαθέτει η πλατφόρμα.
- Δημιουργία λογαριασμού στο relayr cloud έτσι ώστε ο χρήστης να μπορεί να έχει πρόσβαση στους αισθητήρες και στις συσκευές που είναι συνδεδεμένες στο Wunderbar.
- Ο χρήστης θα πρέπει να έχει smart phone με λειτουργικό σύστημα Android με έκδοση Jelly Bean ή μεταγενέστερη έκδοση (από έκδοση 4.1 και πάνω).
- Το smart phone θα πρέπει να επιτρέπει τις εξής άδειες και προγράμματα (permissions και apps):
 - I. Internet για επικοινωνία με τις πλατφόρμες «έξυπνου» σπιτιού
 - II. Access network state : επιτρέπει την παροχή πληροφοριών σχετικά με το δίκτυο(αναγκαίο για την πλατφόρμα «έξυπνου» σπιτιού WunderBar)
 - III. Record audio: επιτρέπει την εγγραφή φωνής του χρήστη έτσι ώστε να μπορεί να εκτελεί τις φωνητικές εντολές
 - IV. Vibration: επιτρέπει την δόνηση του «έξυπνου» κινητού τηλεφώνου όταν ο χρήστης πατήσει το εικονίδιο για φωνητική εντολή.
 - V. Στο smart phone θα πρέπει να υπάρχει εγκατεστημένο το πρόγραμμα google text to speech έτσι ώστε να μπορεί να υπάρχει κατάλληλη φωνητική απάντηση σε κάποια εντολή που έχει δώσει προηγουμένως ο χρήστης.
 - VI. Στο smart phone θα πρέπει να είναι ενεργοποιημένο το Google text to speech με εγκατεστημένη γλώσσα English United States.

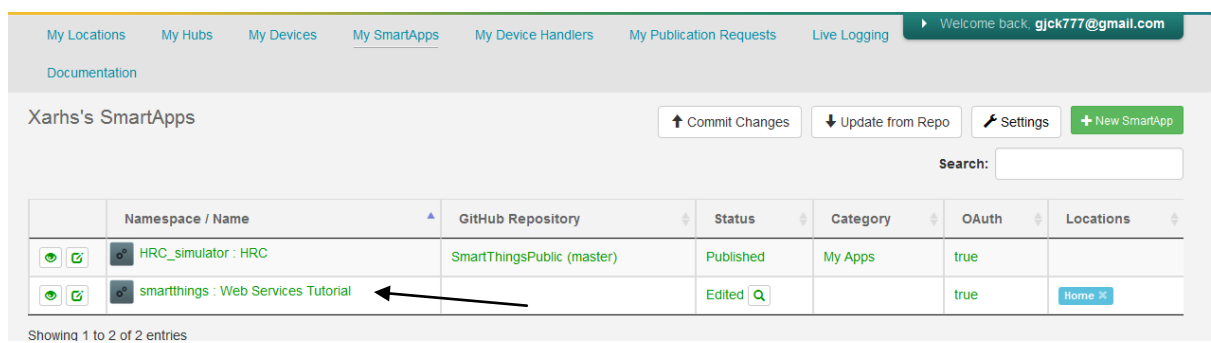
6.6 Λειτουργία εφαρμογής

Σε αυτό το υποκεφάλαιο γίνεται μια περιγραφή της εφαρμογής SAG με την χρήση στιγμιότυπων οθόνης και επεξήγηση τους(τα screen shots έχουν γίνει με το smart phone Sony Xperia Z3).

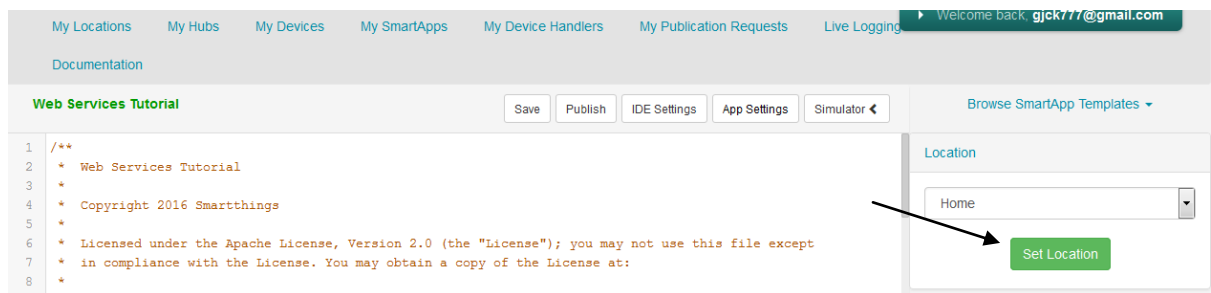
Στην εικόνα 6.17 ο χρήστης καλείται να επιλέξει την πλατφόρμα «έξυπνου» σπιτιού με την οποία επιθυμεί να εκτελεί τις φωνητικές εντολές . Για παράδειγμα αν επιλέξει την πλατφόρμα Smart Things τότε θα μπορέσει να δώσει φωνητικές εντολές οι οποίες θα γίνονται στον προσομοιωτή της πλατφόρμας. Σε αυτή την περίπτωση θα πρέπει να συνδεθεί με τον λογαριασμό του Smart Things που έχει ήδη δημιουργήσει και να ακολουθήσει τις διαδικασίες που φαίνονται στις πιο κάτω εικόνες.



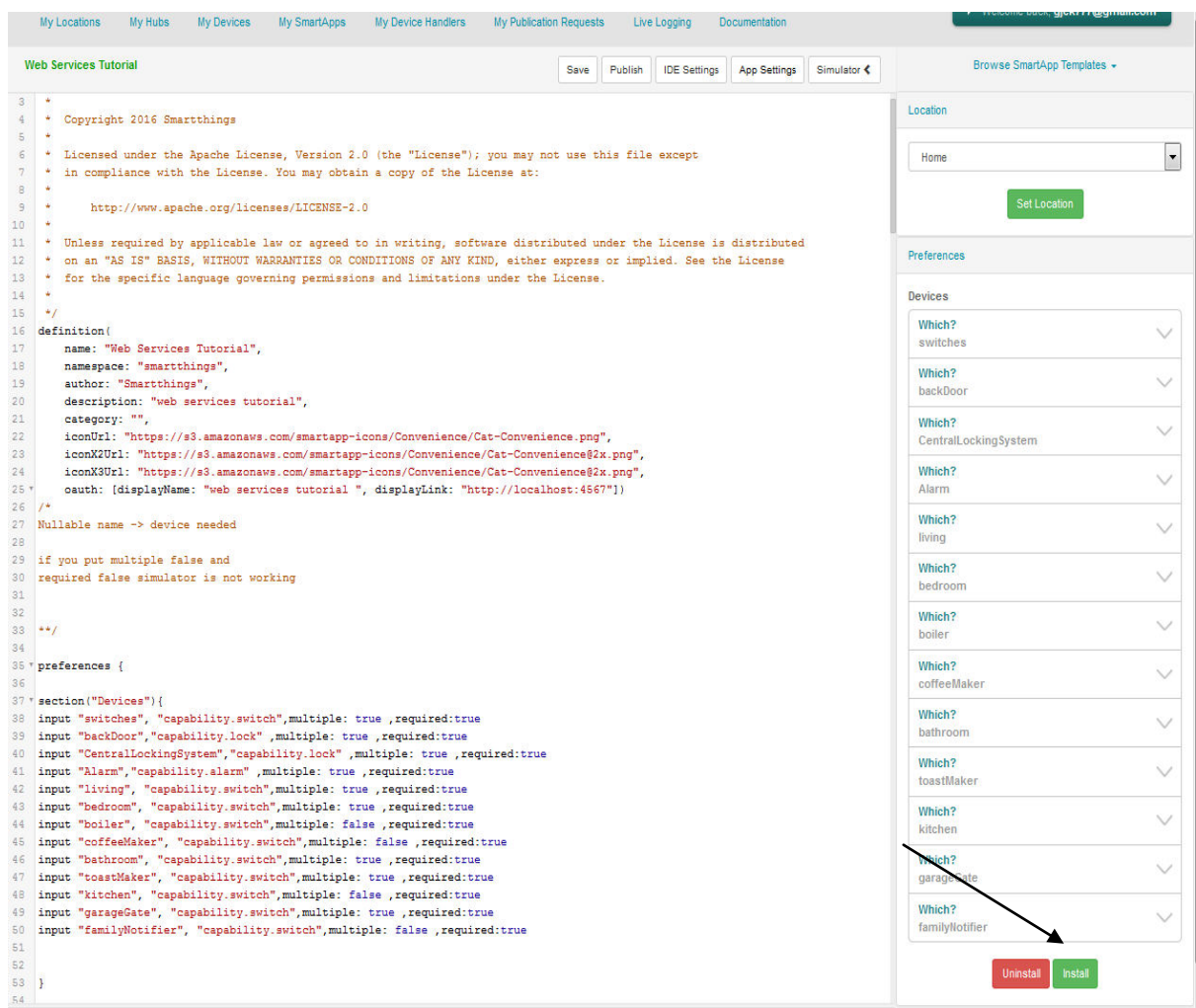
Σχήμα 6.13 Επιλογή εφαρμογής που έχει δημιουργηθεί στην πλατφόρμα Smart Things



Σχήμα 6.14 Επιλογή του προγράμματος προσομοίωσης

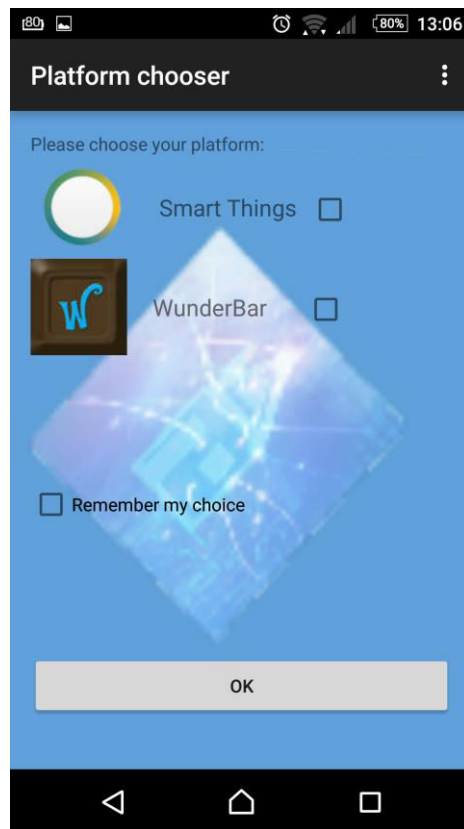


Σχήμα 6.15 Καθορισμός χώρου εργασίας



Σχήμα 6.16 Εγκατάσταση «έξυπνων» συσκευών προσομοίωσης

Κατά την εκκίνηση της εφαρμογής ο χρήστης θα πρέπει να επιλέξει τις πλατφόρμες που επιθυμεί να χρησιμοποιήσει .



Σχήμα 6.17 Επιλογή πλατφόρμων IoT



Σχήμα 6.18 Μενού εφαρμογής

Στο πιο πάνω σχήμα φαίνεται το Μενού στο οποίο θα δίνει φωνητικές εντολές ο χρήστης. Τα διάφορα κουμπιά που υπάρχουν μπορούν είτε να παρέχουν κάποιες πληροφορίες στον χρήστη είτε να μπορεί να ενεργοποιεί την λειτουργία διάφορων έξυπνων συσκευών. Ακολουθεί πίνακας στον οποίο εξηγούνται οι λειτουργίες των κουμπιών της πιο πάνω σχήματος.

Κατηγορίες κουμπιών :

Activation button: με το πάτημα ενός κουμπιού ανάβουν οι λάμπες τότε ανήκει στην κατηγορία activation

informational button : παρέχει πληροφορίες στον χρήστη τότε θα ανήκει στην κατηγορία informational.

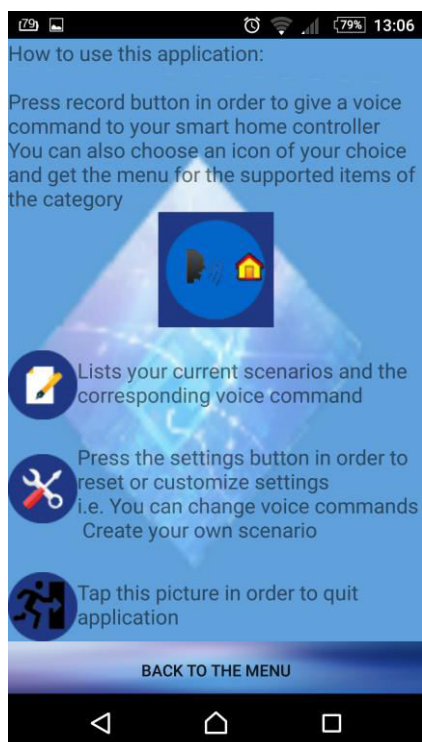
both: activation & informational

monitoring: παρακολούθηση μιας κατάστασης

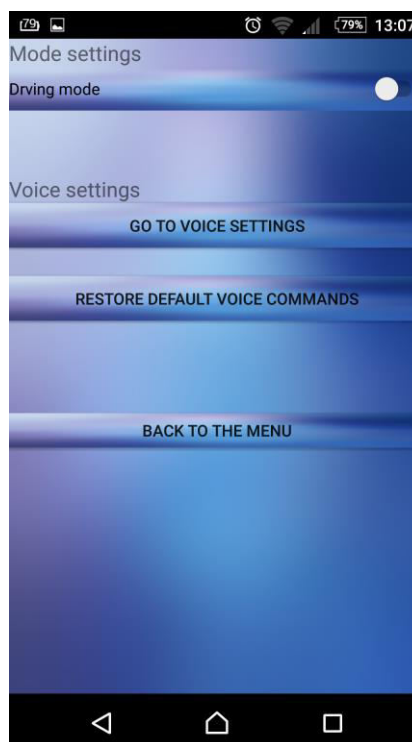
Αριθμός κουμπιού	Περιγραφή	Κατηγορία
1	Παρέχει πληροφορίες σχετικά με τον τρόπο λειτουργίας της εφαρμογής	Informational
2	Όταν ο χρήστης πατήσει αυτό το κουμπί τότε ενεργοποιεί την φωνητική εγγραφή έτσι ώστε να μπορεί να δώσει την φωνητική εντολή.	Activation
3	Έξοδος από το πρόγραμμα	Activational
4	Ρυθμίσεις της εφαρμογής(driving/normal mode, modify voice commands)	Activational
5	Σενάρια (εμφάνιση διαθέσιμων σεναρίων, ενεργοποίηση απενεργοποίηση σεναρίων)	Both
6	«Έξυπνο ρολόι» . Ρύθμιση «έξυπνου» ρολογιού	Both
7	Panic button . Το «έξυπνο» σπίτι έχει παραβιαστεί(απενεργοποίηση- κλείδωμα όλων των «έξυπνων» συσκευών)	Activational
8	Ένδειξη θερμοκρασία νερού	Informational
9	«Έξυπνες» συσκευές . Ενεργοποίηση – απενεργοποίηση «έξυπνων» συσκευών (επιλογή «έξυπνων» συσκευών ανάλογα με την κατηγορία πχ. Φωτισμός, «έξυπνες» συσκευές της κουζίνας, υπνοδωματίου κ.α.)	Activational
10	Ασφάλεια «έξυπνου» σπιτιού. Ένδειξη κατάστασης για τα «έξυπνα» παράθυρα – πόρτες. Ενεργοποίηση – Απενεργοποίηση «έξυπνου» σπιτιού .	Both
11	Ένδειξη διαφόρων μετρήσεων(υγρασία, θερμοκρασία, ηλιοφάνεια κτλ.)	Informational
12	«Έξυπνος» φωτισμός(ενεργοποίηση – απενεργοποίηση φωτισμού , αλλαγή	Activational

	φωτεινότητας)	
13	Ενεργοποίηση – απενεργοποίηση ασφάλειας οχημάτων	Activational
14	Ενεργοποίηση-απενεργοποίηση σεναρίου εξοικονόμησης ενέργειας	Activational
15	Ενεργοποίηση-απενεργοποίηση σεναρίου διασκέδασης	Activational
16	Παρακολούθηση χώρων από τις κάμερες ασφαλείας	Monitoring

Πιο κάτω εμφανίζονται οι οθόνες όταν ο χρήστης πατήσει τα κουμπιά 1,4,6 και 11 αντίστοιχα.



