Πολυτεχνείο Κρήτης
Σχολή Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών

Διπλωματική Εργασία

Πλοήγηση σε εσωτερικούς χώρους με χρήση αισθητήρων στο υπολογιστικό νέφος

Indoors navigation using sensors on the cloud

Παναγιώτης Κωνσταντόπουλος

Εξεταστική επιτροπή

Ευριπίδης Πετράκης, Καθηγητής (Επιβλέπων)

Βασίλης Σαμολαδάς, Επίκουρος Καθηγητής

Δρ. Στέλιος Σωτηριάδης, Επιστημονικός Συνεργάτης

ΧΑΝΙΑ 2017

Περίληψη

Αξιοποιώντας τις τεχνολογίες του Υπολογιστικού Νέφους και του Διαδικτύου των Πραγμάτων, υλοποιήσαμε ένα σύστημα πλοήγησης σε εσωτερικούς χώρους κτηρίων. Αισθητήρες ανίχνευσης προσέγγισης τοποθετούνται σε κάθε χώρο του κτηρίου. Κάνοντας χρήση του πρωτοκόλλου ασύρματης συνδεσιμότητας BLE (Bluetooth Low Energy) τα κινητά τηλέφωνα των επισκεπτών είναι σε θέση να προσδιορίσουν την απόστασή τους από τον κάθε αισθητήρα. Μέσω ειδικής εφαρμογής για το τηλέφωνο, η πληροφορία αυτή αποστέλλεται στο Υπολογιστικό Νέφος όπου υπολογίζεται η διαδρομή που πρέπει να ακολουθηθεί. Οι οδηγίες πλοήγησης εν συνεχεία αποστέλλονται πίσω στην εφαρμογή, από όπου και προβάλλονται από τη συσκευή του χρήστη. Παράλληλα, αναπτύξαμε και ένα ολοκληρωμένο σύστημα διαχείρισης, προσβάσιμο μέσω διαδικτυακής διεπαφής. Μεταξύ των άλλων η διαδικτυακή εφαρμογή περιλαμβάνει: Παραχώρηση αδειών πρόσβασης σε επιλεγμένους χώρους ανά χρήστη, παρακολούθηση σε πραγματικό χρόνο της πλήρους δραστηριότητας ανά επισκέπτη και ανά χώρο, επικοινωνία μέσω μηνυμάτων ανάμεσα σε χρήστες και διαχειριστή, καθώς και εξατομικευμένες ειδοποιήσεις σε περίπτωση εσφαλμένης πορείας.

Abstract

Taking advantage of Cloud Computing and Internet of Things technologies, we created an indoors navigation system that runs on the cloud and supports communication with user's mobile devices. Proximity detection sensors are placed in every location of the building. Using the wireless connectivity BLE (Bluetooth Low Energy) protocol, supported by the majority of smart devices, the visitors' smartphones are capable of estimating their distance from these sensors. Making use of a specially designed mobile application, this information is sent to the cloud, where the computation of the desired route take place. The navigation instructions are then sent back to the user's device, where they are displayed by the mobile application. Furthermore, we have also developed a complete administration system, accessible via web interface. Besides the above, the following functionality is supported as well: providing access permissions in selected indoors areas per user, real time monitoring of areas' and users' activity, communication with the administrator by exchanging messages and individual notifications in case of user's disorientation.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον κύριο Πετράκη για την ουσιαστική καθοδήγηση του μέχρι την ολοκλήρωση της διπλωματικής μου εργασίας. Ευχαριστώ επίσης τους βοηθούς και όλα τα μέλη του εργαστηρίου - προπτυχιακούς και μεταπτυχιακούς φοιτητές - για την άριστη επικοινωνία και συνεργασία μας όλο αυτό το διάστημα.

Περιεχόμενα

Κεφάλαιο 1 - Εισαγωγή

1.1 Σκοπός της Εργασίας

Το πρόβλημα της πλοήγησης σε εσωτερικούς χώρους γίνεται κυρίως αντιληπτό σε μεγάλες κτηριακές εγκαταστάσεις (π.χ. γραφεία εταιριών, πολυκαταστήματα). Κατά την είσοδο σε μια τέτοια υποδομή, πραγματοποιείται αρχικά έλεγχος, ώστε να διαπιστωθεί αν πράγματι ο επισκέπτης έχει δικαίωμα πρόσβασης. Η περιήγηση στο χώρο, πραγματοποιείται με την βοήθεια ενδείξεων και επιγραφών που είναι κατάλληλα τοποθετημένες σε ευδιάκριτα σημεία. Μία συνήθης επίσης τακτική είναι η έκκληση για βοήθεια προς τους υπαλλήλους της εταιρείας που βρίσκονται στην εγγύτερη περιοχή. Ωστόσο, λόγω του μεγάλου αριθμού επισκεπτών κάθε στιγμή, το διαθέσιμο προσωπικό είναι πολύ πιθανόν να μην επαρκεί για να εξυπηρετήσει όλους τους επισκέπτες ταυτόχρονα. Επίσης οι ενδείξεις και επιγραφές ενδέχεται να μην δίνουν πάντοτε σαφείς οδηγίες για την επιθυμητή διαδρομή. Ως αποτέλεσμα, ο χρήστης πιθανόν να ταλαιπωρηθεί για αρκετή ώρα περιπλανώμενος, απασχολώντας προσωπικό και καθυστερώντας να φτάσει στον προορισμό του.

Σκοπός της παρούσης εργασίας, είναι η επίλυση του παραπάνω προβλήματος μέσω της σχεδίασης και ανάπτυξης ενός συστήματος αυτοματοποιημένης πλοήγησης για εσωτερικούς χώρους. Εκμεταλλευόμενοι γνωστές τεχνολογίες ασύρματης δικτύωσης και πρόσβασης στο διαδίκτυο, οι οποίες είναι διαθέσιμες σε κάθε χρήστη μέσα από το κινητό του τηλέφωνό (smartphone), θα δημιουργήσουμε μια εφαρμογή για κινητές συσκευές (ταμπλέτες, κινητά τηλέφωνα), ικανή να εντοπίζει την θέση του επισκέπτη στο χώρο και να αποστέλλει με βάση αυτή, οδηγίες κατεύθυνσης προς τον επιθυμητό προορισμό.

1.2 Προτεινόμενη Λύση

Κατασκευάσαμε ένα σύστημα αυτοματοποιημένης πλοήγησης για εσωτερικούς χώρους με την χρήση αισθητήρων ανίχνευσης προσέγγισης (proximity beacons). Οι αισθητήρες αυτού του είδους χρησιμοποιούν το πρωτόκολλο ασύρματης δικτύωσης Bluetooth Low Energy[1] (BLE) για να επικοινωνήσουν με άλλες συσκευές. Το συγκεκριμένο πρωτόκολλο παρέχει τα εξής οφέλη: α) Μεγάλη αυτονομία μπαταρίας σε συνδυασμό με πολύ χαμηλό κόστος αγοράς και συντήρησης των αισθητήρων β) Συμβατότητα με όλα σχεδόν τις σύγχρονες κινητές συσκευές, καθώς το BLE αποτελεί το πλέον καθιερωμένο εμπορικό στάνταρντ ασύρματης συνδεσιμότητας ηλεκτρονικών συσκευών, ανεξαρτήτως κατασκευαστή. Μέσω μιας εφαρμογής για το τηλέφωνο, είναι δυνατόν να εντοπίζονται αισθητήρες που βρίσκονται στο χώρο και να

[1] https://developer.android.com/guide/topics/connectivity/bluetooth-le.html

ταξινομούνται με βάση την απόσταση τους. Η πληροφορία αυτή, παράλληλα με τον προορισμό που έχει επιλεγεί από τον χρήστη, αποστέλλεται στο Υπολογιστικό Νέφος (ΥΝ).

Στο τμήμα της εφαρμογής που βρίσκεται στο νέφος, εκτελείται ο αλγόριθμος προσδιορισμού της επιθυμητής διαδρομής. Αρχικά γίνεται αντιστοίχιση του κάθε αισθητήρα σε μία και μοναδική περιοχή, με βάση το αναγνωριστικό του. Αφού προσδιοριστεί η θέση του χρήστη στο χώρο, υπολογίζεται με βάση τον επιλεγμένο προορισμό η επόμενη τοποθεσία που ανήκει στην διαδρομή. Σε περίπτωση που η τοποθεσία του χρήστη δεν ανήκει σε αυτή, τότε και μόνο εξετάζονται αν υπάρχει γειτονική τοποθεσία η οποία αποτελεί μέρος της, ώστε να δοθεί νέα κατευθυντήρια οδηγία. Σε αντίθετη περίπτωση ο χρήστης θεωρούμε ότι έχει χάσει τον προσανατολισμό του, οπότε στέλνουμε αίτημα για βοήθεια στον διαχειριστή του συστήματος.

Υπάρχει ένα αρχικό σημείο εισόδου και ένας ή περισσότεροι προορισμοί. Υποθέτουμε ότι ο επισκέπτης έχει εκδηλώσει τον ενδιαφέρον του να επισκεφθεί μία συγκεκριμένη τοποθεσία μέσα στο κτήριο, έχει πάρει άδεια για αυτό τον σκοπό και παράλληλα ο διαχειριστής του κτηρίου του έχει στείλει στο λογαριασμό του (π.χ. email) τις οδηγίες κατεύθυνσης που πρέπει να ακολουθήσει. Η διαδρομή για κάθε προορισμό έχει οριστεί εκ των προτέρων από τον διαχειριστή του συστήματος. Αυτή δεν είναι απαραίτητα η πιο σύντομη διαδρομή, αλλά μπορεί να είναι η πιο εύκολη ή η επιτρεπόμενη διαδρομή για επισκέπτες. Οι οδηγίες πλοήγησης που αποστέλλονται ικανοποιούν την παραπάνω συνθήκη. Η χρήση του ΥΝ μας παρέχει τα κατάλληλα εργαλεία και τις υπηρεσίες ώστε να φτιάξουμε ένα ολοκληρωμένο συστήματα διαχείρισης των επιτρεπόμενων διαδρομών των επισκεπτών και παρακολούθηση της θέσης αυτών μέσα στο κτήριο σε πραγματικό χρόνο, μέσω ειδικά διαμορφούμενου ιστοχώρου.