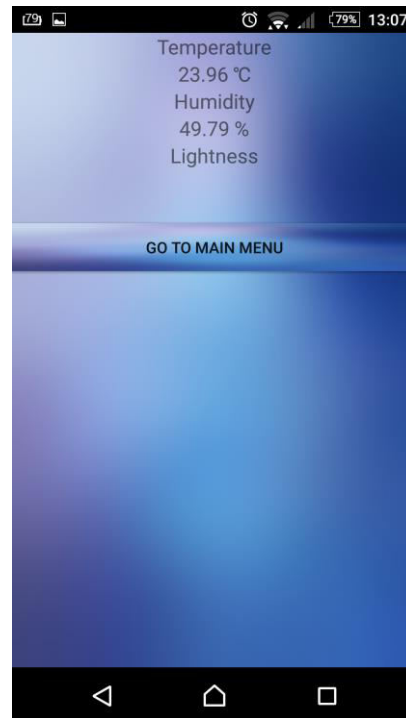
Σχήμα 6.19 Οθόνη βοήθειας



Σχήμα 6.20 Οθόνη Ρυθμίσεων



Σχήμα 6.21 Οθόνη λειτουργίας οδήγησης



Σχήμα 6.22 Οθόνη εμφάνισης μετρήσεων

6.7 Σύγκριση εφαρμογής με παρόμοιες εφαρμογές

Παρόμοιες εφαρμογές που έχουν την δυνατότητα να παρέχουν αλληλεπίδραση με το «έξυπνο» σπίτι περιορίζονται στην εκτέλεση εντολών με βάση διαφόρων χειριστηρίων(κουμπιών) που παρέχονται μέσα από την γραφική διεπαφή του χρήστη . Πιο κάτω επισημαίνονται μερικές παρόμοιες εφαρμογές που παρέχουν την δυνατότητα αλληλεπίδρασης με την αντίστοιχη πλατφόρμα «έξυπνου» σπίτιου.

Control4:

Μερικά χαρακτηριστικά της εφαρμογής Control4 είναι:

- audio playback
- video control
- Lighting control
- Heating & A/C control
- Πρόσβαση από οπουδήποτε (προϋποθέτει συνδρομή στην εταιρεία 4Sight)

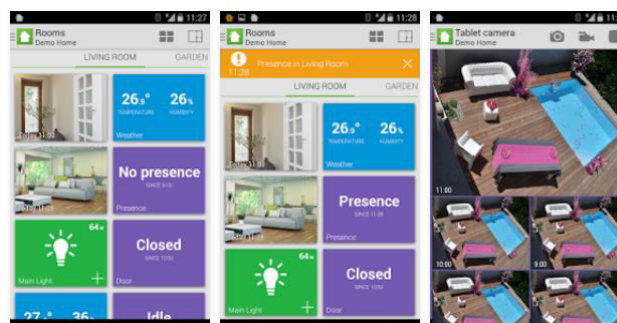
Σε αυτή την εφαρμογή υπάρχουν 6 κατηγορίες οι οποίες είναι Watch ,Listen, Lighting,Comfort, Security, More. Επιλέγοντας μια από αυτές τις κατηγορίες ο χρήστης μεταφέρεται σε μια νέα οθόνη που περιλαμβάνει διάφορες «έξυπνες» συσκευές της κατηγορίας.Στο σχήμα δσφσδ έχει επιλεχθεί η κατηγορία φωτισμός η οποία περιέχει διάφορες ρυθμίσεις σχετικά με τον φωτισμό.[22]



Σχήμα 6.23 Εφαρμογή Control4

Archos:

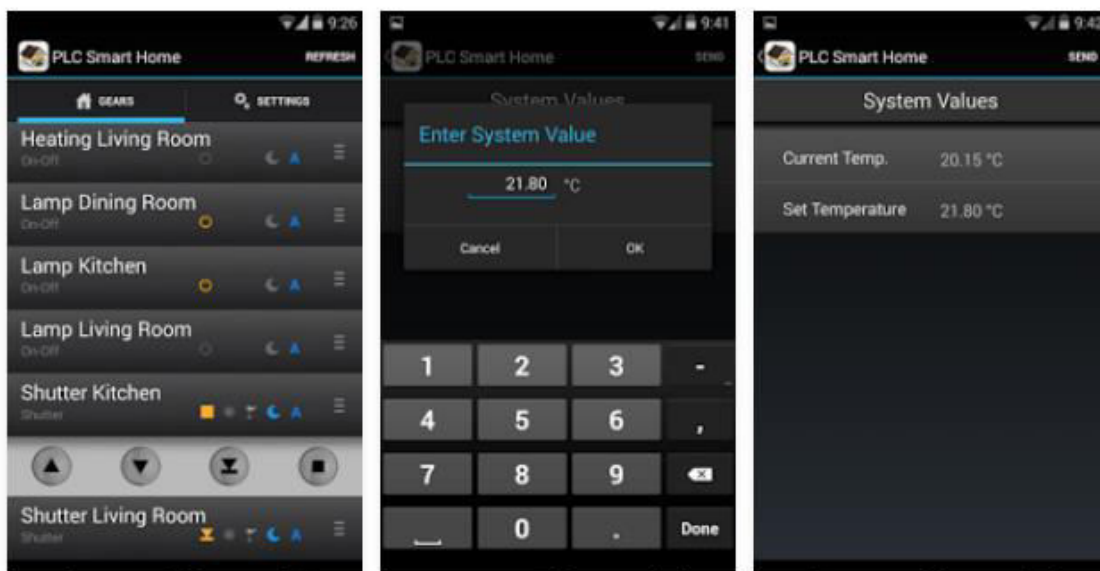
Αυτή η πλατφόρμα «έξυπνου» σπιτιού διαθέτει αισθητήρες έτσι ώστε να εκτελούνται διάφορες ενέργειες σε περίπτωση που παρουσιαστεί κάποια αλλαγή.Για παράδειγμα motion ball που ανιχνεύει την κίνηση σε ένα δωμάτιο,weather tag για μέτρηση θερμοκρασίας και υγρασίας , smart plug για εξοικονόμηση ενέργειας κ.α.Ο χρήστης μπορεί να ελέγξει το δωμάτιο που επιθυμεί εφόσον η εφαρμογή κατηγοριοποιεί τις «έξυπνες» συσκευές ανάδωμάτιο όπως φαίνεται και στο Σχήμα 6.24.[23]



Σχήμα 6.24 Εφαρμογή Archos

PLC Smart Home:

Σε αυτή την πλατφόρμα γίνεται χρήση διάφορων Crouzet PLC controls που κατασκευάζονται από την ίδια την εταιρεία. Η εταιρεία δημιουργεί διάφορα χειριστήρια έτσι ώστε να έχουμε έλεγχο σε διάφορες «έξυπνες» συσκευές. Μερικά παραδείγματα τέτοιων χειριστηρίων είναι roller shutter control, blind control, radio roller shutter control κ.α. Το πλεονέκτημα που παρουσιάζουν αυτές οι «έξυπνες» συσκευές είναι το χαμηλό κόστος αγοράς τους. Ο χρήστης χρησιμοποιώντας την εφαρμογή plc smart home μπορεί να ρυθμίσει αυτά τα χειριστήρια ανάλογα με τις προτιμήσεις του όπως φαίνεται και στο σχήμα 6.25. [24]



Σχήμα 6.25 Εφαρμογή PLC Smart Home

Η εφαρμογή SAG θα πρέπει να έχει παρόμοια κατηγοριοποίηση (όπως με τις προηγούμενες εφαρμογές) έτσι ώστε όταν κάποιος χρήστης πατήσει το κουμπί 9 όπως φαίνεται στο σχήμα 6.18 να μπορεί να βρίσκει με εύκολο τρόπο την «έξυπνη» συσκευή που επιθυμεί. Η εφαρμογή παρέχει επιπλέον τις εξής δυνατότητες:

- υποστηρίζει δυνατότητα εκτέλεσης εντολών μέσα από την φωνητική αλληλεπίδραση. Στο Σχήμα 6.18 βλέπουμε στο πάνω κεντρικό σημείο της

διεπαφής ένα εικονίδιο το οποίο αναπαριστά μια φιγούρα που παράγει ήχο(το οποίο όταν το ενεργοποιήσει ο χρήστη θα μπορέσει να δώσει φωνητική εντολή).

- Χρήση φωνητικών ειδοποιήσεων κατά την επιτυχή/ ανεπιτυχή εκτέλεση μιας ενέργειας
- υποστηρίζει περισσότερες πλατφόρμες σε αντίθεση με τις άλλες εφαρμογές που είναι υλοποιημένες σε μια συγκεκριμένη πλατφόρμα.(Σχήμα 6.17)
- με την δυνατότητα φωνητικής αλληλεπίδρασης ο χρήστης μπορεί να οδηγήσει χωρίς να χρειάζεται να αποσπάται η προσοχή του στην εύρεση του αντίστοιχου κουμπιού.(Σχήμα 6.21)

Κεφάλαιο 7

Πειραματισμός - Αξιολόγηση

7.1 Εισαγωγή	72
7.2 Αξιολόγηση	75

7.1 Εισαγωγή

Η εφαρμογή SAG έχει υλοποιηθεί στο Android Studio και έχει δοκιμαστεί στο SP Sony Z3. Για την καλύτερη αξιολόγηση της εφαρμογής έχουν γίνει δοκιμές και έχουν απαντηθεί ερωτηματολόγια με τα οποία έχουν δημιουργηθεί γραφικές παραστάσεις σχετικά με την απόδοση της εφαρμογής. Ακολουθούν οι ερωτήσεις του ερωτηματολογίου καθώς επίσης και οι γραφικές παραστάσεις για τα 3 πιο σημαντικά χαρακτηριστικά της εφαρμογής η ευχρηστία, ο χρόνος α και η φωνητική αλληλεπίδραση .

Ερωτήσεις που έχουν γίνει στο ερωτηματολόγιο:

1. Ευχρηστία συστήματος:

Η εφαρμογή έχει ευκολία στην πλοήγηση ;

Η εφαρμογή έχει κατανοητή διεπαφή

	1	2	3	4	5	
Διαφωνώ πλήρως	☐	☐	☐	☐	☐	Συμφωνώ πλήρως

2. Απόδοση του συστήματος:

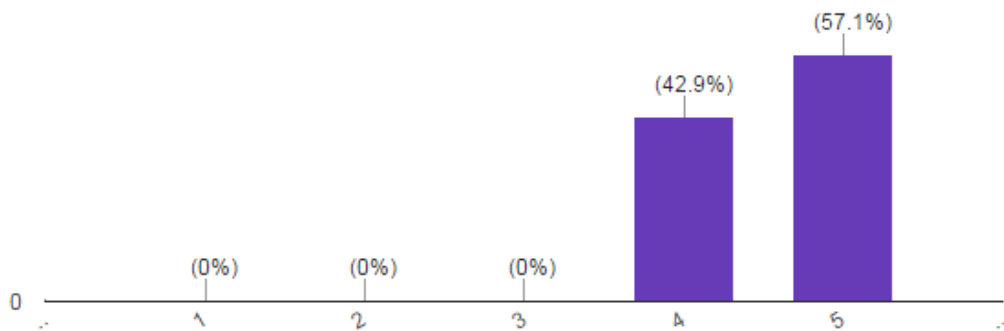
Πως θα χαρακτηρίζατε το χρόνο απόκρισης της εφαρμογής

	1	2	3	4	5	
Αργό	☐	☐	☐	☐	☐	Γρήγορο

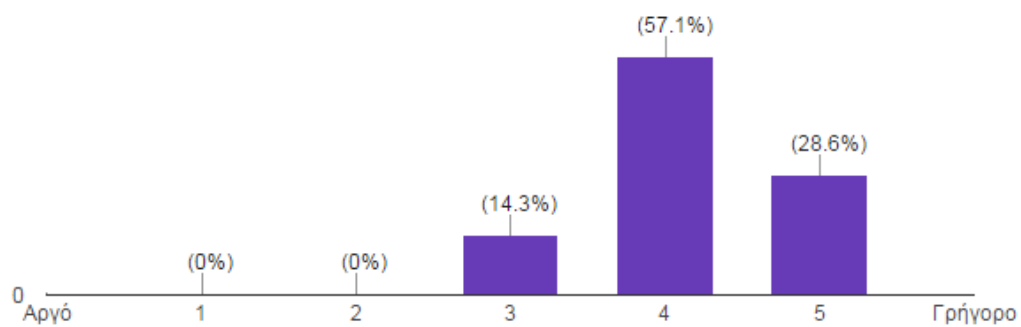
3. Φωνητική αλληλεπίδραση:

Τα φωνητικά μηνύματα βοηθούν τον χρήστη (αυτοματοποίηση λειτουργιών - ειδοποιήσεις)

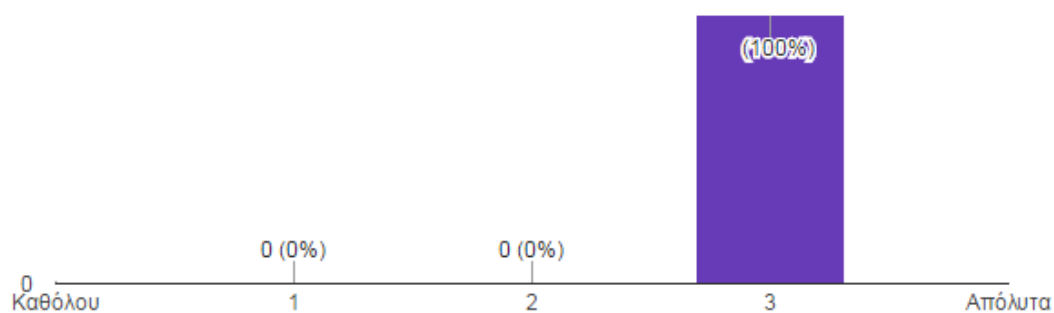
	1	2	3	
Καθόλου	☐	☐	☐	Απόλυτα



Σχήμα 7.1 Γραφική παράσταση για την ευχρηστία του συστήματος (διεπαφή και ευκολία πλοήγησης)



Σχήμα 7.2 Χρόνος απόκρισης του συστήματος



Σχήμα 7.3 Λειτουργικότητα συστήματος (Φωνητική αλληλεπίδραση)

7.2 Αξιολόγηση

Από τις γραφικές παραστάσεις που έχουν δημιουργηθεί από τα ερωτηματολόγια καταλήγουμε στα πιο κάτω συμπεράσματα:

1. Η εφαρμογή καλύπτει την βασική λειτουργία για την οποία έχει δημιουργηθεί εφόσον από την Σχήμα 7.3 συμπεραίνουμε πως οι χρήστες που έχουν δοκιμάσει την εφαρμογή είναι ικανοποιημένοι όσον αφορά την φωνητική αλληλεπίδραση που δέχονται από την εφαρμογή.
2. Η εφαρμογή έχει αρκετά γρήγορη απόκριση στις φωνητικές εντολές που δέχεται εφόσον η πλειοψηφία των χρηστών υποστηρίζει ότι η εφαρμογή είναι αρκετά γρήγορη με ποσοστό 57.1 % ενώ το 28.6 % υποστηρίζει πως είναι γρήγορο (Σχήμα 7.2)
3. Το 57.1% των χρηστών που έχουν ερωτηθεί, συμφωνεί απόλυτα πως η εφαρμογή είναι εύκολη στην πλοήγηση και έχει κατανοητή διεπαφή ενώ το 42.9% υποστηρίζουν πως είναι αρκετά καλή. Αυτό καθιστά την εφαρμογή εύκολη τόσο στην χρήση όσο και στην πλοήγηση.

Κεφάλαιο 8

Συμπεράσματα

8.1 Συμπεράσματα	76
8.2 Μελλοντικές επεκτάσεις	77

8.1 Συμπεράσματα

Σε αυτή την διπλωματική εργασία έχει μελετηθεί ένας νέος τρόπος με τον οποίο μπορούμε να δώσουμε εντολές σε ένα SP η οποία ονομάζεται φωνητική αλληλεπίδραση. Με αυτή την μέθοδο μπορούμε να δώσουμε , με την χρήση της ομιλίας , φωνητικές εντολές με τις οποίες η μηχανή μπορεί να «καταλάβει» τι επιθυμεί ο χρήστης και να πραγματοποιήσει τις κατάλληλες εντολές.

Το IoT έχει αρχίσει να γίνεται όλο και πιο γνωστό μέσα από διάφορους τομείς όπως υγεία , αυτοματοποίηση , παρακολούθηση περιβάλλοντος και χαρακτηρίζεται από καινοτομία, «έξυπνάδα» και εφευρετικότητα. Επίσης αποτελεί απαρχή για την δημιουργία «έξυπνου» σπιτιού εφόσον το «έξυπνο» σπίτι αποτελείται από διάφορες «έξυπνες» συσκευές που επικοινωνούν μεταξύ τους έτσι ώστε να ανταλλάζουν πληροφορίες και να εκτελούν τις λειτουργίες- ενέργειες για τις οποίες έχουν καθοριστεί να εκτελέσουν.

Η εφαρμογή έχει δοκιμαστεί τόσο σε πλατφόρμα προσομοίωσης (Smart Things) τόσο και σε κανονική πλατφόρμα (Wunderbar). Η εφαρμογή μπορεί να λειτουργήσει και με τις 2 πλατφόρμες . Παρ' όλα αυτά παρατηρείται πιο γρήγορη απόκριση στην πλατφόρμα Smart Things παρά στην πλατφόρμα WunderBar και οφείλεται στο

γεγονός ότι η προσομοίωση δεν εκτελεί πραγματικές εντολές σε «έξυπνες» συσκευές με αποτέλεσμα να ολοκληρώνονται οι ενέργειες πιο γρήγορα. Αντίθετα η πλατφόρμα WunderBar έχει ενσωματωμένους αισθητήρες που καταμετρούν συνεχώς διάφορες μετρήσεις(θερμοκρασία, υγρασία κτλ) τις οποίες και αποστέλλει στο relay cloud από το οποίο παραλαμβάνει στο τέλος η εφαρμογή SAG.Αυτή η διαδικασία μπορεί να πραγματοποιείται γρήγορα όμως η εφαρμογή χρειάζεται λίγο περισσότερο χρόνο για να εκτελέσει τις εντολές που πραγματοποιούνται στην πλατφόρμα WunderBar.

8.2 Μελλοντικές επεκτάσεις

Η εφαρμογή SAG δεν έχει ολοκληρωθεί πλήρως , όμως υποστηρίζει απλές βασικές λειτουργίες που ικανοποιούν τις βασικές ανάγκες για τις οποίες χρειάστηκε να δημιουργηθεί αυτή η εφαρμογή. Παρακάτω επισημαίνονται μελλοντικές επεκτάσεις-βελτιώσεις οι οποίες μπορούν να γίνουν έτσι ώστε να μπορεί να είναι ακόμα καλύτερη και πιο αποδοτική:

1. Ολοκλήρωση λειτουργίας όλων των «έξυπνων» συσκευών και σεναρίων έτσι ώστε να εξυπηρετεί όλες τις ανάγκες για τις οποίες έχει προταθεί.
2. Χρήση περισσότερης «εξυπνάδας» του προγράμματος. Οι κλάδοι της Πληροφορικής Artificial Intelligent και Learning Machine είναι 2 αρκετά σημαντικοί στην ανάπτυξη αυτού του είδους εφαρμογών εφόσον με διάφορους αλγόριθμους και μεθόδους επιτρέπουν στην μηχανή να «σκέφτεται» και να «βγάζει» τα δικά της συμπεράσματα με βάση διάφορα δεδομένα που «αντιλαμβάνεται».
3. Δημιουργία τύπου χρηστών π.χ. Απλός χρήστης – Προχωρημένος χρήστης. Μπορούμε να επιτρέψουμε σε ένα άτομο (καλεσμένο) να χρησιμοποιήσει μερικές δυνατότητες που παρέχει το «έξυπνο» σπίτι . Σε αυτή την περίπτωση ο ιδιοκτήτης είναι ο Προχωρημένος Χρήστης ενώ ο επισκέπτης είναι ο Απλός χρήστης.
4. Επιθυμητή ασφάλεια σε περίπτωση που κλαπεί το κινητό τηλέφωνο του χρήστη. Σε περίπτωση κλοπής του SP ο «νέος» κάτοχος έχει πρόσβαση στο SH

με αποτέλεσμα να μπορεί ανενόχλητα να διαπράξει και κλοπή σε περίπτωση που δεν υπάρχει κανένας άτομο στο σπίτι. Θα πρέπει λοιπόν να παρέχεται ένας τρόπος με τον οποίο να εξακριβώνεται η ταυτότητα του ατόμου που εκτελεί εντολές είτε κανονικές μέσω των κουμπιών της εφαρμογής είτε φωνητικές .π.χ. μέσω δακτυλικών αποτυπωμάτων, αναγνώριση προσώπου, αναγνώριση φωνής κτλ.

Βιβλιογραφία

- [1] B. Danny and H.Pat , “Smart Home for Dummies”, page 12
- [2] G. Erich and J. Ralph and K. Richard and V. John , “Design patterns Elements of Reusable Object-Oriented Software”, page 23
- [3] S. Murray and I. Sara and A. Christopher , “A pattern language” , page x (10)
- [4] Z. Panayiotis and S. A. Chee , “Human Computer Interaction Concepts, Methodologies, Tools and Applications”, Chapter 1.5 “Human Factors in Public Information Systems”, page 121
- [5] Breville smart oven
<http://www.brevilleusa.com/the-smart-oven-r.html>
- [6] Eazy Tutz – Android Architecute
<http://www.eazytutz.com/android/android-architecture/>
- [7] Microsoft - Chapter 3: Architectural Patterns and Styles
<https://msdn.microsoft.com/en-us/library/ee658117.aspx#ComponentBasedStyle>
- [8] Phone arena - sensors
http://www.phonearena.com/news/Did-you-know-how-many-different-kinds-of-sensors-go-inside-a-smartphone_id57885
- [9] Smart grid – Energy management
https://www.smartgrid.gov/the_smart_grid/smart_home.html
- [10] Statista – Market share
<http://www.statista.com/statistics/272307/market-share-forecast-for-smartphone-operating-systems/>

- [11] Wikipedia – Android operating system
https://en.wikipedia.org/wiki/Android_operating_system
- [12] Wikipedia - Αλληλεπίδραση ανθρώπου υπολογιστή
https://el.wikipedia.org/wiki/%CE%91%CE%BB%CE%BB%CE%B7%CE%B%CE%B5%CF%80%CE%AF%CE%B4%CF%81%CE%B1%CF%83%CE%B7_%CE%B1%CE%BD%CE%B8%CF%81%CF%8E%CF%80%CE%BF%CF%85-%CF%85%CF%80%CE%BF%CE%BB%CE%BF%CE%B3%CE%B9%CF%83%CF%84%CE%AE
- [13] Wikipedia – Internet of Things
https://en.wikipedia.org/wiki/Internet_of_Things
- [14] Wikipedia Mobile Operating systems
https://en.wikipedia.org/wiki/Mobile_operating_system#Windows_Mobile
- [15] Wunderbar platform
<http://www.allaboutcircuits.com/projects/an-introduction-to-the-wunderbar/>
- [16] Ακολουθιακό διάγραμμα Command Pattern
<http://richard.jp.leguen.ca/images/soen343-f2010/TheCommandPattern/command-pattern-sequence-diagram.jpg>
- [17] Online offline recognition
<http://www.zdnet.com/article/always-on-google-ai-gives-android-voice-recognition-that-works-on-or-offline/>
- [18] Speech Recognizer
<https://developer.android.com/reference/android/speech/SpeechRecognizer.html>

- [19] RecognitionListener
<https://developer.android.com/reference/android/speech/RecognitionListener.html#onBufferReceived%28byte%5B%29>

- [20] Smart Things Architecture
<http://docs.smarthings.com/en/latest/architecture/>

- [21] Smart Things installation flow
<http://docs.smarthings.com/en/latest/smartapp-web-services-developers-guide/overview.html>

- [22] Εφαρμογή Control4
<https://play.google.com/store/apps/details?id=com.control4.myhome>

- [23] Εφαρμογή Archos
<https://play.google.com/store/apps/details?id=com.archos.athome.center>

- [24] Εφαρμογή PLC Smart Home
<https://play.google.com/store/apps/details?id=de.in4matics.iHomeControl&hl=en>

Παράρτημα Α

1.1 Πηγαίος Κώδικας Smart Things

A-1

1.1 Πηγαίος Κώδικας Smart Things

Σε αυτό το παράρτημα έχει συμπεριληφθεί ο κώδικας για την πλατφόρμα Smart Things. Ο κώδικας είναι από την ιστοσελίδα <https://community.smarthings.com/t/web-services-smartapp-for-multiple-switches-but-only-control-one/18178> και έχει τροποποιηθεί για τις ανάγκες της εφαρμογής

```
/**
 * Web Services Tutorial
 *
 * Copyright 2016 Smarththings
 *
 * Licensed under the Apache License, Version 2.0 (the "License"); you may not use
this file except
 * in compliance with the License. You may obtain a copy of the License at:
 *
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software distributed under
the License is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY
KIND, either express or implied. See the License
 * for the specific language governing permissions and limitations under the License.
```

*

*/

definition(

name: "Web Services Tutorial",

namespace: "smarththings",

author: "Smarththings",

description: "web services tutorial",

category: "",

iconUrl: "https://s3.amazonaws.com/smartapp-icons/Convenience/Cat-
Convenience.png",

iconX2Url: "https://s3.amazonaws.com/smartapp-icons/Convenience/Cat-
Convenience@2x.png",

iconX3Url: "https://s3.amazonaws.com/smartapp-icons/Convenience/Cat-
Convenience@2x.png",

oauth: [displayName: "web services tutorial ", displayLink: "http://localhost:4567"])

/*

Nullable name -> device needed

if you put multiple false and

required false simulator is not working

**/

preferences {

section("Devices"){

input "switches", "capability.switch",multiple: true ,required:true

input "backDoor", "capability.lock" ,multiple: true ,required:true

input "CentralLockingSystem", "capability.lock" ,multiple: true ,required:true

input "Alarm", "capability.alarm" ,multiple: true ,required:true

input "living", "capability.switch",multiple: true ,required:true

input "bedroom", "capability.switch",multiple: true ,required:true

```

input "boiler", "capability.switch",multiple: false ,required:true
input "coffeeMaker", "capability.switch",multiple: false ,required:true
input "bathroom", "capability.switch",multiple: true ,required:true
input "toastMaker", "capability.switch",multiple: true ,required:true
input "kitchen", "capability.switch",multiple: false ,required:true
input "garageGate", "capability.switch",multiple: true ,required:true
input "familyNotifier", "capability.switch",multiple: false ,required:true

```

```

}}

```

```

mappings {

```

```

    path("/switches") {
        action: [
            GET: "listSwitches"
        ]
    }

```

```

    path("/switches/:command/:command1") {
        action: [
            PUT: "updateSwitches"
        ]
    }

```

```

/*

```

```

path("/dr") {
    action: [
        GET: "listdr"
    ]
}

```

```

path("/dr/:command") {
    action: [

```

```
    PUT: "updatedr"  
  ]  
}
```

```
path("/gar") {  
  action: [  
    GET: "getgar",  
    PUT: "putgar"  
  ]  
}  
*/
```

```
path("/lock") {  
  action: [  
    GET: "listDoor"  
  ]  
}
```

```
path("/lock/:cmd1/:cmd2") {  
  action: [  
    PUT: "updategarageDoor"  
  ]  
}
```

```
path("/alarm"){  
  
  action:[GET:"list_win"]}  
  
  path("/alarm/:command"){action:[PUT:"updateShade"]}  
  
}
```

```

def listDoor() {

    def resp = []
    lock.each {
        resp << [name: it.displayName, value: it.currentValue("lock")]
    }
    return resp
}

void updateSwitches() {
    // use the built-in request object to get the command parameter
    def command = params.command
    def command2 = params.command1
    //log.debug "co param: $command"
    //log.debug "co pm: $command2"
    def devicesMan; /*
    List<String>                                Mydevices=new
    ArrayList<String>();Mydevices.add("switches");Mydevices.add("living");

    int i;
    for(String elementI:Mydevices)
    if(elementI==command2){
    devicesMan=elementI;
    break;
    }*/

    if(command2=="switches")
    devicesMan=switches

    if(command2=="living")
    devicesMan=living

    if(command2=="bedroom")