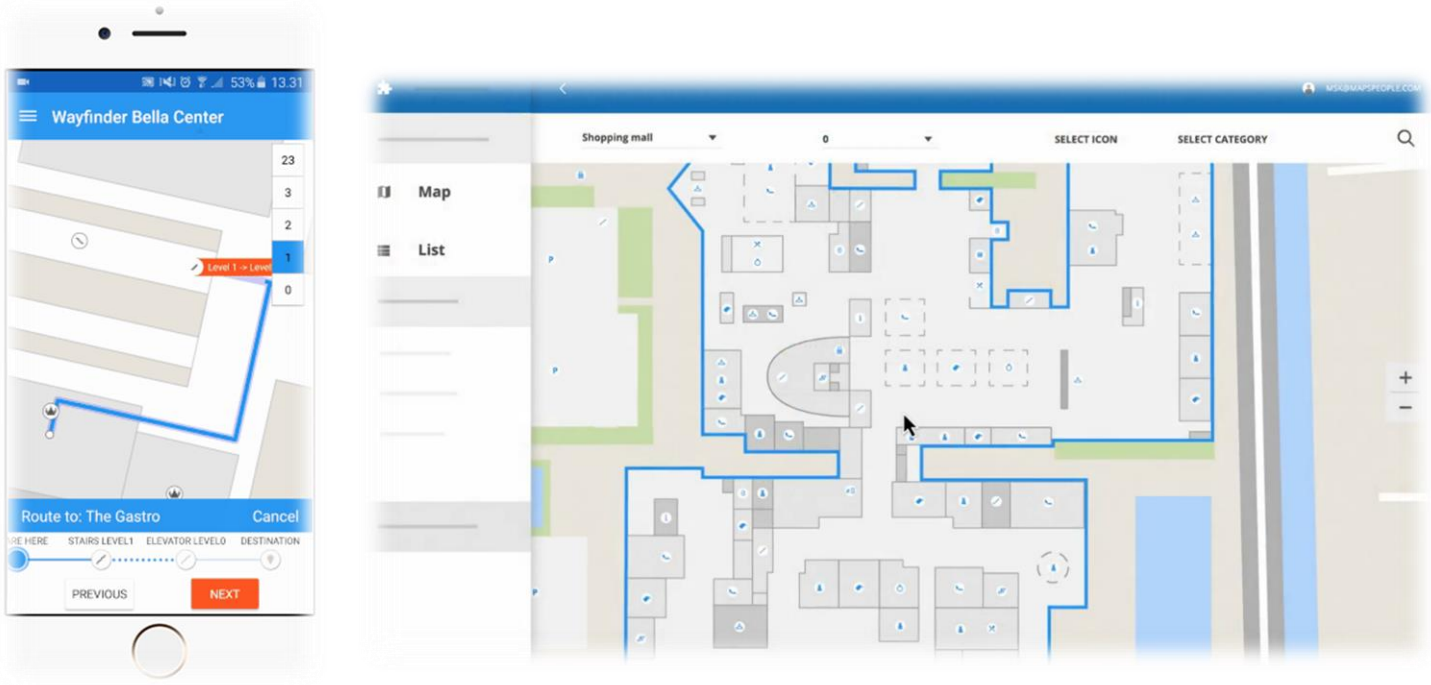
1.3 Σχετικά Συστήματα

Η ιδέα της πλοήγησης σε εσωτερικούς χώρους υποβοηθούμενη από λογισμικό δεν είναι καινούρια. Υπάρχει ήδη πληθώρα εμπορικών εφαρμογών από διάφορους κατασκευαστές που καλύπτουν σχεδόν στο σύνολό τους όλες τις ανάγκες για το συγκεκριμένο θέμα. Οι εμπορικές εφαρμογές είναι κλειστού κώδικα, συνεπώς δεν μπορούμε να γνωρίζουμε λεπτομέρειες για τον αλγόριθμο πλοήγησης που χρησιμοποιούν. Η λειτουργία ωστόσο των εφαρμογών αυτών βασίζεται κυρίως σε αισθητήρες εντοπισμού θέσης οι οποίοι είναι τοποθετημένοι σε διάφορα σημεία ενδιαφέροντος. Ακριβώς όπως και στην δική μας εργασία, οι αισθητήρες αυτοί χρησιμοποιούν κάποιο non-IP πρωτόκολλο επικοινωνίας με το κινητό τηλέφωνο (π.χ. Bluetooth, RFID, ZigBee, Wi-Fi Direct κλπ.). Ο χάρτης του κτηρίου συνήθως ενσωματώνεται στα εκτελέσιμα αρχεία της εφαρμογής από τον ίδιο τον κατασκευαστή. Κατόπιν ανάλογα με την εφαρμογή μπορεί να υποστηρίζεται είτε λειτουργία πλοήγησης χωρίς σύνδεση, όπου οι χάρτες και οι οδηγίες βρίσκονται όλες αποθηκευμένες στη συσκευή του χρήστη, είτε δικτυακή λειτουργία όπου ορισμένες ενέργειες πραγματοποιούνται τοπικά στη

συσκευή, ενώ άλλες πραγματοποιούνται απομακρυσμένα σε κάποιον εξυπηρετητή. Στην δεύτερη περίπτωση, ο διαχειριστής της εφαρμογής έχει συνήθως πρόσβαση σε κάποια ιστοσελίδα όπου μπορεί να διαμορφώσει τους χάρτες του κτηρίου. Ορισμένες από τις πιο γνωστές εμπορικές εφαρμογές είναι οι παρακάτω:

mapspeople[2]

Το ενδιαφέρον που παρουσιάζει αυτή η υλοποίηση είναι πως χρησιμοποιεί το API της Google και εκτελείται μέσα από την ίδια την εφαρμογή του Google Maps ως πρόσθετο. Η γραφική διεπαφή χρήστη είναι αυτή του Google Maps, συνεπώς γνώριμη στην εμφάνισή της, ενώ ο επισκέπτης συνδέεται κατευθείαν μέσω του Google λογαριασμού του για να αποκτήσει πρόσβαση στην υπηρεσία. Η εφαρμογή στο κινητό τηλέφωνο εκτελείται μέσω του διαδικτύου, από όπου και λαμβάνονται οι οδηγίες πλοήγησης, ωστόσο υπάρχει και η δυνατότητα της εκ των προτέρων φόρτωσης του χάρτη του κτηρίου στη συσκευή ώστε να επιτυγχάνεται και πλοήγηση χωρίς σύνδεση. Η πλοήγηση πραγματοποιείται μέσω αισθητήρων ανίχνευσης προσέγγισης που τοποθετήθηκαν από τον διαχειριστή του κτηρίου σε συνεννόησή με τον κατασκευαστή. Ο διαχειριστής επίσης έχει πρόσβαση σε μια ιστοσελίδα με κατάλληλα διαμορφωνόμενο περιβάλλον που διευκολύνει την επεξεργασία και τροποποίηση του χάρτη.



Εικόνα 1: Γραφική διεπαφή της εφαρμογής "maps people", για το τηλέφωνο (αριστερά) και για τη δικτυακή εφαρμογή διαχείρισης χαρτών (δεξιά)

[2] https://www.mapspeople.com/mapsindoors/

infsoft smart connected locations [3]

Η εταιρεία προσφέρει λύσεις πλοήγησης εσωτερικών χώρων, επαγγελματικών προδιαγραφών, πλήρως παραμετροποιήσιμες για να καλύπτουν τις ανάγκες οποιασδήποτε κτηριακής υποδομής. Η πλοήγηση και εδώ πραγματοποιείται μέσω αισθητήρων που τοποθετούνται στους χώρους ενδιαφέροντος σύμφωνα με τις οδηγίες του κατασκευαστή. Ανάλογα με τις συνθήκες που επικρατούν στο κτήριο (έκταση χώρων, επιθυμητή ακρίβεια εντοπισμού κλπ.) η εταιρεία θα μας προτείνει το ιδανικό μοντέλο αισθητήρων με το κατάλληλο πρωτόκολλο ασύρματης μεταφοράς δεδομένων. Η εφαρμογή διαθέτει συμβατότητα με οποιοδήποτε πρωτόκολλο επικοινωνίας επιλέξουμε, με την προϋπόθεση φυσικά ότι το πρωτόκολλο υποστηρίζεται και από το υλικό της συσκευής όπου αυτή θα τρέχει. Η πλοήγηση μπορεί να πραγματοποιείται είτε τοπικά στη συσκευή είτε μέσω διαδικτύου, με ανάλογο τρόπο που περιγράψαμε και στην προηγούμενη υλοποίηση. Υπάρχει και σε αυτήν την περίπτωση ιστοσελίδα διαχείρισης για τους χάρτες του κτηρίου. Το κύριο χαρακτηριστικό ωστόσο της συγκεκριμένης εφαρμογής, που την κάνει να ξεχωρίζει, είναι ότι εκτός της πλοήγησης προσφέρει και πολλές περισσότερες λειτουργίες μέσω διαδικτύου. Συγκεκριμένα επιτρέπει την συλλογή στατιστικών στοιχείων πληρότητας χώρων, τον έλεγχο σε πραγματικό χρόνο της κίνησης των επισκεπτών στο κτήριο, ενώ επίσης είναι εφικτή και η αμφίδρομη επικοινωνία με τους αισθητήρες, αξιοποιώντας τους και ως ενεργητικές συσκευές. Έτσι μπορούμε μέσω των αισθητήρων αυτών να ανοίγουμε πόρτες σε επιλεγμένους χρήστες που πλησιάζουν, να έχουμε σύστημα αυτόματου ελέγχου εισόδου και πληρωμών μέσα από την εφαρμογή κλπ. Με άλλα λόγια η εφαρμογή δεν περιορίζεται στην πλοήγηση, αλλά επιτρέπει και την αλληλεπίδραση με πλήθος έξυπνων συσκευών.

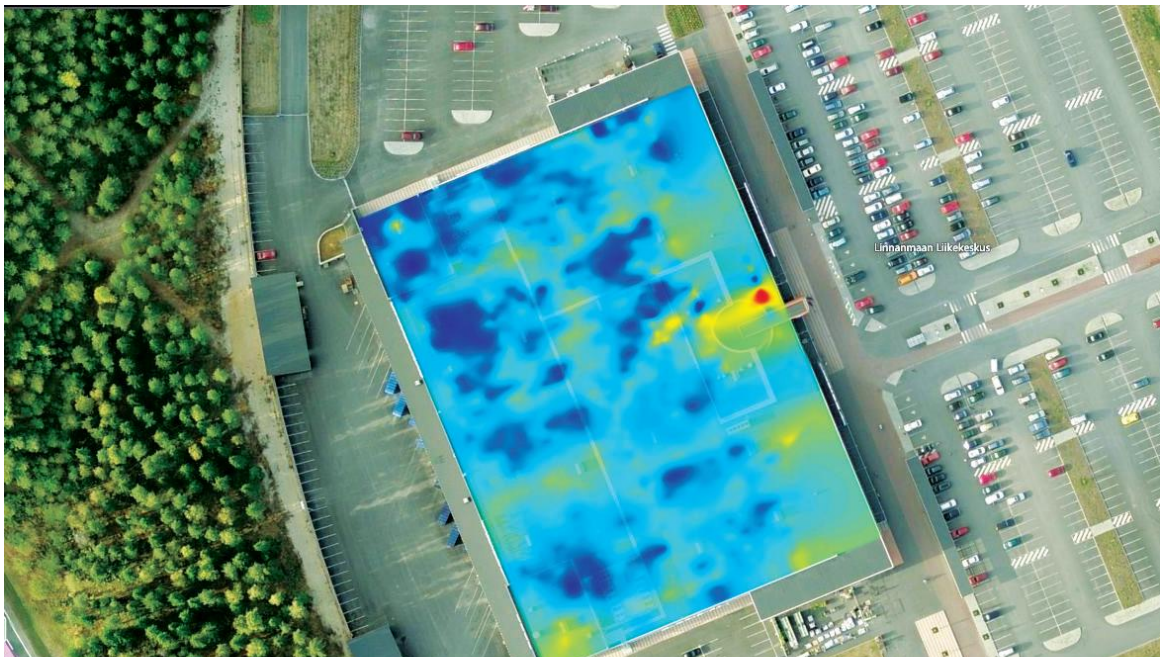
Εικόνα 2: Η εφαρμογή πλοήγησης εσωτερικών χώρων της "infsoft"