```

```
devicesMan=bedroom
```

```
if(command2=="bathroom")
```

```
devicesMan=bathroom
```

```
if(command2=="toastMaker")
```

```
devicesMan=toastMaker
```

```
if(command2=="boiler")
```

```
devicesMan=boiler
```

```
if(command2=="coffeeMaker")
```

```
devicesMan=coffeeMaker
```

```
if(command2=="kitchen")
```

```
devicesMan=kitchen
```

```
else
```

```
if(command2=="familyNotifier")
```

```
devicesMan=familyNotifier
```

```
if(command2=="garageGate")
```

```
devicesMan=garageGate
```

```
if(command2=="bedroom")
```

```
devicesMan=bedroom
```

```
if(command2=="bathroom")
```

```
devicesMan=bathroom
```

```
if (command) {
```

```
    // check that the switch supports the specified command
```

```
    // If not, return an error using httpError, providing a HTTP status code.
```

```

    devicesMan.each {
        if (!it.hasCommand(command)) {
            httpError(501, "$command is not a valid command for all switches specified")
        }
    }

    // all switches have the comand
    // execute the command on all switches
    // (note we can do this on the array - the command will be invoked on every
element
    devicesMan."$command"()
}
}
// returns a list like
// [[name: "kitchen lamp", value: "off"], [name: "bathroom", value: "on"]]
def listSwitches() {

    def resp = []
    switches.each {
        resp << [name: it.displayName, value: it.currentValue("switch")]
    }
    return resp
}

void updategarageDoor() {
    // use the built-in request object to get the command parameter
    def command = params.cmd1
    def command2 = params.cmd2
    def deviceManager2;

    if(command2=="backDoor")
    deviceManager2=backDoor

```

```

if(command2=="CentralLockingSystem")
deviceManager2=CentralLockingSystem

if (command) {

    // check that the switch supports the specified command
    // If not, return an error using httpError, providing a HTTP status code.
    deviceManager2.each {
        if (!it.hasCommand(command)) {
            httpError(501, "$command is not a valid command for all switches specified")
        }
    }

    // all switches have the comand
    // execute the command on all switches
    // (note we can do this on the array - the command will be invoked on every
element
    deviceManager2."$command"()
}

}

```

```

def list_win() {

    def resp = []
    win_shade.each {
        resp << [name: it.displayName, value: it.currentValue("alarm")]
    }
}

```

```

    }
    return resp
}

void updateShade() {
    // use the built-in request object to get the command parameter
    def command = params.command

    if (command) {

        // check that the switch supports the specified command
        // If not, return an error using httpError, providing a HTTP status code.
        Alarm.each {
            if (!it.hasCommand(command)) {
                httpError(501, "$command is not a valid command for all switches specified")
            }
        }

        // all switches have the comand
        // execute the command on all switches
        // (note we can do this on the array - the command will be invoked on every
element
        Alarm."$command"()
    }
}

/*

def listdr() {

    def resp = []
    dr.each {

```

```

    resp << [name: it.displayName, value: it.currentValue("doorControl")]

  }
  return resp
}

void updatedr() {
  // use the built-in request object to get the command parameter
  def command = params.command

  if (command) {

    // check that the switch supports the specified command
    // If not, return an error using httpError, providing a HTTP status code.
    dr.each {
      if (!it.hasCommand(command)) {
        httpError(501, "$command is not a valid command for all switches specified")
      }
    }

    // all switches have the comand
    // execute the command on all switches
    // (note we can do this on the array - the command will be invoked on every
    element
    dr."$command"()
  }
}
*/
def installed() {}

def updated() {}

```

Παράρτημα Β Πηγαίος Κώδικας Android

1.1	Αρχείο Manifest	A-1
1.2	Διεπαφές	A-3
1.3	Πηγαίος Κώδικας για Activities	A-5
1.4	Πηγαίος Κώδικας για Κλάσεις	A-23
1.5	Πηγαίος Κώδικας για Layouts	A-47

Σε αυτό το παράρτημα έχει συμπεριληφθεί ο κώδικας για την εφαρμογή αλλά ο οποίος περιλαμβάνει και τον κώδικα για την πλατφόρμα WunderBar . Για την εκτέλεση του κώδικα έχει χρησιμοποιηθεί η βιβλιοθήκη της Java για το relayr – wunderbar

1.1 Αρχείο Manifest

Android Manifest

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.charis.hrc" >

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.RECORD_AUDIO" />
    <uses-permission android:name="android.permission.VIBRATE" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/logo"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
```

```
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
<activity
    android:name=".TaskManager"
    android:label="@string/title_activity_task_manager" >
</activity>
<activity
    android:name=".Platform_chooser"
    android:label="@string/title_activity_platform_chooser" >
</activity>
<activity
    android:name=".Scenarios"
    android:label="@string/title_activity_scenarios" >
</activity>
</application>

</manifest>
```

1.2 Πηγαίος Κώδικάς για Διεπαφές

Interface Command

```
package com.example.charis.hrc;

/**
 * Created by Charis on 06-Nov-15.
 */
public interface Command {
    String error = "An error has been occurred!";

    String execute();

    // void response();
}
```

Smart Things Interface

```
package com.example.charis.hrc;

/**
 * @author Charis Charalambous
 *
 * @description This interface provides the necessary data for authorization
 to Smart Things simulator
 */
public interface Smart_Things_Interface extends VoiceResponses {

    String TOKEN = "Bearer 9ef12702-ac8b-4670-a043-42f32369afb5";
    String SMART_THINGS_URL =
"https://graph.api.smarthings.com/api/smartapps/installations/3ecb9a5d-3137-4216-b25d-61f209c736b5";
    //https://graph.api.smarthings.com/api/smartapps/installations/57c63085-580a-4d16-b4d1-5c703b6d672e";
    int GET_OK = 200;
    int PUT_OK = 204;
    String AUTHORIZATION = "Authorization";

    enum LIGHTS {LIVING_ROOM, KITCHEN_ROOM}

    enum DEVICES {SWITCHES, LOCK, ALARM}

    String getAuthorization();

    int makeRESTRequest(String action);

    String getResponseMessage();
    // ResponseStatus initializeResponseStatus();
}
```

ResponseStatus Interface

```
package com.example.charis.hrc;

/**
 * Created by Charis on 29-Feb-16.
 * <p/>
 * <p/>
 * <p/>
 * This interface is used in order to get the response code from a thread in
 * order to make corresponding actions
 */
public interface ResponseStatus {

    void get_Response_Code(int code);

}
```

1.3 Πηγάιος Κώδικας για Activities

MainActivity

```
package com.example.charis.hrc;

import android.app.Activity;
//import android.support.v7.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.os.Handler;
public class MainActivity extends Activity {

    public void
onResume() {super.onResume(); //if(getIntent().getExtras().getInt("exit")==0) {fi
nish(); System.exit(0); }

        Handler delay=new Handler();
        delay.postDelayed(new Runnable() {
            @Override
            public void run() {
                Intent i;
                i = new Intent(MainActivity.this, Platform_chooser.class);
                startActivity(i);
            }
        }, 3000);}

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    Handler delay=new Handler();
    delay.postDelayed(new Runnable()
    {
        @Override
        public void run() {
            Intent i;
            i = new Intent(MainActivity.this, Platform_chooser.class);
            startActivity(i);
        }
    }, 3000);
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    // Inflate the menu; this adds items to the action bar if it is
present.
    getMenuInflater().inflate(R.menu.menu_main, menu);
    return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {
    // Handle action bar item clicks here. The action bar will
    // automatically handle clicks on the Home/Up button, so long
    // as you specify a parent activity in AndroidManifest.xml.
    int id = item.getItemId();

    //noinspection SimplifiableIfStatement
    if (id == R.id.action_settings) {
```

```
        return true;
    }

    return super.onOptionsItemSelected(item);
}
}
```

Platform_chooser Activity

```
package com.example.charis.hrc;

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.Toast;

public class Platform_chooser extends AppCompatActivity {
    CheckBox smartThingsPlatform;
    CheckBox wunderbarPlatform;
    Intent next;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_platfrom_chooser);
        smartThingsPlatform = (CheckBox)
findViewById(R.id.smartThingsPlatform);
        wunderbarPlatform = (CheckBox) findViewById(R.id.wunderbarPlatform);

        Button chosenPlatforms = (Button) findViewById(R.id.chosenPlatforms);

        chosenPlatforms.setOnClickListener(new View.OnClickListener() {
            /**
             * Called when a view has been clicked.
             *
             * @param v The view that was clicked.
             */
            @Override
            public void onClick(View v) {

                next = new Intent(Platform_chooser.this, TaskManager.class);
                next.putExtra("exit",1);
                if (smartThingsPlatform.isChecked())
                    next.putExtra("smartThingsPlatform", true);
                else next.putExtra("smartThingsPlatform", false);
                if (wunderbarPlatform.isChecked())
                    next.putExtra("wunderBarPlatform", true);
                else
                    next.putExtra("wunderBarPlatform", false);

                startActivity(next);
            }
        });
    }

    // public void onResume(){super.onResume();
    //     boolean a=next.getExtras().getBoolean("smartThingsPlatform");
    //     Toast.makeText(getApplicationContext(), String.valueOf(a),
    Toast.LENGTH_SHORT).show();
    // }
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is
        present.
        getMenuInflater().inflate(R.menu.menu_platfrom_chooser, menu);
    }
}
```

```

        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();

        //noinspection SimplifiableIfStatement
        if (id == R.id.action_settings) {
            return true;
        }

        return super.onOptionsItemSelected(item);
    }
}

```

TaskManager Activity

```
package com.example.charis.hrc;

import android.animation.Animator;
import android.animation.ObjectAnimator;

import android.app.Activity;
import android.content.Context;
import android.content.Intent;

import android.os.Bundle;
import android.os.Vibrator;
import android.speech.RecognitionListener;
import android.speech.RecognizerIntent;
import android.speech.SpeechRecognizer;
import android.speech.tts.TextToSpeech;
import android.util.Log;

import android.view.Menu;
import android.view.MenuItem;
import android.view.MotionEvent;
import android.view.View;
import android.view.animation.DecelerateInterpolator;
import android.widget.Button;
import android.widget.CompoundButton;

import android.widget.ImageButton;

import android.widget.ProgressBar;
import android.widget.Switch;
import android.widget.TextView;
import android.widget.Toast;

import java.util.ArrayList;
import java.util.Locale;

public class TaskManager extends Activity implements RecognitionListener,
TextToSpeech.OnInitListener, Smart_Things_Interface {
    private String Appear = "Appear";
    private static EnvironmentMeasures measures;
    private static VoiceCommands vc;
    private TextView recorderWords;
    private ImageButton imageButton;
    private ImageButton temperature;
    private ProgressBar progressBar;
    private Intent recognizer;
    private TextView instruction;
    private SpeechRecognizer speaker;
    private ObjectAnimator progress;
    private Button speechButton;
    private TextToSpeech translator;
    private String textCommand = " easy with rotation!";
    private VoiceCommands availableCommands;
    private RemoteControl remoteCon;
    private Light light;
    Command flipUp;
    Command flipDown;
    private WunderBarConnection wundercon;
    private WakeUpScenario wake;
    private ImageButton arrayButton[] = new ImageButton[17];
    private boolean pressed;
    private ImageButton driverMode;
    private Button voicecmdssettings;
```

```

        private Button settingsback;
        private boolean driverRequest;
        static int rotateCounter = 1;
private ImageButton temperatureButton;
        private Switch switcherDriver;
        private Switch drivingSwitch;
        private boolean smartThingsPlatform;
        private boolean wunderbarPlatform;
        private Button infoback;
        VoiceListener listener = new VoiceListener();
        MainListener gotoMain = new MainListener();

        private Button measuresBoardback;
        private Button voiceCommandsback;

        class MainListener implements View.OnClickListener {
            /**
             * Called when a view has been clicked.
             *
             * @param v The view that was clicked.
             */
            @Override
            public void onClick(View v) {
                setContentView(R.layout.activity_task_manager);
                finish();
                startActivity(getIntent());
            }
        }

        class VoiceListener implements View.OnClickListener {
            /**
             * Called when a view has been clicked.
             *
             * @param v The view that was clicked.
             */
            @Override
            public void onClick(View v) {
                {
                    pressed = !pressed;
                    if (pressed) {
                        progress.start();
                        instruction.setText("");

                        progressBar.setVisibility(View.VISIBLE);
                        progressBar.setIndeterminate(true);
                        speaker.startListening(recognizer);
                    } else {
                        instruction.setText("Click image to speak");
                        progressBar.setIndeterminate(false);
                        progressBar.setVisibility(View.INVISIBLE);
                        speaker.stopListening();
                    }
                }
            }
        }

        /**
         * This function initializes classes which will be used for creating the
         smart things devices
         * and for voice interaction
         * i.e. lamps of the house, translator ,recognizer
         */

```

```

void initializeClasses() {
    light = new Light(false);
    wake = new WakeUpScenario(null, measures,
7);////////////////////////////////////
    remoteCon = new RemoteControl();
    flipUp = new FlipUpCommand(light);
    flipDown = new FlipDownCommand(light);

    translator = new TextToSpeech(getApplicationContext(), this);
    recognizer = new Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
    availableCommands = new VoiceCommands();

}

void initializeWunderBarModule() {
    measures = new EnvironmentMeasures();
    wundercon = new WunderBarConnection(measures);
}

/**
 * This function initializes the elements for using them in graphical
 * interface and manipulate
 * them for some actions
 */

void initializeComponents() {
    recorderWords = (TextView) findViewById(R.id.recordedWords);
    progressBar = (ProgressBar) findViewById(R.id.progressBar);
    imageButton = (ImageButton) findViewById(R.id.imageButton);
    instruction = (TextView) findViewById(R.id.instruction1);
    temperatureButton= (ImageButton) findViewById(R.id.temperatureIcon);
}

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_task_manager);
    initializeClasses();

    smartThingsPlatform =
getIntent().getExtras().getBoolean("smartThingsPlatform");
    wunderbarPlatform =
getIntent().getExtras().getBoolean("wunderBarPlatform");
    initializeComponents();
    vc = null;

//      ObjectInputStream fileContent;
//      FileInputStream voiceCommandsFile;
//      try {
//          voiceCommandsFile=openFileInput("voiceCommands.ser");
//fileContent=new ObjectInputStream(voiceCommandsFile);
//          vc=(VoiceCommands) fileContent.readObject();
//
//          fileContent.close();
//          voiceCommandsFile.close();
//
//      } catch (FileNotFoundException e) {
//          try {
//              vc=new VoiceCommands();
//              FileOutputStream
newFile=openFileOutput("voiceCommands.ser",Context.MODE_PRIVATE);
//

```

```

//
//         } catch (FileNotFoundException e1) {
//             e1.printStackTrace();
//         }
//
//         //     e.printStackTrace();
//     }catch (IOException e) {
//         //::: TODO
//     }
//
//     } catch (ClassNotFoundException e) {
//         e.printStackTrace();
//     }
//
//temperature=(ImageButton)
//drivingSettings.setOnCheckedChangeListener(new
CompoundButton.OnCheckedChangeListener() {
//    @Override
//    public void onCheckedChanged(CompoundButton buttonView, boolean
isChecked) {
//        if(isChecked){      Toast.makeText(getApplicationContext(),
textCommand, Toast.LENGTH_SHORT).show();
//            driverRequest=true;}
//        else
//            driverRequest=false;
//    }
//});
//
//    drivingMode.setOnClickListener(listener);
//    settingsback.setOnClickListener(new View.OnClickListener() {
//        /**
//         * Called when a view has been clicked.
//         *
//         * @param v The view that was clicked.
//         */
//        @Override
//        public void onClick(View v) {
//            settingsLayout.setVisibility(View.VISIBLE);
//
//            Toast.makeText(getApplicationContext(), "here",
Toast.LENGTH_SHORT).show();
//            layoutSwitcher.setDisplayedChild(5);
//            modeset.setVisibility(View.INVISIBLE);
//            voiceset.setVisibility(View.INVISIBLE);
//            setsLayout.setVisibility(View.INVISIBLE);
//            settingsback.setVisibility(View.INVISIBLE);
//            drivingSettings.setVisibility(View.INVISIBLE);
//            layoutSwitcher.setVisibility(View.VISIBLE);
//
//
//
//        }
//    });
//
//    findViewById(R.id.goingout).setOnClickListener(new
View.OnClickListener() {
//        @Override
//        public void onClick(View v) {
//            finish();
//            Intent a=new Intent(TaskManager.this,MainActivity.class);
//            a.putExtra("exit",0);
//            startActivity(a);
//
//            System.exit(0);
//
//        }
//    });
//
//    findViewById(R.id.securevehicleIcon).setOnClickListener(new

```

```

View.OnClickListener() {
    @Override
    public void onClick(View v) {
    }
});

temperatureButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {

        if (!wunderbarPlatform) {

            textCommand = ENABLE_WUNDERBAR;
            speakOut();
            getIntent().putExtra("wunderBarPlatform", true);
            wunderbarPlatform = true;
        }
        if (measures == null || wundercon == null) {
            initializeWunderBarModule();
            try {
                Thread.sleep(7000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }

        setContentView(R.layout.measuresboard);
        wundercon.getMeasures();
        if (measuresBoardback == null) {

            measuresBoardback = (Button)
findViewById(R.id.measuresBoardback);
            measuresBoardback.setOnClickListener(gotoMain);

        }
        TextView a = (TextView) findViewById(R.id.temperaturehum);
        try {
            Thread.sleep(1000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        a.setText(String.valueOf(measures.getTemperature()) + "
\u2103");

        TextView b;
        b = (TextView) findViewById(R.id.wunderbarhumidity);
        b.setText(String.valueOf(measures.getHumidity()) + " %");

    }
});

findViewById(R.id.settingsIcon).setOnClickListener(new
View.OnClickListener() {
    @Override
    public void onClick(View v) {
        setContentView(R.layout.settings);
        if (drivingSwitch == null) {

            drivingSwitch = (Switch) findViewById(R.id.drivingSwitch);
            drivingSwitch.setOnCheckedChangeListener(new
CompoundButton.OnCheckedChangeListener() {
                @Override
                public void onCheckedChanged(CompoundButton
buttonView, boolean isChecked) {
                    driverRequest = isChecked;
                }
            }

```

```

    });
    settingsback = (Button) findViewById(
        R.id.settingsback);
    settingsback.setOnClickListener(new View.OnClickListener()
    {
        @Override
        public void onClick(View v) {
            if (driverRequest) {
                setContentView(R.layout.drivingmode);
                textCommand = DRIVING_MODE;
                speakOut();
                if (driverMode == null) {

                    driverMode = (ImageButton)
findViewById(R.id.driverMode);
                    driverMode.setOnClickListener(listener);
                    switcherDriver = (Switch)
findViewById(R.id.switcher);

                    switcherDriver.setOnCheckedChangeListener(new
CompoundButton.OnCheckedChangeListener() {
                        @Override
                        public void
onCheckedChanged(CompoundButton buttonView, boolean isChecked) {
                            if (isChecked) {
                                finish();
                                startActivity(getIntent());
                            }
                        }
                    });
                    driverMode.setOnTouchListener(new
View.OnTouchListener() {
                        @Override
                        public boolean onTouch(View v,
MotionEvent event) {
                            {
                                Vibrator t = (Vibrator)
getSystemService(Context.VIBRATOR_SERVICE);
                                t.vibrate(250);
                                return false;
                            }
                        }
                    });
                } else //high loading refresh
                {
                    finish();
                    startActivity(getIntent());
                } // onCreate(null); //
setContentView(R.layout.activity_task_manager);
            }
        });

        voicecmdssettings = (Button) findViewById(R.id.voicecmds);
        voicecmdssettings.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                setContentView(R.layout.voicecommandsettings);
                if (voiceCommandsback == null) {
                    voiceCommandsback = (Button)
findViewById(R.id.voiceCommandsback);
                    voiceCommandsback.setOnClickListener(gotoMain);

```

```

    }
    });
}

});
findViewById(R.id.lampIcon).
    setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            //if(enabledSmartThings())
            Smart_Things_Connection.currentDevice =
DEVICES.SWITCHES;

            Smart_Things_Connection.atROOM = LIGHTS.KITCHEN_ROOM;
            if (light.kitchen_room_lightIsOn)
                textCommand = remoteCon.storeAndExecute(flipDown);
            else
                textCommand = remoteCon.storeAndExecute(flipUp);
            speakOut();

        }
    });
findViewById(R.id.infoIcon).setOnClickListener(new
View.OnClickListener() {
    @Override
    public void onClick(View v) {
        setContentView(R.layout.info);
        if (infoback == null) {
            infoback = (Button) findViewById(R.id.backInfo);
            infoback.setOnClickListener(gotoMain);
        }
    }
});
speechButton = (Button) findViewById(R.id.speechbutton);
speechButton
    .setOnClickListener(new View.OnClickListener() {
        /**
         * Called when a view has been clicked.
         *
         * @param v The view that was clicked.
         */
        @Override
        public void onClick(View v) {
            speakOut();
            Toast.makeText(getApplicationContext(), textCommand,
Toast.LENGTH_SHORT).show();

        }
    });

imageButton.setOnClickListener(new View.OnTouchListener() {
    @Override
    public boolean onTouch(View v, MotionEvent event) {
        Vibrator t = (Vibrator)
getSystemService(Context.VIBRATOR_SERVICE);
        t.vibrate(250);
        return false;
    }
});

progressBar.setVisibility(View.INVISIBLE);
speaker = SpeechRecognizer.createSpeechRecognizer(this);

```

```

        speaker.setRecognitionListener(this);
        progress = ObjectAnimator.ofInt(progressBar, "progress", 0, 100);
        progress.setDuration(3000);
        progress.setInterpolator(new DecelerateInterpolator());
        progress.addListener(new Animator.AnimatorListener() {
            @Override
            public void onAnimationStart(Animator animation) {

            }

            @Override
            public void onAnimationEnd(Animator animation) {
                imageButton.callOnClick();
            }

            @Override
            public void onAnimationCancel(Animator animation) {

            }

            @Override
            public void onAnimationRepeat(Animator animation) {

            }
        });
        recognizer.putExtra(RecognizerIntent.EXTRA_LANGUAGE_PREFERENCE, "en-
us");
        recognizer.putExtra(RecognizerIntent.EXTRA_CALLING_PACKAGE,
this.getPackageName());
        imageButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                pressed = !pressed;
                if (pressed) {
                    progress.start();
                    instruction.setText("");

                    progressBar.setVisibility(View.VISIBLE);
                    progressBar.setIndeterminate(true);
                    speaker.startListening(recognizer);
                } else {
                    instruction.setText("Click image to speak");
                    progressBar.setIndeterminate(false);
                    progressBar.setVisibility(View.INVISIBLE);
                    speaker.stopListening();
                }
            }
        });
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is
present.
        getMenuInflater().inflate(R.menu.menu_task_manager, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long

```

```

        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();

        //noinspection SimplifiableIfStatement
        if (id == R.id.action_settings) {
            return true;
        }

        return super.onOptionsItemSelected(item);
    }

    @Override
    public void onReadyForSpeech(Bundle params) {

    }

    @Override
    public void onBeginningOfSpeech() {
        progressBar.setIndeterminate(false);

        progressBar.setMax(10);
    }

    @Override
    public void onRmsChanged(float rmsdB) {
        progressBar.setProgress((int) rmsdB);
    }

    @Override
    public void onBufferReceived(byte[] buffer) {
        Log.i("Appear", "+" + buffer);
    }

    @Override
    public void onEndOfSpeech() {
        progressBar.setIndeterminate(true);
    }

    @Override
    public void onError(int error) {
        switch (error) {

            case SpeechRecognizer.ERROR_AUDIO: {
                recorderWords.setText("Error audio. Please speak again more clearly.");
                break;
            }
            case SpeechRecognizer.ERROR_CLIENT: {
                if (textCommand.compareTo(" ") == 0) {
                    Toast.makeText(getApplicationContext(), "Error on client",
                        Toast.LENGTH_SHORT).show();
                    recorderWords.setText(" ");
                }
                // recorderWords.setText("Error on client");
                break;
            }
            case SpeechRecognizer.ERROR_INSUFFICIENT_PERMISSIONS: {
                recorderWords.setText("Permission denied");
                break;
            }
            case SpeechRecognizer.ERROR_NETWORK: {
                recorderWords.setText("Network related error");
                break;
            }
            case SpeechRecognizer.ERROR_NETWORK_TIMEOUT: {
                recorderWords.setText("Network time out. Please try again");
                break;
            }
        }
    }

```

```

        }
        case SpeechRecognizer.ERROR_NO_MATCH: {
            recorderWords.setText("Can't find recognised words");
            break;
        }
        case SpeechRecognizer.ERROR_RECOGNIZER_BUSY: {
            recorderWords.setText("Speech recognition service is
busy.Please try again");
            break;
        }
        case SpeechRecognizer.ERROR_SPEECH_TIMEOUT: {
            recorderWords.setText("No input speech ");
            break;
        }
    }

}

}

@Override
public void onResults(Bundle results) {
    ArrayList<String> result =
results.getStringArrayList(SpeechRecognizer.RESULTS_RECOGNITION);
    textCommand = result.get(0);
    recorderWords.setText(result.get(0));
    if (availableCommands.findCommand(textCommand)) {
        if (!textCommand.contains(VoiceResponses.CONFIRM))
            availableCommands.previousCommand = textCommand;
        else if (!smartThingsPlatform) {
            smartThingsPlatform = true;
            getIntent().putExtra("smartThingsPlatform", true);
            textCommand = availableCommands.previousCommand;
        }
        if (smartThingsPlatform) {
            if (textCommand.contains("lights on")) {
                Smart_Things_Connection.currentDevice = DEVICES.SWITCHES;
                if (textCommand.contains("living room"))
                    Smart_Things_Connection.atROOM = LIGHTS.LIVING_ROOM;
                else
                    Smart_Things_Connection.atROOM = LIGHTS.KITCHEN_ROOM;
                textCommand = remoteCon.storeAndExecute(flipUp);
                recorderWords.setText(" ");
            }
            // else

            if (textCommand.contains("lights off")) {
                Smart_Things_Connection.currentDevice = DEVICES.SWITCHES;
                if (textCommand.contains("living room"))
                    Smart_Things_Connection.atROOM = LIGHTS.LIVING_ROOM;
                else
                    Smart_Things_Connection.atROOM = LIGHTS.KITCHEN_ROOM;
                textCommand = remoteCon.storeAndExecute(flipDown);
                recorderWords.setText(" ");
            }
            if (textCommand.contains("wake")) {

                textCommand = wake.execute();
            }
        }
    }
    if(textCommand.contains("temperature") || textCommand.contains("humidity")) {temp
eratureButton.setPressed(true);

temperatureButton.invalidate();temperatureButton.performClick();//temperatureB
utton.setPressed(false);

```

```

        // temperatureButton.invalidate();
    }
    } else
        textCommand = ENABLE_SMART_THINGS;
    }
    speakOut();
}

//:::TODO

@Override
public void onPartialResults(Bundle partialResults) {

}

@Override
public void onEvent(int eventType, Bundle params) {

}

/**
 * Called to signal the completion of the TextToSpeech engine
 * initialization.
 *
 * @param status {@link TextToSpeech#SUCCESS} or {@link
 * TextToSpeech#ERROR}.
 */
@Override
public void onInit(int status) {

    if (status == TextToSpeech.SUCCESS) {
        translator.setLanguage(Locale.UK);
        translator.speak(textCommand, TextToSpeech.QUEUE_FLUSH, null);
        speakOut();
    }

}

public void onDestroy() {

    super.onDestroy();

}

public void speakOut() {

    if (textCommand.contains("easy")) {
        if (rotateCounter % 2 == 0) {
            if (generatedNumber() == 1)
                textCommand = GET_DIZZY_2;
            else
                textCommand = GET_DIZZY;
        }
        rotateCounter++;
    }
    translator.speak(textCommand, TextToSpeech.QUEUE_FLUSH, null);
}

@Override
public String getAuthorization() {
    return null;
}

```

```
@Override
public int makeRESTRequest(String action) {
    return 0;
}

@Override
public String getResponseMessage() {
    return null;
}

public int generatedNumber() {
    int temp = 1 + (int) (Math.random() * 2);
    return temp;
}
}
```

Scenarios Activity

```
package com.example.charis.hrc;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.TimePicker;
import android.widget.ViewFlipper;

/**
 * This activity represents the settings for the scenarios of the application
 * <p/>
 * The user can select a scenario and change some settings according to his
 preferences
 * i.e If he wants to change the alarm time He must choose the wake up
 scenario and then
 * find the appropriate command in the new list which it will appear after he
 tap the scenario.
 */
public class Scenarios extends AppCompatActivity {

    private ViewFlipper scenarioChooser;
    private Button wakeUpScenario;
    private TextView checkTemperature;
    private TimePicker timePicker;

    /**
     * Description This function initialize the components which will be used
 during this
     * activity. The components are all the items which are included in the
 wakeupsenario.xml file
     *
     * @param none
     * @return void
     */
    private void initializeComponents() {
        scenarioChooser = (ViewFlipper) findViewById(R.id.viewFlipper);
        wakeUpScenario = (Button) findViewById(R.id.wakeUpScenarioButton);
        checkTemperature = (TextView) findViewById(R.id.checkWaterTempText);
        timePicker = (TimePicker) findViewById(R.id.timePicker);
    }

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

        initializeComponents();

        wakeUpScenario.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                wakeUpScenario.setVisibility(View.INVISIBLE);
                scenarioChooser.setDisplayedChild(0);
            }
        });
    }
}
```

```

        checkTemperature.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {

                timePicker.setVisibility(View.VISIBLE);

            }
        });

    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is
present.
        getMenuInflater().inflate(R.menu.menu_scenarios, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();

        //noinspection SimplifiableIfStatement
        if (id == R.id.action_settings) {
            return true;
        }

        return super.onOptionsItemSelected(item);
    }
}

```

1.4 Πηγαίος Κώδικας για Κλάσεις

VoiceResponses Class

```
package com.example.charis.hrc;

/**
 * Created by Charis on 29-Feb-16.
 */
public interface VoiceResponses {

    enum VRESPONSE{LIGHT1,LIGHT2,LOCK1,LOCK2}

    String GET_DIZZY="easy with rotation";
    String GET_DIZZY_2="I getting dizzy";
    String LIGHTS_ON="Much brighter now";
    String LIGHTS_ON2="Finally i couldn't see my nose in here";
    String DRIVING_MODE= "Keep your eyes on the road and I'll keep mine on the house";
    String LOCK="Doors and winodws are locked!!!";
    String LOCK1="Your home is locked!!";
    String CONFIRM="activate it please";
    String WAKEUPSCENARIO="Mr. Smith I want to inform you that your family notifier has been activated!";
    String ENABLE_SMART_THINGS="You forgot to enable smart Things platform.You can activate in settings screen or I can activate it if you say activate it please!!";
    String ENABLE_WUNDERBAR="It looks like you didn't enable wunderbar platform .I will activate for you";
}
```

EnvironmentMeasures Class