[3] https://www.infsoft.com/solutions/indoor-navigation

IndoorAtlas[4]

Ένα πολύ ενδιαφέρον εγχείρημα όπου η τεχνολογία που χρησιμοποιείται για τον εντοπισμό της θέσης μέσα σε ένα κτήριο δεν έχει να κάνει αυτή τη φορά με αισθητήρες τοποθετημένους στους χώρους του. Αντίθετα, για τη χαρτογράφηση του χώρου χρησιμοποιείται το μαγνητικό πεδίο του κτηρίου.[5] Τα σύγχρονα κτήρια διαθέτουν ένα μοναδικό μαγνητικό τοπίο το οποίο παράγεται από το μαγνητικό πεδίο της γης που αλληλεπιδρά με τον χάλυβα και άλλα υλικά από τα οποία είναι κατασκευασμένα τα κτήρια. Χρησιμοποιώντας τον ενσωματωμένο μαγνητικό αισθητήρα (πυξίδα) καθώς και άλλες τεχνολογίες ανίχνευσης μέσα σε ένα έξυπνο τηλέφωνο (smartphone), το λογισμικό μπορεί να χρησιμοποιήσει το μαγνητικό πεδίο εντός του κτηρίου ως χάρτη για ακριβή εντοπισμό και παρακολούθηση της θέσης ενός ατόμου σε εσωτερικό χώρο, αποτυπώνοντάς τη στο χάρτη (Εικόνα 3). Μπορούμε επίσης να γνωρίζουμε το σημείο εκκίνησης, το μονοπάτι που έχει πάρει κάποιος και το τελικό του σημείο. Αυτό γίνεται σε πραγματικό χρόνο - ακριβώς όπως το εξωτερικό GPS.

Η εταιρία διαθέτει προς πώληση είτε έτοιμη την εφαρμογή κατά παραγγελία, είτε το API της συγκεκριμένης τεχνολογίας για την χρήση του σε εφαρμογές πλοήγησης, ενώ υπόσχεται μεγάλη ακρίβεια εντοπισμού με σφάλμα που δεν ξεπερνά τα 2 μέτρα. Το βασικό πλεονέκτημα της λύσης αυτής είναι ότι, πέραν της πυξίδας που πρέπει να διαθέτει το τηλέφωνό μας, δεν απαιτούνται πρόσθετες συσκευές αισθητήρων σε κάθε χώρο. Φυσικά η ακρίβεια μπορεί να



Εικόνα 3: Χαρτογράφηση εσωτερικού κτηρίου μέσω του μαγνητικού του πεδίου

[4] http://www.indooratlas.com/

[5] http://www.indooratlas.com/wp-content/uploads/2016/03/magnetic_positioning_opus_jun2014.pdf

αυξηθεί περεταίρω αν χρησιμοποιηθούν συνδυαστικά και αισθητήρες ανίχνευσης προσέγγισης όπως και στις προηγούμενες υλοποιήσεις.

1.4 Συνεισφορά της Προτεινόμενης Εργασίας

Όλες οι εμπορικές εφαρμογές υπόσχονται μεγάλη ακρίβεια εντοπισμού με ελάχιστους χρόνους απόκρισης (latency), ενώ δεσπόζουσα θέση κατέχει το σύγχρονο γραφικό τους περιβάλλον χρήσης. Στόχος της δικής μας υλοποίησής δεν θα είναι να ανταγωνιστούμε σε κανέναν τομέα τις ήδη υπάρχουσες προτάσεις. Απεναντίας θα προσπαθήσουμε χρησιμοποιώντας γνωστές τεχνολογίες ιστού ανοικτού λογισμικού (PHP, HTML, JavaScript, jQuery), αλλά και υπηρεσίες που μας παρέχει το Υπολογιστικό Νέφος (δημόσια πλατφόρμα FIWARE μέσω του ανοιχτού λογισμικού OpenStack), να επιδείξουμε πως αυτές μπορούν να χρησιμοποιηθούν για την επίλυση του προβλήματος της πλοήγησης σε εσωτερικούς χώρους.

Ιδιαίτερη έμφαση στην εκδοχή μας θα δοθεί στο σχεδιασμό υπηρεσιοκεντρικής αρχιτεκτονικής, όπου δηλαδή κάθε λειτουργία της εφαρμογής μας είναι σχεδιασμένη ως μια αυτόνομη υπηρεσία στο ΥΝ, με την καθεμιά ωστόσο να επικοινωνεί με τις υπόλοιπες χρησιμοποιώντας τις αρχές του REST προτύπου για την ανάπτυξη υπηρεσιών ιστού. Εκμεταλλευόμενοι τα οφέλη που η παραπάνω αρχιτεκτονική μας προσφέρει θα επικεντρωθούμε, μέσα από την δικτυακή μας εφαρμογή και σε λειτουργίες διαχείρισης, συμπεριλαμβανομένης της παρακολούθησης της δραστηριότητας των επισκεπτών της υποδομής, ενώ ο κάθε επισκέπτης μεμονωμένα θα αντιμετωπίζεται ως ξεχωριστή οντότητα, επιτρέποντάς του την πρόσβαση στους χώρους όπου εμείς ορίζουμε.

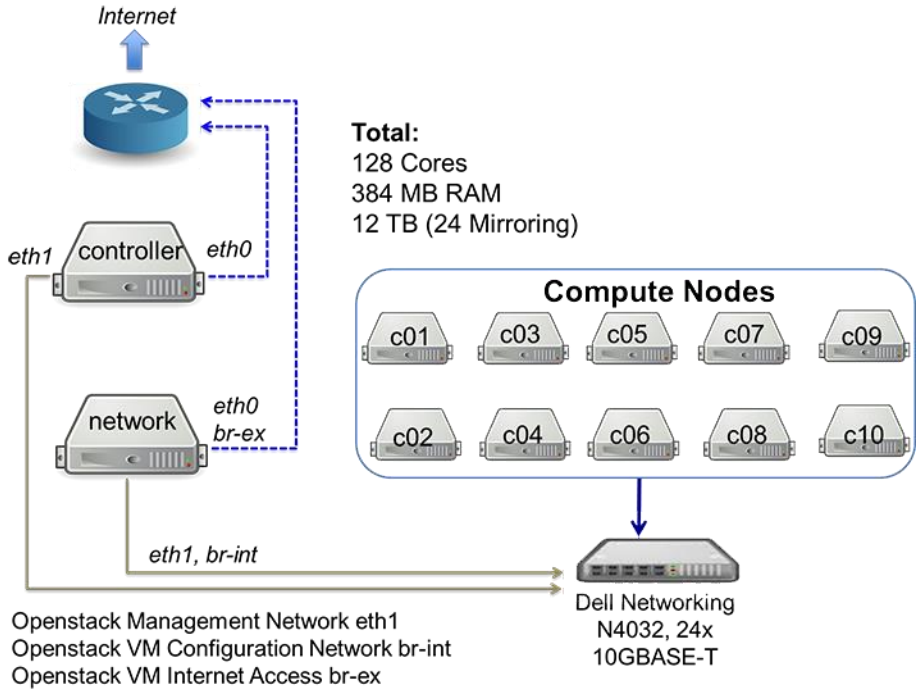
1.5 Περιβάλλον Εργασίας

Η υλοποίηση της εργασίας πραγματοποιήθηκε σε περιβάλλον OpenStack[6] και FIWARE. Οι υπηρεσίες που χρησιμοποιήθηκαν προέρχονται από τον δημόσιο πάροχο FIWARE Lab[7] - που προσφέρει υπηρεσίες νέφους σε χρήστες σε όλη την Ευρώπη - καθώς και από το ιδιωτικό νέφος Intellicloud, που έχει αναπτυχθεί για πειραματικούς σκοπούς από το Εργαστήριο Προγραμματισμού και Ευφυών συστημάτων του Πολυτεχνείου Κρήτης. Για την επικοινωνία των αιτημάτων μέσω διαδικτύου χρησιμοποιήθηκαν κλήσεις REST[8] μέσω του πρωτοκόλλου HTTP[9].

Η υποδομή Υπολογιστικού Νέφους Intellicloud, σχεδιάστηκε και υλοποιήθηκε από το εργαστήριο Ευφυών Συστημάτων του τμήματος ΗΜΜΥ του Πολυτεχνείου Κρήτης, όπου και

[6] https://www.openstack.org/
[7] https://www.fiware.org/lab/
[8] https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm
[9] https://tools.ietf.org/html/rfc2616

φιλοξενείται μέχρι και σήμερα. Σκοπός της είναι η παροχή υπολογιστικών πόρων, για την ανάπτυξη εφαρμογών στο Νέφος. Η υποδομή περιλαμβάνει 128 πυρήνες επεξεργαστικής ισχύος, 384 GB RAM και 12 TB σε σκληρό δίσκο.



Εικόνα 4: Υποδομή Intellicloud Πολυτεχνείου Κρήτης

1.6 Δομή της Εργασίας

- Στο δεύτερο κεφάλαιο γίνεται σύντομη περιγραφή κάποιων γνωστών τεχνολογιών ιστού που θα χρησιμοποιηθούν, ενώ στη συνέχεια ορίζεται η έννοια του ΥΝ και αναλύονται οι βασικές αρχές του.
- Στο τρίτο κεφάλαιο ασχολούμαστε με το σχεδιασμό του συστήματος και την ανάλυση των απαιτήσεων του, με τη βοήθεια διαγραμμάτων της Γλώσσας Μοντελοποίησης Λογισμικού UML[10].
- Στο τέταρτο κεφάλαιο αναλύουμε τις υπηρεσίες που απαρτίζουν τα υποσυστήματα χρήστη (front-end system) και ΥΝ (back-end system) της εφαρμογής μας. Εξετάζουμε την επικοινωνία μεταξύ των υπηρεσιών και παραθέτουμε στιγμιότυπα πραγματικής χρήσης της εφαρμογής.
- Στο πέμπτο κεφάλαιο εξετάζουμε την απόδοση του συστήματός μας υπό συνθήκες υψηλού φόρτου. Διατυπώνουμε κατόπιν τα τελικά μας συμπεράσματα, σχολιάζουμε τις αδυναμίες υλοποίησης και προτείνουμε ιδέες για μελλοντικές βελτιώσεις και επεκτάσεις της εφαρμογής.