```
package com.example.charis.hrc;

/**
 * This class is used for keeping the current measures
 * i.e. temperature, humidity, lightness etc.
 * <p/>
 * It's very abstract so that it could be used by any platform that has
 * sensors. It contains the getters and setters methods and some variables
 * which represent
 * the measures if different sensors.
 *
 * @author Charalambos Charalambous
 */

@SuppressWarnings("ALL")
public class EnvironmentMeasures {

    // the current values of the corresponded measures
    private static double humidity;
    private static double temperature;

    /**
     * This function sets the current value of the Humidity according to the
     current humidity sensor
     *
     * @param currentHumidity
     */
    public void setHumidity(double currentHumidity) {

        humidity = currentHumidity;

    }

    /**
     * This function gets the current humidity specified by a humidity sensor
     *
     * @return The current humidity
     */
    public double getHumidity() {

        return humidity;

    }

    /**
     * This function sets the current value of the Humidity according to the
     current temperature sensor
     *
     * @param currentTemperature
     */
    public void setTemperature(double currentTemperature) {

        temperature = currentTemperature;

    }

    /**
     * This function gets the current temperature specified by a humidity
     sensor
     *
     * @return The current temperature
     */
    public double getTemperature() {
```

```
        System.out.println(temperature);  
        return temperature;  
    }  
}
```

FlipDownCommand Class

```
package com.example.charis.hrc;

public class FlipDownCommand implements Command {

    private Light light;

    public FlipDownCommand(Light light) {

        this.light = light;

    }

    @Override
    public String execute() {

        return light.turnOff();

    }

    // @Override
    // public void response() {
    //
    //     return ;///null;
    // }
}
```

FlipUpCommand Class

```
package com.example.charis.hrc;

import android.util.Log;

public class FlipUpCommand implements Command {

    private Light light;

    public FlipUpCommand(Light light) {

        this.light = light;

    }

    @Override
    public String execute() {
        Log.d("3", "Requesting on !!!!!!!!!!!!!!!");

        return light.turnOn();
    }
    ///return " ";//response();
    }

    //    @Override
    //    public void response() {
    //
    //        if(1==PUT_OK)
    //            return null;
    //    }
    //    return ;// null;
    //
    //    }
}
```

Light Class

```
package com.example.charis.hrc;

import java.util.Random;

/**
 * Created by Charis on 10-Feb-16.
 */
public class Light extends Smart_Things_Connection implements VoiceResponses {

    static boolean living_room_lightIsOn = false;
    static boolean kitchen_room_lightIsOn = false;
    static int randomResponse;

    public Light(boolean state) {

        living_room_lightIsOn = state;
        kitchen_room_lightIsOn = state;
    }

    public String turnOn() {

        Thread temp;
        temp = null;

        switch (atROOM) {
            case LIVING_ROOM: {
                if (living_room_lightIsOn)
                    return "Lights are already on";
                temp = new Thread(new RestThread("/switches/on/living", rc));
                break;
            }
            case KITCHEN_ROOM: {
                if (kitchen_room_lightIsOn)
                    return "Lights are already on";
                temp = new Thread(new RestThread("/switches/on/switches",
rc));
                break;
            }
        }

        temp.start();
        while (temp.isAlive()) {
            try {
                Thread.sleep(100);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }

        switch (atROOM) {
            case LIVING_ROOM: {
                living_room_lightIsOn = true;
                break;
            }
            case KITCHEN_ROOM: {
                kitchen_room_lightIsOn = true;
                break;
            }
        }

        return getResponseMessage();
        /// return "Much brighter now!!!";
    }
}
```

```

    }

    public String turnOff() {

        Thread temp;
        temp = null;
        switch (atROOM) {
            case LIVING_ROOM: {
                if (living_room_lightIsOn == false)
                    return "Lights are already off";
                temp = new Thread(new RestThread("/switches/off/living", rc));
                break;
            }
            case KITCHEN_ROOM: {
                if (kitchen_room_lightIsOn == false)
                    return "Lights are already off";
                temp = new Thread(new RestThread("/switches/off/switches",
rc));
                break;
            }
        }

        temp.start();
        while (temp.isAlive()) {
            try {
                Thread.sleep(100);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }

        switch (atROOM) {
            case LIVING_ROOM: {
                living_room_lightIsOn = false;
                break;
            }
            case KITCHEN_ROOM: {
                kitchen_room_lightIsOn = false;
                break;
            }
        }

        //      if(!lightIsOn)
        //          return "Lights are already off";

        //      lightIsOn=false;
        return getResponseMessage();
        /// return "Much brighter now!!!";

    }

    @Override
    public String getResponseMessage() {

        if (kitchen_room_lightIsOn || living_room_lightIsOn
        ) {
            if (status_code == 204) {
                randomResponse = (int) (1 + Math.random() * 2);
                if (randomResponse == 1)
                    return LIGHTS_ON;
                else
                    return LIGHTS_ON2;
            }
        }

        return (status_code == 204) ? " " : "Error occured";
    }

```

```
}  
}
```

RelayrEntities Class

```
package com.example.charis.hrc;

import java.util.List;

import io.relayr.java.model.Device;
import io.relayr.java.model.Transmitter;
import io.relayr.java.model.User;
import io.relayr.java.model.groups.Group;
import rx.Observer;

public class RelayrEntities {

    private User user;

    public RelayrEntities(User user) {
        this.user = user;
    }

    public void start() {
        getGroups();
        getTransmitters();
        getDevices();
    }

    private void getGroups() {
        user.getGroups().subscribe(new Observer<List<Group>>() {
            @Override
            public void onCompleted() {
            }

            @Override
            public void onError(Throwable e) {
                System.err.println("Something went wrong while
fetching groups.");
                e.printStackTrace();
            }

            @Override
            public void onNext(List<Group> groups) {
                System.out.println(groups.size() + " GROUPS");

                for (Group group : groups)
                    System.out.println(group.getName());
            }
        });
    }

    private void getTransmitters() {
        user.getTransmitters().subscribe(new Observer<List<Transmitter>>() {
            @Override
            public void onCompleted() {
            }

            @Override
            public void onError(Throwable e) {
                System.err.println("Something went wrong while
fetching transmitters.");
                e.printStackTrace();
            }

            @Override
            public void onNext(List<Transmitter> transmitters) {
                System.out.println(transmitters.size() + "
```

```

TRANSMITTERS");
        for (Transmitter transmitter : transmitters)
            System.out.println(transmitter.getName());
    }
}

private void getDevices() {
    user.getDevices()
        .subscribe(new Observer<List<Device>>() {
            @Override
            public void onCompleted() {
            }

            @Override
            public void onError(Throwable e) {
                System.err.println("Something went wrong while
fetching devices.");
                e.printStackTrace();
            }

            @Override
            public void onNext(List<Device> devices) {
                System.out.println(devices.size() + " DEVICES");

                for (Device device : devices)
                    System.out.println(device.getName());
            }
        });
}
}

```

RelayrThermometer

```
package com.example.charis.hrc;

import android.util.Log;

import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.Timer;
import java.util.TimerTask;
import java.util.concurrent.TimeUnit;
import java.util.concurrent.TimeoutException;

import io.relayr.java.RelayrJavaSdk;
// import io.relayr.java.RelayrJavaSdk;
import io.relayr.java.model.Device;
//import io.relayr.java.model.Transmitter;
//import io.relayr.java.model.TransmitterDevice;
import io.relayr.java.model.User;
import io.relayr.java.model.action.Reading;
import io.relayr.java.model.models.DeviceFirmware;
import io.relayr.java.model.models.DeviceModel;
import io.relayr.java.model.models.error.DeviceModelsException;
import io.relayr.java.model.models.schema.SchemaType;
import io.relayr.java.model.models.transport.DeviceReading;
import io.relayr.java.model.models.transport.Transport;
import rx.Observable;
import rx.Observer;
import rx.functions.Func1;

/**
 * This purpose of this class is obtaining the current temperature from the
 * wunderbar module through
 * the relayr cloud . The part of the registration to the relayr has been
 * implemented
 * by Kyriakos Petrou. So in this class a generated device id and an api
 * key(in WunderBarConnection class)
 * is used for getting access to the relayr cloud and getting the measure of
 * the temperature and humidity.
 */
public class RelayrThermometer {

    static double temperature;
    static double humidity;
    static int counter = 0;
    // the generated key which is used for a 15-day period
    private static final String DEVICE_ID = "3b2b8e0e-1cb0-4acd-80ef-0c2f3ad9b14e";

    private User user;

    public RelayrThermometer(User user) {
        this.user = user;
    }

    /**
     * The following functions have been created by a project in GitHub.The
     * purpose of this function
     * is getting the current temperature from the relayr cloud server using
     * the subscribe method.
     */
    public void start() {
        //Load user devices
        System.out.println("Start thermometer example.");
    }
}
```

```

        user.getDevices()
            .subscribe(new Observer<List<Device>>() {
                @Override
                public void onCompleted() {
                }

                @Override
                public void onError(Throwable e) {
                    System.out.println("Problem while getting transmitter
data");

                    e.printStackTrace();
                }

                @Override
                public void onNext(List<Device> devices) {
                    System.out.println("Find thermometer");
                    boolean found = false;
                    for (Device device : devices) {
                        //iterate though all user devices and find device
                        //Here you must provide your deviceId
                        if (device.getId().equals(DEVICE_ID)) {
                            found = true;
                            //Simple method
                            subscribeToReadings(device);
                            //Complex method. Shows how to parse any
                            //device data with provided device model
                            //subscribeToReadingsUsingModel(device);
                        }
                    }
                    if (!found) System.out.println("Thermometer NOT
found");
                }
            });
    }

    /**
     * Simple example method for subscribing to cloud data through Device
entity.
     */
    private void subscribeToReadings(final Device device) {
        System.out.println("Subscribe to readings.");

        device.subscribeToCloudReadings()
            .subscribe(new Observer<Reading>() {
                @Override
                public void onCompleted() {
                }

                @Override
                public void onError(Throwable e) {
                    System.out.println("Problem while subscribing for
data");

                    e.printStackTrace();
                    // System.exit(0);
                }

                @Override
                public void onNext(Reading reading) {

                    if (reading.meaning.equals("temperature")) {
                        if (counter % 3600 == 0) {
                            Log.i("Temperature " + reading.value,
"Temperature " + reading.value);
                            temperature = (double) reading.value;
                        }
                    }
                }
            });
    }

```

```

        }
        if (reading.meaning.equals("humidity")) {
            //if(counter%3600==0)
            {
                counter++;
                Log.i("Humidity " + reading.value, "humi " +
reading.value);

                humidity = (double) reading.value;
            }
        }
        // counter++;
    }
});
}

/**
 * Method shows how to use relayr DeviceModel to parse Thermoneter data
 */
private void subscribeToReadingsUsingModel(final Device device) {
    final Map<String, DeviceReading> modelReadings = new HashMap<>();

    //Check if cache is loaded
    //Cache is not loaded by default. It must be set with Builder
argument:
    //RelayrJavaSdk.Builder()...cacheModels(true).build();
    if (RelayrJavaSdk.getDeviceModelsCache().isLoading()) {
        new Timer().schedule(new TimerTask() {
            @Override
            public void run() {
                System.out.printf("Cache loading...");
                subscribeToReadingsUsingModel(device);
            }
        }, 1000);
    } else {
        //Get device model for thermometer sensor and list all possible
readings
        try {
            DeviceModel model =
RelayrJavaSdk.getDeviceModelsCache().getModelByName("Wunderbar Thermometer",
false);

            DeviceFirmware firmware = model.getLatestFirmware();
            Transport transport = firmware.getDefaultTransport();

            Log.i("error", "Device model found");

            //Every device reading defines ValueSchema that describes
value type and other details
            for (DeviceReading reading : transport.getReadings())
                modelReadings.put(reading.getMeaning(), reading);
        } catch (DeviceModelsException e) {
            System.out.printf("Device model not found");
            e.printStackTrace();
        }

        System.out.println("Subscribe to readings using model.");

        device.subscribeToCloudReadings()
            .timeout(10, TimeUnit.SECONDS)
            .subscribe(new Observer<Reading>() {
                @Override
                public void onCompleted() {
                }

                @Override
                public void onError(Throwable e) {

```

```

        if (e instanceof TimeoutException)
            System.out.println("subscribeToCloudReadings -
error.");
        else
            System.out.println("subscribeToCloudReadings -
timeout.");

        e.printStackTrace();
        System.exit(0);
    }

    @Override
    public void onNext(Reading reading) {
        DeviceReading deviceReading =
modelReadings.get(reading.meaning);

        if (deviceReading.getValueSchema().getSchemaType()
== SchemaType.NUMBER)
            if
(deviceReading.getMeaning().equals("temperature"))
                Log.i("temperature", "Temperature " +
reading.value);

            if (deviceReading.getMeaning().equals("humidity"))
                Log.i("humidity", "Humidity " +
reading.value);
        }
    }
}
}
}

```

RemoteControl Class

```
package com.example.charis.hrc;

import java.util.ArrayList;
import java.util.List;

/**
 * Created by Charis on 23-Jan-16.
 */
public class RemoteControl {

    private List<Command> history = new ArrayList<Command>();
    // private WunderBarController wbcontroller=new WunderBarController();

    public String storeAndExecute(Command cmd) {

        this.history.add(cmd);

        return cmd.execute();
    }

    public String showHistory() {

        String str = "";

        for (Command cmd : history)
            str += "\n" + cmd.getClass().getName();

        return str;
    }

    public Command identifyCommand(String cmd) {

        ///return wbcontroller.findCommand(cmd);
        return null;
    }
}
```

RestThread Class

```
package com.example.charis.hrc;

/**
 * Created by Charis on 29-Feb-16.
 */
public class RestThread extends Smart_Things_Connection implements Runnable {

    /**
     * Starts executing the active part of the class' code. This method is
     * called when a thread is started that has been created with a class
     which
     * implements {@code Runnable}.
     */
    private final String get_Request;
    private ResponseStatus rc;

    // private void extractResponseCode(ResponseStatus rc){this.rc=rc;}
    @Override
    public void run() {
        int response_code;
        getAuthorization();
        response_code = makeRESTRequest(get_Request);
        rc.get_Response_Code(response_code);
    }

    public RestThread(String getRequest, ResponseStatus rc) {
        this.rc = rc;
        this.get_Request = getRequest;
    }

    @Override
    public String getResponseMessage() {
        return null;
    }
}
```

Smart Things Connection Class

```
package com.example.charis.hrc;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.URL;

/**
 * Created by Charis on 10-Feb-16.
 */
abstract class Smart_Things_Connection implements Smart_Things_Interface {
    //static String SMART_THINGS_URL ;
    int status_code = 0;
    static LIGHTS atROOM;
    static DEVICES currentDevice;

    public String getAuthorization() {

        URL url = null;

        HttpURLConnection con;//con= null;
        try {
            switch (currentDevice) {
                case SWITCHES: {
                    url = new URL(SMART_THINGS_URL + "/switches");
                    break;
                }
                case ALARM: {
                    url = new URL(SMART_THINGS_URL + "/alarm");
                    break;
                }
                case LOCK: {
                    url = new URL(SMART_THINGS_URL + "/lock");
                    break;
                }
            }

            con = (HttpURLConnection) url.openConnection();
            con.setRequestProperty(AUTHORIZATION, TOKEN);
            if (con.getResponseCode() != GET_OK)
                return "An error has been occurred...Please try again!";

            //        BufferedReader in = new BufferedReader(new
            InputStreamReader(con.getInputStream()));
            //        String decodedString;
            //        while ((decodedString = in.readLine()) != null) {
            //            System.out.println(decodedString);
            //        }
            //        in.close();
        } catch (IOException e) {
            e.printStackTrace();
        }

        return "Much lighter now";

    }

    public synchronized int makeRESTRequest(String action) {

        int rc = 0;
        try {
```

```

        URL url2 = new URL(SMART_THINGS_URL + action);
        HttpURLConnection con2;
        con2 = (HttpURLConnection) url2.openConnection();
        con2.setRequestProperty(AUTHORIZATION, TOKEN);
        con2.setDoOutput(true);
        con2.setDoInput(true);
        con2.setRequestMethod("PUT");
        System.out.println(rc = con2.getResponseCode());

    } catch (IOException e) {
        e.printStackTrace();
    }

    return rc;
}

public ResponseStatus rc = new ResponseStatus() {
    @Override
    public void get_Response_Code(int code) {

        status_code = code;

    }
};
}

```

Voice Commands Class

```
package com.example.charis.hrc;

import android.content.Context;

import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.ObjectInputStream;
import java.io.Serializable;
import java.util.HashMap;
import java.util.HashSet;
import java.util.Map;

/**
 * Created by Charis on 23-Jan-16.
 */
public class VoiceCommands implements Serializable {

    int settings;
    String previousCommand;
    private HashSet<String> cmds=new HashSet<String>();
    private Map<Integer, String> defaultcommands;
    private Map<Integer, String> customcommands;

    public VoiceCommands(){
        cmds.add("lights on");
        cmds.add("living room");
        cmds.add("lights off");
        cmds.add("wake");
        cmds.add("activate it please");
        cmds.add("temperature");

        cmds.add("humidity");

        defaultcommands=new HashMap<Integer,String>();
        defaultcommands.put(1,"");
    }

    public boolean findCommand(String str) {
        return cmds.contains(str);
    }
}
```

WakeUpScenario Class

```
package com.example.charis.hrc;

/**
 * Created by Charis on 25-Mar-16.
 */
public class WakeUpScenario extends Smart_Things_Connection {

    static EnvironmentMeasures measures;
    private static int tempratureThreshold;

    /**
     * This contructor initialize the
     *
     * @param preferredTime
     * @param environmentMeasures contains info for the temperature
     * @param preferredTemperature the user determines a threshold temperature
     for boiler system
     */
    public WakeUpScenario(int preferredTime[], EnvironmentMeasures
environmentMeasures, int preferredTemperature) {

        tempratureThreshold = preferredTemperature;

    }

    /**
     * @return
     */
    public String execute() {

        Thread commands[] = new Thread[20]; //null;
        for (Thread element : commands)
            element = null;
        Smart_Things_Connection.currentDevice = DEVICES.LOCK;
        startThread(new Thread(new RestThread("/lock/lock/backDoor", rc)));

        Smart_Things_Connection.currentDevice = DEVICES.SWITCHES;
        commands[0] = new Thread(new RestThread("/switches/on/boiler", rc));
        Smart_Things_Connection.currentDevice = DEVICES.ALARM;
        commands[1] = new Thread(new RestThread("/alarm/both", rc));

        commands[2] = new Thread(new RestThread("/alarm/off", rc));
        Smart_Things_Connection.currentDevice = DEVICES.SWITCHES;
        commands[3] = new Thread(new RestThread("/switches/on/bedroom", rc));
        commands[4] = new Thread(new RestThread("/switches/on/bathroom", rc));
        commands[5] = new Thread(new RestThread("/switches/on/boiler", rc));
        commands[6] = new Thread(new RestThread("/switches/on/coffeeMaker",
rc));
        commands[7] = new Thread(new RestThread("/switches/on/toastMaker",
rc));
        commands[8] = new Thread(new RestThread("/switches/on/kitchen", rc));
        commands[9] = new Thread(new RestThread("/switches/off/bedroom", rc));
        commands[10] = new Thread(new RestThread("/switches/off/bathroom",
rc));
        commands[11] = new Thread(new RestThread("/switches/off/coffeeMaker",
rc));
        commands[12] = new Thread(new RestThread("/switches/off/toastMaker",
rc));
        commands[13] = new Thread(new RestThread("/switches/off/kitchen",
rc));
        Smart_Things_Connection.currentDevice = DEVICES.LOCK;
        commands[14] = new Thread(new RestThread("/lock/unlock/backDoor",
rc));

        Smart_Things_Connection.currentDevice = DEVICES.SWITCHES;
```

```

        commands[15] = new Thread(new RestThread("/switches/on/garageGate",
rc));

        commands[16] = new Thread(new RestThread("/switches/off/garageGate",
rc));
        Smart_Things_Connection.currentDevice = DEVICES.LOCK;
        commands[17] = new Thread(new
RestThread("/lock/lock/CentralLockingSystem", rc));
        commands[18] = new Thread(new RestThread("/lock/lock/backDoor", rc));
        Smart_Things_Connection.currentDevice = DEVICES.SWITCHES;
        commands[19] = new Thread(new
RestThread("/switches/on/familyNotifier", rc));

        //      commands[9] = new Thread(new RestThread("/switches/on/boiler",
rc));

        for (Thread element : commands)
            startThread(element);

        return VoiceResponses.WAKEUPSCENARIO;
    }

    private void startThread(Thread element) {
        element.start();
        while (element.isAlive()) {
            try {
                Thread.sleep(100);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }

    @Override
    public String getResponseMessage() {
        return null;
    }
}

```

WunderBarConnection Class

```
package com.example.charis.hrc;

import android.util.Log;

import java.util.concurrent.TimeUnit;

import io.relayr.java.RelayrJavaSdk;
import io.relayr.java.model.User;
import rx.Observable;
import rx.functions.Action1;
import rx.functions.Func1;

/**
 * @author Charalambos Charalambous
 * <p/>
 * This class is used for initializing a connection with the relayr
cloud so that I can get the
 * measures from the wunderbar's sensors.
 */
public class WunderBarConnection {
    // global variables which can be used by any module that contains sensors
    static EnvironmentMeasures measures;

    // the following variables represent the sensors of the wunderbar
    private static RelayrThermometer temperature_humidity_sensor;

    /**
     * This constructor initialize the field measures
     *
     * @param measure
     */
    public WunderBarConnection(EnvironmentMeasures measure) {

        measures = measure;

        getTemperatureAndHumidityMeasures();
    }

    /**
     * This function is used for accessing the relayr cloud server using a
token and starting the
     * procedure for getting the the measure for the temperature and the
humidity (because
     * of the same device id the measures can be obtained by changing a
parameter).
     */
    public void getTemperatureAndHumidityMeasures() {

        new RelayrJavaSdk.Builder().setToken("Bearer
1R_457ANNx145Ez6oulq8CrFOqyR4bW0").cacheModels(true).build();

        //Gets user, starts RelayrEntities object to display all user
        entities.
        //After 3 seconds it starts the RelayrThermometer object and
        subscribes for data
        RelayrJavaSdk.getUser()
            .flatMap(new Func1<User, Observable<User>>() {
                @Override
                public Observable<User> call(User user) {
                    new RelayrEntities(user).start();
                    return Observable.just(user);
                }
            })
            .delay(3, TimeUnit.SECONDS)
    }
```

```

        .subscribe(new Action1<User>() {
            @Override
            public void call(User user) {
                temperature_humidity_sensor =
                    new RelayrThermometer(user);
                temperature_humidity_sensor.start();
            }
        }, new Action1<Throwable>() {
            @Override
            public void call(Throwable e) {
                Log.i("error in thermometer", "thermo error");
                e.printStackTrace();
            }
        });
    }

    /**
     * This function sets the values of the environment measures
     * according to corresponded sensor
     *
     * @param
     * @return void
     */
    public void getMeasures() {

        measures.setTemperature(temperature_humidity_sensor.temperature);
        measures.setHumidity(temperature_humidity_sensor.humidity);

    }
}

```

WakeUpSettings Class

```
package com.example.charis.hrc;

/**
 * Created by Charis on 12-Apr-16.
 */
public class WakeUpSettings {

    public WakeUpSettings() {}

}
```

WunderBarController

```
package com.example.charis.hrc;

import java.io.Serializable;
import java.util.Map;

/**
 * Created by Charis on 07-Nov-15.
 */

/**
 * Created by Charis on 06-Nov-15.
 */
public class WunderBarController implements Serializable {

    int settings;
    private Map<Integer,String> defaultcommands;
    private Map<Integer,String> customcommands;

    public Command findCommand(String str){

        return null;

    }

    public void executeCommand(Command cmd){

        cmd.execute();

    }

}
```

1.5 Πηγαίος Κώδικας για Layouts

activity_main.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity"
    android:background="@drawable/conne" >

    <LinearLayout
        android:orientation="vertical"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_alignParentBottom="false"
        android:layout_alignParentTop="false"
        android:layout_alignParentRight="false"
        android:layout_alignParentEnd="false"
        android:paddingBottom="@dimen/activity_vertical_margin"
        android:paddingRight="@dimen/activity_horizontal_margin"
        android:paddingTop="@dimen/activity_vertical_margin"
        android:paddingLeft="@dimen/activity_horizontal_margin" >

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/title"
            android:id="@+id/title"
            android:textSize="33sp"
            android:textStyle="bold|italic"
            android:layout_gravity="center" />

    </LinearLayout>

</RelativeLayout>
```

Activity_platform_chooser.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context="com.example.charis.hrc.Platform_chooser"
    android:background="@drawable/unt">

    <LinearLayout
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <TextView android:text="Please choose your platform:"
android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

        <ScrollView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/scrollView5" >

            <LinearLayout
                android:orientation="vertical"
                android:layout_width="match_parent"
                android:layout_height="match_parent">

                <LinearLayout
                    android:orientation="horizontal"
                    android:layout_width="match_parent"
                    android:layout_height="match_parent">

                    <ImageView
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:id="@+id/imageView"
                        android:src="@drawable/smartthings" />

                    <TextView
                        android:layout_width="wrap_content"
                        android:layout_height="match_parent"
                        android:textAppearance="?android:attr/textAppearanceMedium"
                        android:id="@+id/textView16"
                        android:text="Smart Things"
                        android:gravity="center_vertical|center_horizontal"
                        android:layout_marginLeft="20dp" />

                    <CheckBox
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:id="@+id/smartThingsPlatform"
                        android:layout_marginLeft="10dp" />

                </LinearLayout>

                <LinearLayout
                    android:orientation="horizontal"
                    android:layout_width="match_parent"
                    android:layout_height="match_parent" >

                    <ImageView
```

```

        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/imageView2"
        android:src="@drawable/wunderbar" />

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
android:textAppearance="?android:attr/textAppearanceMedium"
            android:id="@+id/textView20"
            android:text="WunderBar"
            android:gravity="center_vertical|center_horizontal"
            android:layout_marginLeft="20dp" />

        <CheckBox
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/wunderbarPlatform"
            android:layout_marginLeft="30dp" />

    </LinearLayout>
</LinearLayout>
</ScrollView>

<CheckBox
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Remember my choice"
    android:id="@+id/checkBox"
    android:layout_marginTop="100dp" />

<Button
    android:layout_width="match_parent"
    android:layout_height="50dp"
    android:text="OK"
    android:id="@+id/chosenPlatforms"
    android:layout_marginTop="100dp" />
</LinearLayout>
</RelativeLayout>

```

Activity_scenarios.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context="com.example.charis.hrc.Scenarios">

    <ScrollView
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:id="@+id/scrollView6" >

        <LinearLayout
            android:orientation="vertical"
            android:layout_width="fill_parent"
            android:layout_height="fill_parent">

            <Button
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:text="Wake Up scenario"
                android:id="@+id/wakeUpScenarioButton" />

        </LinearLayout>
    </ScrollView>

    <ViewFlipper
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/viewFlipper2"
        android:visibility="invisible">

        <include android:id="@+id/wakeup" layout="@layout/wakeupsenario"/>

    </ViewFlipper>
</RelativeLayout>
```

Activity_task_manager.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context="com.example.charis.hrc.TaskManager"
    android:id="@+id/mainLayout"
    android:background="@drawable/backgroundimage">

    <ScrollView
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:id="@+id/scrollView3"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentBottom="true"
        android:layout_alignParentRight="true">

        <LinearLayout
            android:orientation="vertical"
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <LinearLayout
                android:orientation="horizontal"
                android:layout_width="match_parent"
                android:layout_height="match_parent">

                <ImageButton
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:id="@+id/infoIcon"
                    android:src="@drawable/info"
                    android:background="@null" />

                <ImageButton
                    android:layout_width="85dp"
                    android:layout_height="85dp"
                    android:id="@+id/imageButton"
                    android:src="@drawable/speaker"
                    android:layout_gravity="center_horizontal"
                    android:adjustViewBounds="false"
                    android:background="@null"
                    android:layout_marginLeft="50dp" />

                <ImageButton
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:id="@+id/goingout"
                    android:src="@drawable/goingout"
                    android:background="@null"
                    android:layout_gravity="top"
                    android:layout_marginLeft="80dp" />

            </LinearLayout>

        </LinearLayout>
```

```

<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <ImageButton
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/settingsIcon"
        android:src="@drawable/setting"
        android:background="@null"
        android:layout_marginTop="10dp" />

    <ProgressBar
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/progressBar"
        android:layout_gravity="center"
        android:layout_marginLeft="70dp" />

    <ImageButton
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/scenariosIcon"
        android:src="@drawable/scenarios"
        android:background="@null"
        android:layout_marginTop="10dp"
        android:paddingLeft="75dp"
        android:layout_marginLeft="20dp" />
</LinearLayout>

    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <TextView
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:id="@+id/recordedWords"
            android:textAlignment="center"
            android:visibility="visible"
            android:layout_gravity="fill_horizontal|center"
            android:layout_marginLeft="110dp" />
        </LinearLayout>

    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <GridLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <ImageButton
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:id="@+id/clockIcon"
                android:src="@drawable/clock"
                android:background="@null"
                android:layout_marginTop="15dp" />

            <ImageButton
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"

```

```

        android:id="@+id/goingout2Icon"
        android:src="@drawable/goingout2"
        android:background="@null"
        android:layout_marginTop="20dp"
        android:paddingLeft="75dp"
        android:layout_marginLeft="135dp" />
    </GridLayout>

    </LinearLayout>
    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <ImageButton
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/watertemperature"
            android:src="@drawable/watertemperature"
            android:background="@null"
            android:layout_marginTop="15dp" />