[10] http://www.uml.org/what-is-uml.htm

Κεφάλαιο 2 - Υπόβαθρο και Υπάρχουσες Τεχνολογίες

2.1 Υπηρεσιοκεντρική Αρχιτεκτονική

Μία υπηρεσία μπορεί να οριστεί ως ένα επαναχρησιμοποιήσιμο συστατικό στοιχείο λογισμικού με χαμηλό βαθμό σύζευξης, το οποίο ενθυλακώνει διακριτή λειτουργικότητα και μπορεί να είναι κατανεμημένο και προσπελάσιμο από άλλα προγράμματα. Η σύζευξη (coupling) αναφέρεται στο βαθμό εξάρτησης μεταξύ δύο συστατικών λογισμικού. Η εξάρτηση μπορεί να οφείλεται σε πολλούς λόγους, όπως η κλήση μεθόδων (συναρτήσεων) του ενός συστατικού από το άλλο, το "πέρασμα" δεδομένων από το ένα συστατικό στο άλλο ή η χρήση συστατικών ενός τύπου ως ιδιότητες (μεταβλητές) σε κάποιο άλλο. Δύο μονάδες λογισμικού που εξαρτώνται σε μεγάλο βαθμό μεταξύ τους λέγεται ότι έχουν ισχυρή σύζευξη, ενώ αν η εξάρτηση είναι ασθενής πρόκειται για χαλαρή σύζευξη. Μία υπηρεσία Ιστού προσπελάζεται με τη χρήση τυποποιημένων πρωτοκόλλων τα οποία βασίζονται στο Διαδίκτυο. Η κρίσιμη διαφορά μεταξύ υπηρεσίας και συστατικού στοιχείου λογισμικού όπως ορίζεται στη CBSE[11], είναι ότι οι υπηρεσίες είναι ανεξάρτητες, καθώς δεν έχουν «απαιτούμενες» διασυνδέσεις, ενώ η επικοινωνία τους βασίζεται στην ανταλλαγή μηνυμάτων, εκφρασμένων σε XML.

Η Υπηρεσιοκεντρική Αρχιτεκτονική (Service Oriented Architecture - SOA) αποτελεί ένα τρόπο ανάπτυξης κατανεμημένων συστημάτων σύμφωνα με τον οποίο τα συστατικά στοιχεία κάθε συστήματος είναι αυτόνομες υπηρεσίες. Οι υπηρεσίες μπορεί να εκτελούνται σε απομακρυσμένους υπολογιστές από πολλούς διαφορετικούς παρόχους υπηρεσιών. Ορισμένα από τα οφέλη των Υπηρεσιοκεντρικών Αρχιτεκτονικών είναι τα ακόλουθα:

- **Ευελιξία για συνεργασία:** Η SOA παρέχει την ικανότητα ασφαλούς και εύκολου διαμοιρασμού πληροφοριών μεταξύ συνεργατών και ενδιαφερομένων (stakeholders) παρέχοντας μια βασική υπηρεσία την οποία κάθε εξουσιοδοτημένος επιχειρηματικός συνεργάτης μπορεί να χρησιμοποιήσει, επεκτείνοντας έτσι την επιχείρηση.
- **Ευελιξία για προσαρμογή:** Η SOA προάγει την ικανότητα ταχύρρυθμης επαναδιαμόρφωσης των επιχειρηματικών διαδικασιών επιλέγοντας διαφορετικές κάθε φορά υπηρεσίες από ένα διαθέσιμο σύνολο υπηρεσιών και έτσι παρέχει τη δυνατότητα προσαρμογής της επιχείρησης στις νέες επιχειρηματικές απαιτήσεις που θα προκύψουν.
- **Μείωση του κόστους:** Η SOA βασίζεται σε πρότυπα όπως για παράδειγμα το WSDL, SAML, SOAP, UDDI, τα οποία παρέχουν μια εξαρτηματική (που βασίζεται σε υπηρεσίες-εξαρτήματα) προσέγγιση της αρχιτεκτονικής λογισμικού και δίνει έτσι τη δυνατότητα διαμοιρασμού και επαναχρησιμοποίησης υπηρεσιών.
- **Βελτίωση της αποδοτικότητας:** Η προαναφερθείσα εξαρτηματική προσέγγιση της SOA εξασφαλίζει υψηλό βαθμό επαναχρησιμοποίησης των επιχειρηματικών υπηρεσιών. Η καταγραφή και κωδικοποίηση των υπηρεσιών σε κατάλληλους πίνακες

[11] https://en.wikipedia.org/wiki/Component-based_software_engineering

αποθετήρια επιτρέπει τη διασφάλιση της συνέπειας, δηλαδή την αποφυγή ανάπτυξης δυο ή περισσότερων υπηρεσιών για το ίδιο πράγμα.

2.1.1 Το πρωτόκολλο SOAP

Το SOAP είναι ένα ελαφρύ πρωτόκολλο, το οποίο προορίζεται για την ανταλλαγή δομημένων πληροφοριών σε ένα κατανεμημένο περιβάλλον. Χρησιμοποιεί τεχνολογίες XML, για να καθορίσει ένα επεκτάσιμο πλαίσιο ανταλλαγής μηνυμάτων, το οποίο παρέχει μια δομή μηνυμάτων που μπορεί να ανταλλαχθεί πάνω από ποικίλα δικτυακά πρωτόκολλα. Το πλαίσιο έχει σχεδιαστεί για να είναι ανεξάρτητο από οποιοδήποτε μοντέλο προγραμματισμού και σημασιολογίας υλοποίησης. Καθορίζει ένα σύνολο κανόνων κωδικοποίησης για τα δεδομένα και μια σύμβαση για την παραγωγή απομακρυσμένων κλήσεων διαδικασίας (Remote Procedure Call/RPC). Αποτελεί τη βάση για ένα ευρύ φάσμα πρωτοκόλλων, που τρέχουν πάνω από άλλα πρωτόκολλα, όπως πάνω από το HTTP. Ένα μμήνυμα SOAP είναι βασισμένο στην XML και περιέχει τα ακόλουθα μέρη (Εικόνα 5):

- Το **Envelope**, το ανώτερο στρώμα, το οποίο αντιπροσωπεύει το μήνυμα.
- Το **Header**, το στρώμα για τα προστιθέμενα δεδομένα σε ένα μήνυμα SOAP. Το SOAP καθορίζει τις ιδιότητες, ώστε να μπορεί να προσδιορίσει ποιος θα πρέπει να ασχοληθεί με ένα συγκεκριμένο στοιχείο και εάν η κατανόησή του είναι προαιρετική ή υποχρεωτική.
- Το **Body**, το στρώμα που περιλαμβάνει τις υποχρεωτικές πληροφορίες, οι οποίες προορίζονται για το δέκτη των μηνυμάτων.

Εικόνα 5: Δομή μηνυμάτων στο SOAP

2.1.2 Γλώσσα Περιγραφής Υπηρεσιών Ιστού (WSDL)

Ένα αρχείο WSDL είναι ένα έγγραφο XML στο οποίο περιγράφονται ένα σύνολο μηνυμάτων SOAP και ο τρόπος ανταλλαγής αυτών των μηνυμάτων. Ένα WSDL καθορίζει τί πρέπει να περιέχει ένα μήνυμα αίτησης και πώς θα μοιάζει το μήνυμα απόκρισης με σαφή σημειογραφία. Η σημειογραφία που χρησιμοποιεί ένα αρχείο WSDL για την περιγραφή μορφών μηνύματος βασίζεται αφενός στο πρότυπο XML Schema, δηλαδή είναι ουδέτερη των προγραμματιστικών γλωσσών και αφετέρου σε πρότυπα τα οποία την καθιστούν κατάλληλη για την περιγραφή διεπαφών των Υπηρεσιών Ιστού - ΥΙ (Web Services – WS), που είναι προσπελάσιμές από ένα μεγάλο σύνολο πλατφορμών και προγραμματιστικών γλωσσών. Η WSDL καθορίζει πού βρίσκεται διαθέσιμη η υπηρεσία και ποιο πρωτόκολλο επικοινωνίας χρησιμοποιείται για να μιλήσει στην υπηρεσία. Το αρχείο WSDL καθορίζει ό,τι χρειάζεται ένα πρόγραμμα για να συνεργαστεί με μια ΥΙ. Υπάρχουν διαθέσιμα διάφορα εργαλεία για την ανάγνωση ενός αρχείου WSDL και την παραγωγή του απαραίτητου κώδικα για την επικοινωνία με μια ΥΙ. Υπάρχουν πολλά εργαλεία για το SOAP που επιτρέπουν την παραγωγή αρχείων WSDL από υπάρχουσες διεπαφές προγραμμάτων, αλλά όμως υπάρχουν λίγα εργαλεία για την άμεση συγγραφή μιας WSDL. Έτσι, το έγγραφο WSDL είναι ένα σχήμα XML για την περιγραφή ΥΙ, ως ένα σύνολο από τελικά σημεία, τα οποία λειτουργούν με ανταλλαγή μηνυμάτων, που περιέχουν πληροφορίες προσανατολισμένες είτε στα έγγραφα είτε στις διαδικασίες. Ένα έγγραφο WSDL χρησιμοποιείται για να περιγράψει τι μπορεί να κάνει μια ΥΙ, πού βρίσκεται και πώς μπορεί κάποιος να το καλέσει. Τα βασικότερα στοιχεία ενός εγγράφου WSDL είναι τα εξής:

- <portType>: οι λειτουργίες της ΥΙ,
- <message>: τα μηνύματα που χρησιμοποιούνται από την ΥΙ,
- <types>: οι τύποι δεδομένων που χρησιμοποιούνται από την ΥΙ,
- <binding>: τα πρωτόκολλα επικοινωνίας που χρησιμοποιούνται από την ΥΙ. Καθορίζει συνήθως τη σύναψη του HTTP με το SOAP.

2.1.3 Υπηρεσίες ιστού RESTful

Οι RESTful ΥΙ είναι κατασκευασμένες για να δουλεύουν πιο γρήγορα και πιο αποτελεσματικά στον παγκόσμιο ιστό. Το Representational State Transfer (REST) επιφέρει επιθυμητές ιδιότητες, όπως επίδοση (performance), επεκτασιμότητα (scalability) και τροποποιησιμότητα (modifiability), ιδιότητες που δίνουν τη δυνατότητα στις ΥΙ να δουλεύουν βέλτιστα στον παγκόσμιο ιστό. Στο αρχιτεκτονικό πρότυπο REST, τα δεδομένα και η λειτουργικότητα θεωρούνται πόροι (resources) και είναι προσβάσιμα με τη χρήση των Uniform Resource Identifiers (URI), τα οποία δεν είναι τίποτε άλλο από υπερσύνδεσμοι του διαδικτύου. Αυτοί οι πόροι ενεργοποιούνται χρησιμοποιώντας ένα σύνολο από απλές και καλά καθορισμένες λειτουργίες.

Το REST υποστηρίζει την αρχιτεκτονική πελάτη-εξυπηρετητή και είναι σχεδιασμένο να χρησιμοποιεί ένα πρωτόκολλο επικοινωνίας που δεν κρατά την κατάστασή του (stateless), όπως το HTTP, το οποίο είναι και το σύνηθες. Στο REST, οι πελάτες και οι εξυπηρετητές ανταλλάσσουν αναπαραστάσεις πόρων (resource representation), χρησιμοποιώντας ένα προτυποποιημένο περιβάλλον διεπαφής και πρωτοκόλλου. Οι εφαρμογές που βασίζονται στις υπηρεσίες REST είναι απλές, ελαφριές και γρήγορες, χάρη στις εξής ιδιότητες:

Χρήση των μεθόδων του πρωτοκόλλου HTTP για την αποστολή αιτημάτων

Οι πόροι ελέγχονται χρησιμοποιώντας τέσσερις λειτουργίες: τη δημιουργία (PUT), την ανάγνωση (GET), την ενημέρωση (POST) και τη διαγραφή (DELETE) ενός πόρου. Η PUT δημιουργεί έναν νέο πόρο, ο οποίος μπορεί να διαγραφεί με την DELETE. Η GET ανακτά την τρέχουσα κατάσταση ενός πόρου σε μερική αναπαράσταση, ενώ η POST μεταφέρει σε έναν πόρο μια νέα κατάσταση.