        <ImageButton
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/homeappliancesIcon"
            android:src="@drawable/homeappliances"
            android:background="@null"
            android:layout_marginTop="20dp"
            android:paddingLeft="75dp" />

        <ImageButton
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/securityIcon"
            android:src="@drawable/security"
            android:background="@null"
            android:layout_marginTop="20dp"
            android:paddingLeft="75dp" />
    </LinearLayout>

    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <ImageButton
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/temperatureIcon"
            android:src="@drawable/temperature"
            android:background="@null"
            android:layout_marginTop="15dp" />

        <ImageButton
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/lampIcon"
            android:src="@drawable/lamp"
            android:background="@null"
            android:layout_marginTop="20dp"
            android:paddingLeft="75dp" />

```

```

        <ImageButton
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/securevehicleIcon"
            android:src="@drawable/securevehicle"
            android:background="@null"
            android:layout_marginTop="20dp"
            android:paddingLeft="75dp" />
    </LinearLayout>

    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <ImageButton
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/greenIcon"
            android:src="@drawable/green"
            android:background="@null"
            android:layout_marginTop="20dp" />

        <ImageButton
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/cinemaIcon"
            android:src="@drawable/cinema"
            android:background="@null"
            android:layout_marginTop="20dp"
            android:paddingLeft="75dp" />

        <ImageButton
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/cameraIcon"
            android:src="@drawable/camera"
            android:background="@null"
            android:layout_marginTop="20dp"
            android:paddingLeft="75dp" />
    </LinearLayout>

    <TextView android:text="@string/instruction1"
android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/instruction1"
            android:layout_gravity="center"
            android:visibility="gone" />

    <Button
        android:layout_width="330dp"
        android:layout_height="wrap_content"
        android:text="Text to speech"
        android:id="@+id/speechbutton"
        android:background="@null"
        android:visibility="gone" />
</LinearLayout>
</ScrollView>

<ViewFlipper
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/viewFlipper"

```

```
        android:visibility="invisible">
        <include android:id="@+id/generalsettings" layout="@layout/settings"/>
        <include android:id="@+id/voices"
layout="@layout/voicecommandsettings"/>
        <include android:id="@+id/infolayout" layout="@layout/info"/>

        <include android:id="@+id/voices" layout="@layout/wakeupsenario"/>

        <include android:id="@+id/sensormeasures"
layout="@layout/measuresboard"/>

        <include android:id="@+id/dri" layout="@layout/drivingmode"/>

    </ViewFlipper>

</RelativeLayout>
```

Driving_mode.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/dri"
    android:visibility="visible">

    <ImageButton
        android:layout_width="400dp"
        android:layout_height="500dp"
        android:id="@+id/driverMode"
        android:src="@drawable/drivingmode" />

    <Switch
        android:layout_width="match_parent"
        android:layout_height="66dp"
        android:text="Turn off driving mode"
        android:id="@+id/switcher"
        android:background="@drawable/backgroundimage" />

</LinearLayout>
```

Info.xml

```
<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:id="@+id/infolayout"
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:background="@drawable/unt">

    <ScrollView
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:id="@+id/scrollView4"
        android:layout_alignParentEnd="false"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentBottom="true"
        android:layout_alignParentRight="true"
        android:layout_alignParentStart="false">

        <LinearLayout
            android:orientation="vertical"
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
            android:weightSum="1">

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"

                android:textAppearance="?android:attr/textAppearanceMedium"
                android:text="@string/howtoHelper"
                android:id="@+id/recordText" />

            <ImageView
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:id="@+id/recordHelp"
                android:src="@drawable/speaker" />

            <LinearLayout
                android:orientation="horizontal"
                android:layout_width="match_parent"
                android:layout_height="match_parent" >

                <ImageView
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:id="@+id/scenariosHelp"
                    android:src="@drawable/scenarios"
                    android:layout_marginTop="20dp" />

                <TextView
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"

                    android:textAppearance="?android:attr/textAppearanceMedium"
                    android:text="@string/scenariosHelper"
                    android:id="@+id/scenariosText"
                    android:layout_marginTop="25dp" />

            </LinearLayout>

            <LinearLayout
                android:orientation="horizontal"
```

```

        android:layout_width="match_parent"
        android:layout_height="match_parent" >

        <ImageView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/settingsHelp"
            android:src="@drawable/setting"
            android:layout_marginTop="30dp" />

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:text="@string/settingsHelp"
            android:id="@+id/settignsText"
            android:layout_marginTop="20dp" />
    </LinearLayout>

    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="match_parent"
        android:layout_height="match_parent" >

        <ImageView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/quitHelp"
            android:src="@drawable/goingout"
            android:layout_marginTop="25dp" />

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:text="@string/exitText"
            android:id="@+id/quitText"
            android:layout_marginTop="30dp" />
    </LinearLayout>

    <Button
        android:layout_width="match_parent"
        android:layout_height="50dp"
        android:text="Back to the menu"
        android:id="@+id/backInfo"
        android:background="@drawable/backgroundimage"
        android:layout_weight="12.93" />
    </LinearLayout>
</ScrollView>
</RelativeLayout>

```

Measures_board.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/sensormeasures"
    android:background="@drawable/backgroundimage">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceMedium"
        android:text="Temperature"
        android:id="@+id/textView5"
        android:layout_gravity="center" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceMedium"
        android:id="@+id/temperaturehum"
        android:layout_gravity="center"
        android:textAlignment="center" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceMedium"
        android:text="Humidity"
        android:id="@+id/textView8"
        android:layout_gravity="center" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceMedium"
        android:id="@+id/wunderbarhumidity"
        android:layout_gravity="center"
        android:textAlignment="center" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceMedium"
        android:text="Lightness"
        android:id="@+id/textView12"
        android:layout_gravity="center"
        android:textAlignment="center" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceMedium"
        android:id="@+id/humidityText"
        android:layout_gravity="center" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceMedium"
        android:id="@+id/textView14"
        android:layout_gravity="center" />

    <Button
        android:layout_width="match_parent"
        android:layout_height="43dp">
```

```
        android:text="Go to main menu"
        android:id="@+id/measuresBoardback"
        android:background="@drawable/backgroundimage" />
</LinearLayout>
```

Settings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/generalsettings"
    android:background="@drawable/backgroundimage"
    android:weightSum="1">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceMedium"
        android:text="Mode settings"
        android:id="@+id/modesettings" />

    <Switch
        android:layout_width="fill_parent"
        android:layout_height="38dp"
        android:text="Driving mode"
        android:id="@+id/drivingSwitch"
        android:background="@drawable/backgroundimage" />

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:id="@+id/textView7"
        android:layout_weight="0.10" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceMedium"
        android:text="Voice settings"
        android:id="@+id/voicelabel" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="39dp"
        android:text="Go to voice settings"
        android:id="@+id/voicecmds"
        android:background="@drawable/backgroundimage" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceMedium"
        android:id="@+id/textView10" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="39dp"
        android:text="Restore default voice commands"
        android:id="@+id/button"
        android:background="@drawable/backgroundimage" />

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:id="@+id/textView"
        android:layout_weight="0.10" />

    <TextView
        android:layout_width="match_parent"
```

```
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceMedium"
        android:id="@+id/textView23" />

    <Button
        android:layout_width="match_parent"
        android:layout_height="30dp"
        android:text="Back to the menu"
        android:id="@+id/settingsback"
        android:background="@drawable/backgroundimage" />

</LinearLayout>
```

Voicecommandsettings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/voices"
    android:background="@drawable/backgroundimage">

    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content">

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:text="Device"
            android:id="@+id/textView4" />

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:text="Default commands"
            android:id="@+id/textView2"
            android:layout_marginLeft="100dp" />
    </LinearLayout>

    <TableRow
        android:layout_width="wrap_content"
        android:layout_height="wrap_content">

    </TableRow>

    <ScrollView
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:id="@+id/scrollView" >

        <TableLayout
            android:layout_width="fill_parent"
            android:layout_height="fill_parent">

            <TableRow
                android:layout_width="match_parent"
                android:layout_height="match_parent">

                <TextView
                    android:layout_width="fill_parent"
                    android:layout_height="fill_parent"
                    android:textAppearance="?android:attr/textAppearanceLarge"
                    android:text="ON/OFF"
                    android:id="@+id/textView21" />
            </TableRow>

            <TableRow
                android:layout_width="match_parent"
                android:layout_height="match_parent">

                <TextView
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
```

```

android:textAppearance="?android:attr/textAppearanceMedium"
    android:text="coffee maker"
    android:id="@+id/textView3" />

    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/coffeeTxt"
        android:text="make coffee" />
</TableRow>

<TableRow
    android:layout_width="match_parent"
    android:layout_height="match_parent" >

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"

android:textAppearance="?android:attr/textAppearanceMedium"
    android:text="toast maker"
    android:id="@+id/textView6" />

    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/makeToast"
        android:text="make toast"
        android:clickable="false" />
</TableRow>

<TableRow
    android:layout_width="match_parent"
    android:layout_height="match_parent" >

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"

android:textAppearance="?android:attr/textAppearanceMedium"
    android:text="motion sensors"
    android:id="@+id/textView9" />

    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/editText3"
        android:text="enable motion sensors"
        android:editable="false" />
</TableRow>

<TableRow
    android:layout_width="match_parent"
    android:layout_height="match_parent" >

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"

android:textAppearance="?android:attr/textAppearanceMedium"
    android:text="cameras"
    android:id="@+id/textView11" />

    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"

```

```

        android:text="enable cameras"
        android:id="@+id/editText4" />
</TableRow>

<TableRow
    android:layout_width="match_parent"
    android:layout_height="match_parent" >

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"

android:textAppearance="?android:attr/textAppearanceMedium"
        android:text="bedroom 1 lights"
        android:id="@+id/textView13" />

    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="room lights 1"
        android:id="@+id/editText5" />
</TableRow>

<TableRow
    android:layout_width="match_parent"
    android:layout_height="match_parent" >

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"

android:textAppearance="?android:attr/textAppearanceMedium"
        android:text="bedroom 2 lights"
        android:id="@+id/textView15" />

    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="bedroom 2 lights"
        android:id="@+id/editText6" />
</TableRow>

<TableRow
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:id="@+id/textView17" />
</TableRow>

<TableRow
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:text="OPEN/CLOSE"
        android:id="@+id/textView18" />
</TableRow>

```

```

        <TableRow
            android:layout_width="match_parent"
            android:layout_height="match_parent" >

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"

android:textAppearance="?android:attr/textAppearanceMedium"
                android:text="garage gate"
                android:id="@+id/textView22" />


            <EditText
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="garage door"
                android:id="@+id/editText8" />
        </TableRow>

        <TableRow
            android:layout_width="match_parent"
            android:layout_height="match_parent" >

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"

android:textAppearance="?android:attr/textAppearanceMedium"
                android:text="rear door"
                android:id="@+id/textView19" />


            <EditText
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="rear door"
                android:id="@+id/editText7" />
        </TableRow>

        <TableRow
            android:layout_width="match_parent"
            android:layout_height="match_parent" >

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"

android:textAppearance="?android:attr/textAppearanceMedium"
                android:text="front door"
                android:id="@+id/textView24" />


            <EditText
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="front door"
                android:id="@+id/editText9" />
        </TableRow>

        <TableRow
            android:layout_width="match_parent"
            android:layout_height="match_parent" >

```

```

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"

android:textAppearance="?android:attr/textAppearanceMedium"
            android:text="kitchen window"
            android:id="@+id/textView26" />

        <EditText
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="kitchen door"
            android:id="@+id/editText10" />
    </TableRow>

    <TableRow
        android:layout_width="match_parent"
        android:layout_height="match_parent" >

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceLarge"
            android:text="CHECK"
            android:id="@+id/textView28" />
    </TableRow>

    <TableRow
        android:layout_width="match_parent"
        android:layout_height="match_parent" >

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"

android:textAppearance="?android:attr/textAppearanceMedium"
            android:text="water temperature"
            android:id="@+id/textView29" />

        <EditText
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="for cold water"
            android:id="@+id/editText11" />
    </TableRow>

    <TableRow
        android:layout_width="match_parent"
        android:layout_height="match_parent" >

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"

android:textAppearance="?android:attr/textAppearanceMedium"
            android:text=" temperature"
            android:id="@+id/textView31" />

        <EditText
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="for cold temperature"
            android:id="@+id/editText12" />
    </TableRow>

```

```

        <Button
            android:layout_width="match_parent"
            android:layout_height="50dp"
            android:text="Save Changes"
            android:id="@+id/button2"
            android:background="@drawable/backgroundimage" />

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:id="@+id/textView25" />

        <Button
            android:layout_width="match_parent"
            android:layout_height="50dp"
            android:text="Go to menu"
            android:id="@+id/voiceCommandsback"
            android:background="@drawable/backgroundimage" />

    </TableLayout>
</ScrollView>

</LinearLayout>

```

Wakeup_scenario.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/wakeup">

    <ScrollView
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:id="@+id/scrollView2" >

        <LinearLayout
            android:orientation="vertical"
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <TextView
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:textAppearance="?android:attr/textAppearanceMedium"
                android:text="Check temperature of water at: "
                android:id="@+id/checkWaterTempText" />

            <TextView
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:textAppearance="?android:attr/textAppearanceMedium"
                android:text="Set boiler preferred temperature at: "
                android:id="@+id/setBoilerText" />

            <TextView
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:textAppearance="?android:attr/textAppearanceMedium"
                android:text="Activate alarm at: "
                android:id="@+id/activateAlarmText" />

            <TextView
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:textAppearance="?android:attr/textAppearanceMedium"
                android:text="Turn on bedroom's lights at:"
                android:id="@+id/bedroomText" />

            <TextView
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:textAppearance="?android:attr/textAppearanceMedium"
                android:text="Turn on bathroom's lights at:"
                android:id="@+id/bathroomText" />

            <TextView
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:textAppearance="?android:attr/textAppearanceMedium"
                android:text="Turn on coffee maker at:"
                android:id="@+id/coffeeMakerText" />

            <TextView
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:textAppearance="?android:attr/textAppearanceMedium"
                android:text="Turn on toast maker at:"
                android:id="@+id/toastMakerText" />

        </LinearLayout>

    </ScrollView>

</RelativeLayout>
```

```

        <TextView
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:text="Turn on kitchen's lights at: "
            android:id="@+id/kitchenText" />

        <TextView
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:text="Unlock back door at: "
            android:id="@+id/unlockBackDoorText" />

        <TextView
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:text="Open garage door at: "
            android:id="@+id/openGarageDoorText" />

        <TextView
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:text="Close garage gate at: "
            android:id="@+id/closeGarageGateText" />

        <TextView
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:text="Enable security guard at : "
            android:id="@+id/enableSecurityGuardText" />

        <TextView
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:text="Enable family notifier "
            android:id="@+id/enableFamilyNotifierText" />

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:text="Add command"
            android:id="@+id/addCommandText" />
    </LinearLayout>
</ScrollView>

<EditText
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:inputType="number"
    android:ems="10"
    android:id="@+id/editText"
    android:layout_centerInParent="true"
    android:visibility="invisible" />

<TimePicker
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/timePicker"
    android:visibility="invisible" />
</RelativeLayout>

```