Μη διατήρηση προηγούμενων καταστάσεων (Service statelessness)

Κάθε αλληλεπίδραση με έναν πόρο είναι stateless. Αυτό σημαίνει ότι τα μηνύματα αιτήσεων είναι ανεξάρτητα. Οι αλληλεπιδράσεις stateful βασίζονται στην ιδέα της μεταφοράς κατάστασης. Υπάρχουν αρκετές τεχνικές για την ανταλλαγή κατάστασης, όπως η επανεγγραφή του URI (URI rewriting), τα cookies και η απόκρυψη από πεδία φορμών. Η κατάσταση μπορεί να ενσωματωθεί σε μηνύματα απόκρισης, ώστε αυτά να δείχνουν σε έγκυρες μελλοντικές καταστάσεις της αλληλεπίδρασης.

Απλοποίηση διευθύνσεων ανάκτησης πόρων (URI)

Λόγω του μεγάλου πλήθους πόρων που συνήθως φιλοξενούνται σε ένα διακομιστή, συχνά για την προσπέλασή τους απαιτείται χρήση περίπλοκων και μεγάλης έκτασης URI. Μέσω της REST αρχιτεκτονικής αποσκοπούμε σε ένα σχεδιασμό κατά τον οποίο τα URI θα αντιστοιχούν σε μια μορφή ιεραρχίας. Πιο συγκεκριμένα, σε περίπτωση που θέλουμε να ζητήσουμε μία ιδιότητα ενός πόρου, ξεκινάμε από την κατηγορία του, στη συνέχεια μεταβαίνουμε στο όνομα του και τέλος στην ιδιότητα που θέλουμε να ανακτήσουμε. Παρότι ο παραπάνω ιεραρχικός σχεδιασμός, δεν είναι δεσμευτικός, καλό θα είναι να λαμβάνεται υπόψιν κατά τον σχεδιασμό μεγάλων διαδικτυακών εφαρμογών με σκοπό να εξασφαλίζεται η σαφήνεια και μοναδικότητα κατά των διαχωρισμό των πόρων.

Μεταφορά δεδομένων μέσω JSON και XML

Οι σχεδιαστές σύγχρονων REST API τείνουν να επιλέγουν ένα από τα δύο μορφότυπα (formats) για την ανταλλαγή δεδομένων μεταξύ των διακομιστών τους και των προγραμμάτων πελατών - XML ή JSON[12]. Αν και έχουν σχεδιαστεί και προωθηθεί πολλά διαφορετικά μορφότυπα δεδομένων, οι ενσωματωμένες ιδιότητες επικύρωσης της XML, αλλά και η

[12] http://www.json.org/

ευελιξία του JSON βοήθησαν τα δύο παραπάνω να ξεχωρίσουν και να αναδειχθούν ως ηγέτες στο χώρο των API.

Τι είναι το JSON;

Το JSON ή JavaScript Object Notation, είναι μια ελάχιστη, αναγνώσιμη μορφή για να δομούνται δεδομένα. Χρησιμοποιείται κυρίως για τη μετάδοση δεδομένων μεταξύ ενός διακομιστή και μιας εφαρμογής Ιστού, ως εναλλακτική λύση της XML.

Κλειδιά και τιμές

Τα δύο βασικά μέρη που συνθέτουν το JSON είναι κλειδιά και τιμές. Μαζί κάνουν ένα ζεύγος κλειδιών / τιμών.

- **Κλειδί**: Ένα κλειδί είναι πάντα μια συμβολοσειρά που περικλείεται σε εισαγωγικά.
- **Τιμή**: Μια τιμή μπορεί να είναι μια συμβολοσειρά, ένας αριθμός, μια έκφραση boolean, ένας πίνακας ή ένα αντικείμενο.
- **Ζεύγος κλειδιού / τιμής**: Ένα ζεύγος βασικών τιμών ακολουθεί μια συγκεκριμένη σύνταξη, με το κλειδί να ακολουθείται από μία τελεία ":" και ακολουθούμενο από την τιμή. Τα ζεύγη κλειδιού / τιμής διαχωρίζονται με κόμμα.

Ας πάρουμε μια γραμμή από το δείγμα JSON παραπάνω και ας προσδιορίσουμε κάθε τμήμα του κώδικα:

```
"foo": "bar"
```

Αυτό το παράδειγμα είναι ένα ζεύγος κλειδιού / τιμής. Το κλειδί είναι "foo" και η τιμή είναι "bar".

2.2 Υπολογιστικό Νέφος

Υπολογιστικό Νέφος[13] (Cloud Computing) ονομάζεται η κατ' αίτηση διαδικτυακή κεντρική διάθεση υπολογιστικών πόρων (όπως δίκτυο, εξυπηρετητές, εφαρμογές και υπηρεσίες) με υψηλή ευελιξία, ελάχιστη προσπάθεια από τον χρήστη και υψηλή αυτοματοποίηση. Στο Υπολογιστικό Νέφος η αποθήκευση, η επεξεργασία και η χρήση δεδομένων, λογισμικού και υπηρεσιών γίνεται διαδικτυακά, μέσω απομακρυσμένων υπολογιστών σε κεντρικούς Σταθμούς Εξυπηρετητών[14] (Datacenter). Υπηρεσίες όπως το διαδικτυακό ηλεκτρονικό ταχυδρομείο ή τα κοινωνικά δίκτυα συχνά βασίζονται στην τεχνολογία του Υπολογιστικού Νέφους. Οι χρήστες εξοικονομούν πόρους από την αγορά και

[13] http://www.nist.gov/itl/csd/cloud-102511.cfm
[14] http://searchdatacenter.techtarget.com/definition/data-center

συντήρηση λογισμικού, τη συντήρηση ακριβών εξυπηρετητών και εγκαταστάσεων αποθήκευσης δεδομένων.

2.2.1 Σύντομη Ιστορική Αναδρομή

Το ΥΝ έχει τις ρίζες του στη δεκαετία του 1950 και τη χρήση των Κεντρικών Υπολογιστών (mainframes). Πολλαπλοί χρήστες μπορούσαν να έχουν πρόσβαση σε έναν Κεντρικό Υπολογιστή μέσω απλών τερματικών, των οποίων η μόνη λειτουργία ήταν να παρέχουν πρόσβαση σε αυτόν. Ωστόσο, λόγω του μεγάλου κόστους αγοράς, αλλά και συντήρησης ενός Κεντρικού Υπολογιστή, οι χρήστες ενός οργανισμού έκαναν κοινή χρήση των υπολογιστικών του πόρων. Τη δεκαετία του 1970, με την έλευση του λειτουργικού συστήματος VM από την IBM, δημιουργήθηκαν ουσιαστικά οι πρώτες εικονικές μηχανές (virtual machines). Με τη χρήση λογισμικού εικονοποίησης κατέστη εφικτή η ταυτόχρονη εκτέλεση περισσότερων του ενός λειτουργικών συστημάτων στο ίδιο φυσικό μηχάνημα.

Τη δεκαετία του 2000, η Amazon έπαιξε καθοριστικό ρόλο στην ανάπτυξη του ΥΝ με τον εκσυγχρονισμό των Datacenter της, ενώ το 2006 εγκαινίασε το Amazon Web Services (AWS) και το Amazon Elastic Compute Cloud (EC2), το οποίο επιτρέπει σε μικρές επιχειρήσεις και ιδιώτες να νοικιάζουν υπολογιστικούς πόρους. Στις αρχές του 2008, το Eucalyptus έγινε η πρώτη συμβατή πλατφόρμα με την προγραμματιστική διεπαφή του AWS για την ανάπτυξη των ιδιωτικών υπολογιστικών νεφών, ενώ το OpenNebula, έγινε το πρώτο λογισμικό ανοιχτού κώδικα για την ανάπτυξη ιδιωτικών και υβριδικών υπολογιστικών νεφών. *(βλ. § 2.2.3)*

Στις μέρες μας, η εξέλιξη του διαδικτύου, με τις υπερύψηλές ταχύτητες που πλέον αυτό προσφέρει, το Διαδίκτυο των Πραγμάτων[15] (Internet of Things), αλλά και η καθιέρωση Υπηρεσιοκεντρικών Αρχιτεκτονικών (Service Oriented Architecture[16] - SOA) από όλο και περισσότερους οργανισμούς οδηγεί στην περαιτέρω ανάπτυξη του ΥΝ.

2.2.2 Πλεονεκτήματα και Μειονεκτήματα του Υπολογιστικού Νέφους

Το υπολογιστικό νέφος έχει ενσωματωθεί σε μεγάλο βαθμό από πολλές υπηρεσίες και οργανισμούς, καθώς προσφέρει πολλά πλεονεκτήματα. Θα αναφέρουμε ενδεικτικά τα πιο σημαντικά:

[15] http://www.businessinsider.com/what-is-the-internet-of-things-definition-2016-8
[16] http://www.service-architecture.com/articles/web-services/service-oriented_architecture_soa_definition.html

Αυτοεξυπηρέτηση κατ' απαίτηση: Ο χρήστης μπορεί όποτε θέλει να χρησιμοποιήσει μια υπηρεσία ή να δεσμεύει υπολογιστικούς πόρους, όπως μονάδες επεξεργασίας και αποθήκευσης, χωρίς την ενδιάμεση αλληλεπίδραση με τον πάροχο των υπηρεσιών.

Ευρεία πρόσβαση στο δίκτυο: Οι υπολογιστικοί πόροι είναι διαθέσιμοι μέσω του δικτύου και η πρόσβαση γίνεται μέσω οποιασδήποτε συσκευής μπορεί να συνδεθεί σε αυτό μέσω ενός προγράμματος περιήγησης (τηλέφωνα, ταμπλέτες, υπολογιστές).

Ταχεία ελαστικότητα: Οι υπολογιστικοί πόροι μπορούν να δεσμεύονται και να αποδεσμεύονται ελαστικά και σε ορισμένες περιπτώσεις, αυτόματα, ώστε το υπολογιστικό νέφος να κλιμακώνεται ανάλογα με τη ζήτηση. Ο χρήστης έχει την αίσθηση ότι οι υπολογιστικοί πόροι είναι απεριόριστοι και μπορούν να διατεθούν σε οποιαδήποτε ποσότητα κι ανά πάσα στιγμή.

Κλιμακωσιμότητα: Στο ΥΝ είναι δυνατή η χειροκίνητη ή δυναμική προσθήκη και αφαίρεση κόμβων επεξεργασίας, εκτέλεσης εργασιών ή αποθήκευσης, ανάλογα με την αυξομείωση των απαιτήσεων, χωρίς να αλλοιώνεται η υπηρεσία που παρέχεται στο χρήστη.

Εικονοποίηση: Οι λεπτομέρειες υλοποίησης και κατάστασης στην οποία βρίσκονται οι υπολογιστικοί πόροι αποκρύπτονται από το χρήστη. Οι Εικονικές Μηχανές δίνουν την εντύπωση στο χρήστη ενός ολοκληρωμένου φυσικού μηχανήματος, ενώ στην πραγματικότητα, είναι ένα σύνολο αρχείων και προγραμμάτων που τρέχουν πάνω σε ένα άλλο (ή άλλα) φυσικά μηχανήματα. Οι Εικονικές Μηχανές μπορεί να έχουν διαφορετικά χαρακτηριστικά από τους φυσικούς πόρους πάνω στους οποίους τρέχουν, τόσο όσον αφορά το λογισμικό, όσο και για το υλικό.

Μετρούμενη υπηρεσία: Τα συστήματα υπολογιστικού νέφους ελέγχονται και βελτιστοποιούνται αυτόματα, αξιοποιώντας ένα σύστημα μέτρησης (συνήθως pay as you go). Η χρήση των πόρων μπορεί να παρακολουθείται, να ελέγχεται, και να γίνεται αναφορά, παρέχοντας διαφάνεια τόσο στον πάροχο όσο και στο χρήστη για την υπηρεσία που χρησιμοποιείται.

Αξιοπιστία: Το υπολογιστικό νέφος εγγυάται την ορθή λειτουργία των προγραμμάτων των χρηστών, την αποθήκευση των δεδομένων τους και την αξιόπιστη μεταφορά τους. Για την εξασφάλιση της αξιοπιστίας, τα δεδομένα αποθηκεύονται σε περισσότερους από έναν κόμβους στο νέφος. Ο συνηθισμένος αριθμός αντιγράφων είναι 3 (replication level three). Επίσης, εφόσον τα δεδομένα του χρήστη βρίσκονται σε κάποια μηχανήματα σε ένα Datacenter, μία βλάβη στο τερματικό του χρήστη δεν θα έχει καμία επίδραση στα δεδομένα αυτά.

Συντήρηση: Η συντήρηση στο ΥΝ, με την έννοια της αποσφαλμάτωσης των εφαρμογών και της αναβάθμισης των εκδόσεών τους, δεν είναι πλέον ευθύνη του χρήστη. Ο πάροχος φροντίζει για την ορθή και ομαλή λειτουργία των υπηρεσιών που προσφέρονται στο ΥΝ, ο οποίος πλέον δεν χρειάζεται να ασχολείται με κάθε χρήστη ξεχωριστά, αφού είναι δυνατό να βελτιώσει την εμπειρία χρήσης σε όλους τους χρήστες ταυτόχρονα. Επίσης, οφείλει να προσφέρει συμβατό

λογισμικό ανεξάρτητα από τη συσκευή-τερματικό (τηλέφωνα, ταμπλέτες, υπολογιστές) που θα επιλέξει ο χρήστης για να τρέξει μια εφαρμογή.

Ωστόσο αξίζει να αναφερθούμε και στα βασικότερα μειονεκτήματα που προκύπτουν από τη χρήση του υπολογιστικού νέφους. Πιο συγκεκριμένα:

Ιδιωτικότητα των δεδομένων: Αποτελεί και την μεγαλύτερη ανησυχία όταν γίνεται χρήση υπηρεσιών στο ΥΝ. Αν και τα δεδομένα των χρηστών είναι σχεδόν πάντοτε κρυπτογραφημένα και αδύνατον να υποκλαπούν, κανένας ωστόσο δεν εγγυάται την μη εκμετάλλευσή τους από τον ίδιο τον πάροχο. Φυσικά οι μεγάλοι πάροχοι διαδικτυακών υπηρεσιών (Google, Amazon κλπ.) είναι αξιόπιστοι και δεσμεύονται για την ιδιωτικότητα των δεδομένων των χρηστών. Θεωρητικά όμως, από τη στιγμή που μοιραζόμαστε τα δεδομένα μας με κάποιον τρίτο, ποτέ δεν μπορούμε να είμαστε απόλυτα σίγουροι.

Νομικά θέματα: Η χρήση του υπολογιστικού νέφους γίνεται έπειτα από σύναψη Συμφωνητικών Παροχής Υπηρεσιών (Service Level Agreement - SLA). Τέτοια συμφωνητικά είναι πολλές φορές ιδιαίτερα δεσμευτικά και μπορούν να «παγιδέψουν» τον χρήστη εν αγνοία του, με υψηλές χρεώσεις, αδυναμία αλλαγής παρόχου κλπ. Η εκτενής ανάγνωση και πλήρης κατανόηση όλων των όρων χρήσης είναι απαραίτητη πριν από τη σύναψη συμφωνίας με οποιονδήποτε πάροχο.

Αλλαγή παρόχου και μεταφορά των δεδομένων: Σε πολλές περιπτώσεις δεν είναι εύκολη η μεταφορά των δεδομένων από το ένα υπολογιστικό νέφος στο άλλο. Αυτό οφείλεται κυρίως σε θέματα τεχνικής φύσεως που αφορούν το περιβάλλον εικονοποίησης του εκάστοτε παρόχου, και της ασυμβατότητάς του με άλλα περιβάλλοντα. Κάτι τέτοιο οδηγεί πολλές φορές τον καταναλωτή να παραμείνει σε ένα υπολογιστικό νέφος παρόλο που αυτό δεν τον καλύπτει πλήρως.

2.2.3 Μοντέλα Υπολογιστικών Νεφών

Δημόσιο νέφος (Public cloud): Τα δημόσια νέφη γίνονται διαθέσιμα στο κοινό από έναν πάροχο που φιλοξενεί την υποδομή. Οι πάροχοι δημόσιων νεφών όπως οι Amazon AWS, Microsoft και Google κατέχουν και χειρίζονται την υποδομή και προσφέρουν πρόσβαση μέσω διαδικτύου. Από τη μεριά τους οι χρήστες δεν έχουν γνώση ή έλεγχο για το που βρίσκονται τα μηχανήματα που φιλοξενούν το cloud. Στα δημόσια νέφη, όλοι οι χρήστες μοιράζονται τους ίδιους υπολογιστικούς πόρους και έχουν περιορισμένη παραμετροποίηση και ασφάλεια.

Ιδιωτικό νέφος (Private cloud): Ο όρος αναφέρεται σε υποδομή νέφους που προορίζεται για έναν συγκεκριμένο οργανισμό. Τα ιδιωτικά νέφη επιτρέπουν σε επιχειρήσεις να φιλοξενούν εφαρμογές στο cloud και παράλληλα να διευθετούν οι ίδιες θέματα που αφορούν την ασφάλεια και των έλεγχο δεδομένων. Να σημειωθεί ότι υπάρχει περίπτωση η δημιουργία ενός ιδιωτικού

νέφους να ανατεθεί σε κάποιον εξωτερικό πάροχο. Ωστόσο σε αντίθεση με το δημόσιο νέφος όπου όλοι οι πόροι είναι κοινόχρηστοι, στην τελευταία περίπτωση ο πάροχος δημιουργεί ένα αποκλειστικό περιβάλλον cloud και εγγυάται πλήρως για την ιδιωτικότητά του.

Υβριδικό νέφος (Hybrid cloud): Τα υβριδικά νέφη αποτελούν σύνθεση δύο ή περισσότερων cloud (ιδιωτικών, δημόσιων ή κοινότητας) τα οποία παραμένουν αυτόνομες οντότητες αλλά συνδέονται έτσι ώστε να προσφέρουν τα πλεονεκτήματα που έχει το κάθε είδος. Σε ένα υβριδικό νέφος μπορεί να γίνει αξιοποίηση των πλεονεκτημάτων των "τρίτων" παρόχων με πλήρη ή μερικό τρόπο, αυξάνοντας την ευελιξία των υπολογιστικών πόρων. Μία συνήθης τακτική από αρκετούς οργανισμούς είναι να συνδέουν το ιδιωτικό τους νέφος με ένα δημόσιο προκειμένου να ανταπεξέλθουν σε συνθήκες υψηλού φόρτου εργασίας που μπορεί να προκύψουν.

Νέφος κοινότητας (Community cloud): Το είδος αυτό του νέφους διαμοιράζεται μεταξύ διάφορων οργανισμών, στο οποίο ο έλεγχος, η διαχείριση και ασφάλεια γίνεται είτε από κοινού από όλους τους συμμετέχοντες, είτε από κάποιο τρίτο. Τα νέφη κοινότητας είναι μια υβριδική μορφή ιδιωτικών νεφών που δημιουργούνται και λειτουργούν συγκεκριμένα για μια ομάδα. Επιδίωξη των νεφών κοινότητας είναι να μπορέσουν οι συμμετέχοντες οργανισμοί να αξιοποιήσουν τα οφέλη ενός δημόσιου νέφους με το επιπρόσθετο επίπεδο ιδιωτικότητας και ασφάλειας που μόνο ένα ιδιωτικό νέφος προσφέρει.

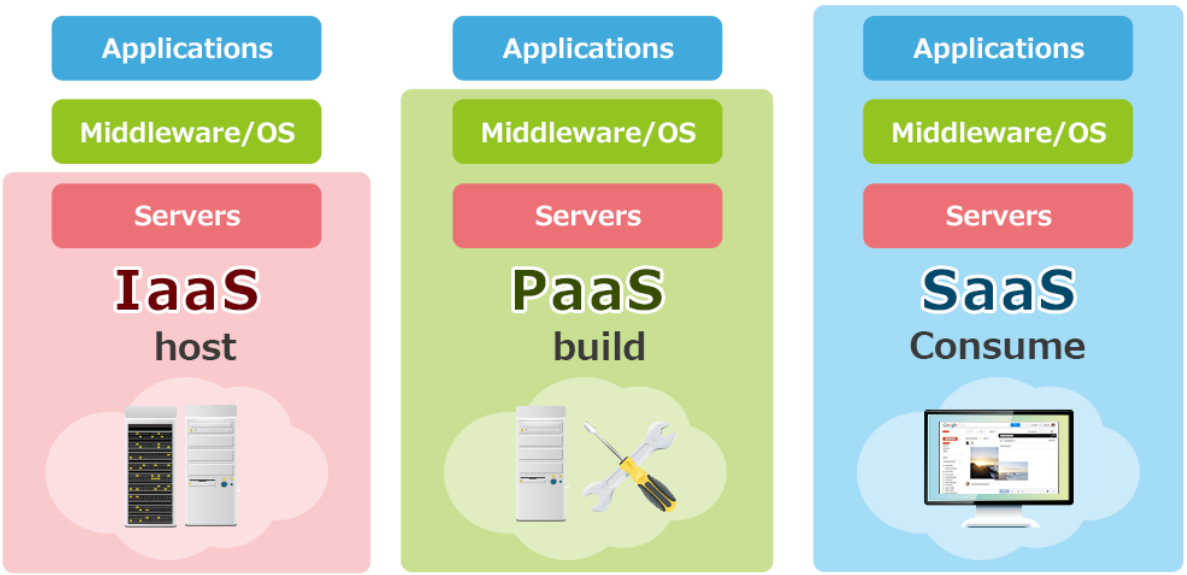
2.2.4 Μοντέλα Παροχής Υπηρεσιών

Υποδομή ως Υπηρεσία (IaaS): Πρόκειται για το πρώτο στρώμα του ΥΝ, μέσω του οποίου παρέχεται πρόσβαση σε ουσιώδεις υπολογιστικούς πόρους. Οι φυσικοί πόροι εικονοποιούνται, δηλαδή μπορούν να μοιραστούν από διαφορετικά λειτουργικά συστήματα και περιβάλλοντα χρηστών. Με βάση αυτό το μοντέλο, ο χρήστης μπορεί να διαχειριστεί τις εφαρμογές, τα δεδομένα, το λειτουργικό σύστημα, το middleware και τον χρόνο λειτουργίας. Ο πάροχος από την άλλη διαχειρίζεται την εικονοποίηση, τους εξυπηρετητές, τη δικτύωση και την αποθήκευση. (Εικόνα 6 IaaS - ροζ χρώμα) Αυτό επιτρέπει στον χρήστη να αποφύγει δαπάνες υλικού, ενώ η χρέωση αφορά μόνο τους πόρους που χρησιμοποιούνται.

Πλατφόρμα ως Υπηρεσία (PaaS): Πρόκειται για το δεύτερο στρώμα του ΥΝ και αποτελεί ουσιαστικά το περιβάλλον για ανάπτυξη cloud εφαρμογών. Ο χρήστης διαχειρίζεται τις εφαρμογές και τα δεδομένα ενώ ο πάροχος αναλαμβάνει και επιβλέπει όλα τα υπόλοιπα (διανομή της εφαρμογής στην υποκείμενη υποδομή, υποστήριξη του χρήστη με ένα σύνολο βασικών υπηρεσιών, εξασφάλιση της κλιμακωσιμότητας και ελαστικότητάς της, παρακολούθηση χρήσης των υπολογιστικών πόρων για επιβολή χρεώσεων κλπ.). (Εικόνα 6 PaaS - πράσινο χρώμα) Οι υπηρεσίες αυτού του μοντέλου προσφέρουν έναν συμβιβασμό μεταξύ πολυπλοκότητας και ευελιξίας, που επιτρέπει την γρήγορη υλοποίηση και διάθεση των εφαρμογών. Ωστόσο τα διαθέσιμα προγραμματιστικά εργαλεία και οι υπηρεσίες είναι πιθανόν

να διαφέρουν ανάμεσα σε διαφορετικές πλατφόρμες, γεγονός που χρειάζεται να λάβει σοβαρά υπόψιν του ο πελάτης πριν επιλέξει πάροχο για την ανάπτυξη της εφαρμογής του.

Λογισμικό ως Υπηρεσία (SaaS): Είναι το τρίτο και τελευταίο στρώμα και προσφέρει πλήρεις εφαρμογές στον τελικό χρήστη του ΥΝ. Σε αυτό το μοντέλο τα πάντα τα διαχειρίζεται ο πάροχος, ενώ ο χρήστης έχει απλά πρόσβαση σε έτοιμες διαδικτυακές εφαρμογές. (Εικόνα 6 SaaS – κυανό χρώμα) Η πρόσβαση σε αυτές γίνεται μέσω διαδικτύου και βασίζεται σε υπηριοσιοκεντρικές αρχιτεκτονικές. Οι χρήστες έχουν τη δυνατότητα να χρησιμοποιήσουν μια τέτοια υπηρεσία από μια πλειάδα συσκευών, ανεξαρτήτως λειτουργικού συστήματος, ενώ υποστηρίζονται και λειτουργίες cross-platforming, όπου πολλοί χρήστες μπορούν να κάνουν ταυτόχρονη και κοινή χρήση της ίδιας εφαρμογής από διαφορετικές συσκευές.



Εικόνα 6: Μοντέλα παροχής υπηρεσιών υπολογιστικού νέφους

2.3 Εικονοποίηση πόρων στο Υπολογιστικό Νέφος

Στην επιστήμη της πληροφορικής, η Εικονοποίηση (Virtualization) είναι ένας ευρύς όρος των υπολογιστικών συστημάτων που αναφέρεται σε έναν μηχανισμό αφαίρεσης, στοχευμένο στην απόκρυψη λεπτομερειών της υλοποίησης και της κατάστασης ορισμένων υπολογιστικών πόρων από πελάτες των πόρων αυτών (π.χ. εφαρμογές, άλλα συστήματα, χρήστες κλπ.). Η εν λόγω αφαίρεση μπορεί είτε να αναγκάζει έναν πόρο να συμπεριφέρεται ως πλειάδα πόρων (π.χ. μία συσκευή αποθήκευσης σε διακομιστή τοπικού δικτύου), είτε πολλαπλούς πόρους να συμπεριφέρονται ως ένας (π.χ. συσκευές αποθήκευσης σε κατανεμημένα συστήματα), δημιουργώντας κατ' αυτόν τον τρόπο μία εξωτερική διασύνδεση η οποία αποκρύπτει την υποκείμενη υλοποίηση.

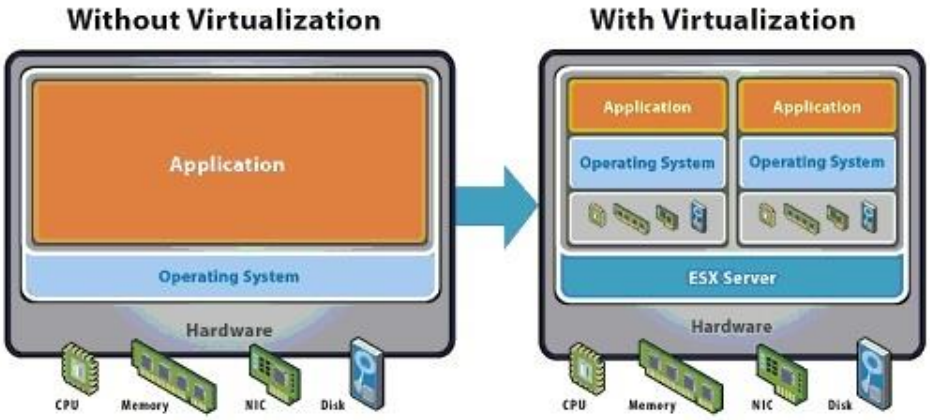
2.3.1 Εικονοποίηση Εξυπηρετητή

Η Εικονοποίηση Εξυπηρετητή (Server Virtualization) είναι μια τεχνική που επιτυγχάνει τον διαμερισμό των φυσικών πόρων ενός υπολογιστή σε πολλαπλά περιβάλλοντα εκτέλεσης, εφαρμόζοντας μία ή περισσότερες τεχνολογίες όπως διαμερισμό σε επίπεδο υλικού ή σε επίπεδο λογισμικού, διαμερισμό σε επίπεδο χρόνου, μερική ή ολική προσομοίωση μηχανής, εξομοίωση, ποιότητα υπηρεσιών κ.α. Είναι η μέθοδος εκτέλεσης πολλαπλών ανεξάρτητων ιδεατών λειτουργικών συστημάτων σε έναν φυσικό υπολογιστή. Είναι η απόκρυψη των φυσικών υπολογιστικών πόρων, συμπεριλαμβανομένων του αριθμού και της ταυτότητας των μεμονωμένων φυσικών εξυπηρετητών, επεξεργαστών και λειτουργικών συστημάτων από τους χρήστες του «ιδεατού» εξυπηρετητή. Η Εικονοποίηση, είναι ο διαμερισμός ενός φυσικού συστήματος σε πολλαπλά απομονωμένα μεταξύ τους εικονικά περιβάλλοντα. Τα εικονικά αυτά περιβάλλοντα συνήθως ονομάζονται εικονικοί ιδιωτικοί εξυπηρετητές (virtual private servers), αλλά μπορεί κανείς να τα συναντήσει και με το όνομα partitions, guests, instances, containers ή emulations ή virtual machines. Είναι ένα αφαιρετικό ενδιάμεσο στρώμα που επιτρέπει σε πολλαπλά ιδεατά μηχανήματα, με ετερογενή λειτουργικά συστήματα να λειτουργούν το καθένα ξεχωριστά μέσα σε ένα απομονωμένο περιβάλλον, το ένα δίπλα στο άλλο, πάνω στο ίδιο φυσικό μηχάνημα. Η Εικονοποίηση είναι μια δοκιμασμένη τεχνολογία λογισμικού που μετατρέπει ραγδαία το τοπίο στο ΥΝ και αλλάζει τον τρόπο με τον οποίο χρησιμοποιούμε τους υπολογιστές. Οι σημερινοί πολύ ισχυροί x86 υπολογιστές σχεδιάστηκαν αρχικά για να «τρέχουν» ένα μόνο λειτουργικό σύστημα και μία μόνο εφαρμογή. Η Εικονοποίηση καταργεί αυτή τη σύμβαση, κάνοντας εφικτό να εκτελούνται πολλαπλά λειτουργικά συστήματα και πολλαπλές εφαρμογές στον ίδιο υπολογιστή την ίδια χρονική στιγμή, αυξάνοντας έτσι την αξιοποίηση και την προσαρμοστικότητα των φυσικών πόρων. Η Εικονοποίηση είναι μια τεχνοτροπία από την οποία μπορούν να ωφεληθούν όλοι όσοι χρησιμοποιούν υπολογιστή, από τους επαγγελματίες της πληροφορικής και τους οπαδούς των Macintosh μέχρι τις εμπορικές επιχειρήσεις και τους κυβερνητικούς οργανισμούς. Χρησιμοποιώντας την Εικονοποίηση γίνεται εξοικονόμηση χρόνου, χρημάτων και ενέργειας ενώ παράλληλα δύναται να επιτευχθούν περισσότερα πράγματα με τους ήδη διαθέσιμους ηλεκτρονικούς υπολογιστές.

2.3.2 Πώς λειτουργεί η Εικονοποίηση Εξυπηρετητή;

Στην ουσία η Εικονοποίηση μας επιτρέπει να μετατρέψουμε το υλικό σε λογισμικό. Μπορούμε να χρησιμοποιήσουμε λογισμικό, όπως το VMware ESXi Server, για να μετατρέψουμε ή αλλιώς να κάνουμε «virtual» τους φυσικούς πόρους ενός υπολογιστή, συμπεριλαμβανομένων των επεξεργαστή (CPU), μνήμη (RAM), δίσκο και ελεγκτή δικτύου (network controller), προκειμένου να δημιουργήσουμε ένα πλήρως λειτουργικό ιδεατό μηχάνημα (virtual machine) που μπορεί να «τρέχει» το δικό του λειτουργικό σύστημα και τις δικές του εφαρμογές ακριβώς όπως ένας «πραγματικός» υπολογιστής. Πολλαπλά ιδεατά μηχάνημα μπορούν να μοιράζονται τους φυσικούς πόρους χωρίς να επηρεάζουν το ένα το

άλλο έτσι ώστε να μπορούμε με ασφάλεια να τρέξουμε πολλαπλά λειτουργικά συστήματα και εφαρμογές παράλληλα σε έναν υπολογιστή, μοιράζοντάς τον ουσιαστικά σε πολλούς ιδεατούς υπολογιστές (virtual machines), όπως φαίνεται και στην Εικόνα 7.



Εικόνα 7: Απεικόνιση λειτουργικού συστήματος σε φυσικό και σε ιδεατό εξυπηρετητή

2.3.3 Πλεονεκτήματα Εικονικών Μηχανών

Συμβατότητα: Ακριβώς όπως ένας φυσικός υπολογιστής, οι εικονικές μηχανές είναι απολύτως συμβατές με όλους τους τυποποιημένους x86 υπολογιστές, λειτουργικά συστήματα, εφαρμογές και οδηγούς συσκευών, έτσι μπορούμε να χρησιμοποιήσουμε μια εικονική μηχανή για να τρέξουμε όλο το ίδιο λογισμικό που εκτελούσαμε σε έναν φυσικό x86 υπολογιστή.

Απομόνωση: Ενώ οι εικονικές μηχανές μπορούν να μοιραστούν τους φυσικούς πόρους ενός ενιαίου υπολογιστή, παραμένουν εντελώς απομονωμένες η μια από την άλλη σαν να ήταν χωριστές φυσικές μηχανές. Εάν, για παράδειγμα, υπάρχουν τέσσερις εικονικές μηχανές σε έναν ενιαίο φυσικό κεντρικό υπολογιστή και μια από τις εικονικές μηχανές παρουσιάσει οποιοδήποτε πρόβλημα, οι άλλες τρεις εικονικές μηχανές παραμένουν διαθέσιμες. Η απομόνωση είναι ένας σημαντικός λόγος για τον οποίο η διαθεσιμότητα και η ασφάλεια των εφαρμογών που τρέχουν σε ένα εικονικό περιβάλλον είναι σαφώς ανώτερες από τις εφαρμογές που τρέχουν σε ένα παραδοσιακό σύστημα.

Φορητότητα: Οι εικονικές μηχανές είναι απίστευτα φορητές και εύκολες στη διαχείρισή τους. Μπορούμε να μετακινήσουμε και να αντιγράψουμε μια εικονική μηχανή ακριβώς όπως οποιαδήποτε άλλο αρχείο λογισμικού ή να σώσουμε μια εικονική μηχανή σε οποιοδήποτε τυποποιημένο μέσο αποθήκευσης, από μια κάρτα μνήμης USB μέχρι και σε κάποιο SAN (Storage Area Network).

Ανεξαρτησία Υλικού: Οι εικονικές μηχανές είναι απολύτως ανεξάρτητες από το φυσικό υλικό πάνω στο οποίο έχουν εγκατασταθεί. Για παράδειγμα, μπορούμε να διαμορφώσουμε μια εικονική μηχανή με τα εικονικά συστατικά (π.χ. επεξεργαστή, κάρτα δικτύων, ελεγκτή SCSI)

που να είναι απολύτως διαφορετικά από τα φυσικά συστατικά που είναι παρόντα στα εκάστοτε υλικά. Οι εικονικές μηχανές στον ίδιο φυσικό κεντρικό υπολογιστή μπορούν ακόμη και να τρέξουν διαφορετικά είδη λειτουργικών συστημάτων (Windows, Linux, κλπ.). Όταν δε αυτό συνδυάζεται με τις ιδιότητες της φορητότητας και της συμβατότητας, η ανεξαρτησία υλικού δίνει την δυνατότητα να μεταφέρουμε μια εικονική μηχανή από έναν τύπο x86 υπολογιστή προς άλλο χωρίς καμία απολύτως αλλαγή στους οδηγούς συσκευών, το λειτουργικό σύστημα, ή τις εφαρμογές. Η ανεξαρτησία υλικού επίσης σημαίνει ότι μπορούμε να τρέξουμε ένα ετερογενές μίγμα λειτουργικών συστημάτων και εφαρμογών σε έναν ενιαίο φυσικό υπολογιστή.

2.3.4 Υπερεπόπτης (Hypervisor - VMM)

Ο αρχικός διαχειριστής εικονικών μηχανών δημιουργήθηκε για να λύσει ένα συγκεκριμένο πρόβλημα, αλλά ο διαχειριστής εικονικών μηχανών έχει εξελιχθεί σε κάτι εντελώς διαφορετικό. Ο όρος διαχειριστής εικονικών μηχανών έχει πάψει να χρησιμοποιείται και έχει αντικατασταθεί από τον όρο Hypervisor. Οι Popek και Goldberg στο κείμενο τους "Formal Requirments for virtualizable third generation Architectures" το 1974, όρισαν τα χαρακτηριστικά που πρέπει να έχει ο Hypervisor :

• Ισοδυναμία (Equivalence): Ένα πρόγραμμα που τρέχει κάτω από τον επόπτη εικονικών μηχανών πρέπει να παρουσιάζει την προβλεπόμενη συμπεριφορά που είναι βασικά πανομοιότυπη με την παρουσιαζόμενη όταν αυτό τρέχει στο υποκείμενο υλικό απ' ευθείας. Αυτό αναφέρεται συχνά σαν πιστότητα (Fidelity).

• Έλεγχος Πόρων (Resource Control): Ο επόπτης εικονικών μηχανών πρέπει να έχει τον πλήρη έλεγχο των πόρων του πραγματικού υλικού που εικονικοποιούνται για τα φιλοξενούμενα (guest) λειτουργικά συστήματα σε οποιαδήποτε στιγμή. Συχνά αναφέρεται και ως ασφάλεια (Safety).

• Αποτελεσματικότητα (Efficiency/Performance): Ένας πολύ μεγάλος αριθμός από εντολές μηχανής πρέπει να εκτελεστούν χωρίς την μεσολάβηση του επόπτη εικονικών μηχανών, ή αλλιώς κατ' ευθείαν από το υλικό. Η απαίτηση απόδοσης στον ορισμό του Popek και Goldberg για ένα VMM αφορά μόνο την εκτέλεση μη προνομιούχων οδηγιών, οι οποίες πρέπει να εκτελούνται εγγενώς. Αυτό διαφοροποιεί ένα VMM από τη γενικότερη κατηγορία του λογισμικού εξομοίωσης υλικού (hardware emulation software). Μια μη προνομιούχος οδηγία (non-privileged instruction) είναι μια εντολή που μπορεί να εκτελέσει οποιαδήποτε εφαρμογή ή χρήστης. Μια προνομιούχος οδηγία (privileged instruction), από την άλλη πλευρά, είναι μια εντολή που μπορεί να εκτελεστεί μόνο σε λειτουργία πυρήνα (kernel mode). Οι οδηγίες διαιρούνται με αυτόν τον τρόπο επειδή οι προνομιούχες οδηγίες θα μπορούσαν να βλάψουν τον πυρήνα.

Οι σημερινοί Υπερεπόπτες μας επιτρέπουν να κάνουμε καλύτερη χρήση των όλο ένα και ταχύτερων επεξεργαστών που εμφανίζονται τακτικά στην αγορά και πιο αποτελεσματική

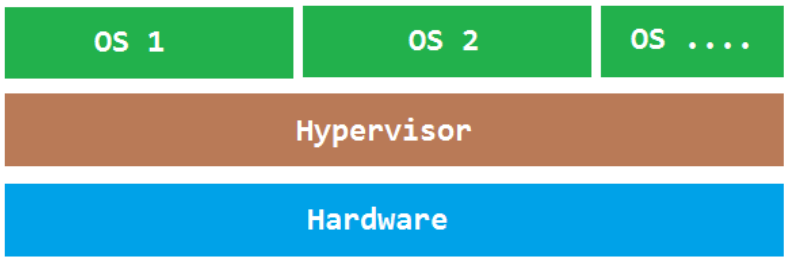
χρήση των μεγαλύτερων και πυκνότερων προσφορών μνήμης που εμφανίζονται μαζί με αυτούς. Η δομή ενός Υπερεπόπτη είναι αρκετά απλή: Αποτελείται από ένα στρώμα λογισμικού που ζει - υπάρχει ανάμεσα στο υλικό του οικοδεσπότη και τις εικονικές μηχανές που υποστηρίζει. Αυτές οι εικονικές μηχανές αποκαλούνται επίσης και φιλοξενούμενες. Ο σκοπός του Υπερεπόπτη είναι η διαχείριση και κατανομή των πόρων στις εικονικές μηχανές, και η εξασφάλιση ότι δεν θα διαταράξει η μία μηχανή την άλλη.

2.3.5 Τύποι Υπερεποπτών

Υπάρχουν δύο τύποι Υπερεποπτών: Ο Τύπος 1 (Type-1) και ο Τύπος 2 (Type-2). Ο πρώτος τύπος χρησιμοποιείται για εικονικοποίηση διακομιστών ενώ ο δεύτερος για εικονικοποίηση φορητών ή επιτραπέζιων υπολογιστών.

Υπερεπόπτης Τύπου 1

Οι Υπερεπόπτες Τύπου 1 είναι συστήματα λογισμικού που τρέχουν απευθείας στο υλικό του οικοδεσπότη χωρίς να υπάρχει λειτουργικό σύστημα από κάτω του. Επειδή δεν υπάρχει κάποιο στρώμα που να μεσολαβεί ανάμεσα στον Υπερεπόπτη και το φυσικό υλικό αυτό αναφέρεται και σαν υλοποίηση "γυμνού υλικού" (bare-metal). Χωρίς λοιπόν να υπάρχει μεσολαβητής ο Υπερεπόπτης Τύπου 1 μπορεί απευθείας να επικοινωνήσει με τους πόρους του υλικού στην κατώτερη στοίβα κάνοντας τον έτσι πιο αποτελεσματικό από τον Υπερεπόπτη Τύπου 2. Εκτός από ότι διαθέτει καλύτερα χαρακτηριστικά επιδόσεων, ο Υπερεπόπτης Τύπου 1, θεωρείται ότι διαθέτει μεγαλύτερη ασφάλεια από τον Υπερεπόπτη Τύπου 2.

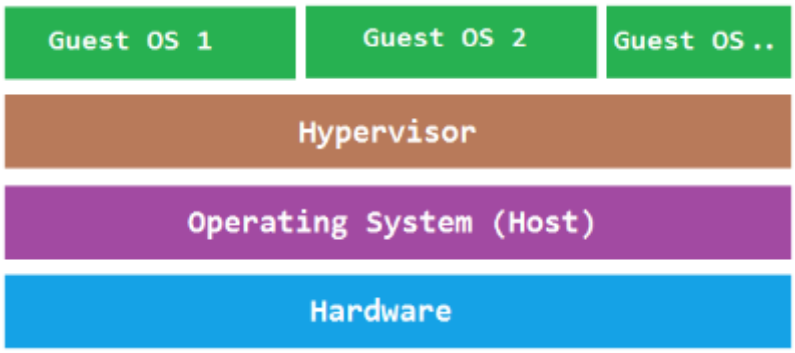


Εικόνα 8: Υπερεπόπτης Τύπου 1

Υπερεπόπτης Τύπου 2

Ο Υπερεπόπτης Τύπου 2 είναι από μόνος του μια εφαρμογή, η οποία τρέχει μέσα από ένα παραδοσιακό λειτουργικό σύστημα, το οποίο και διαχειρίζεται όλους τους πόρους του υλικού. Ένα πλεονέκτημα αυτού του μοντέλου είναι ότι μπορεί να υποστηρίξει ένα μεγάλο φάσμα υλικού διότι το λειτουργικό σύστημα είναι υπεύθυνο για την διαχείριση αυτού, χωρίς να επηρεάζεται ο Υπερεπόπτης. Πολλές φορές οι Τύπου 2 Υπερεπόπτες είναι εύκολο να εγκατασταθούν και να αναπτυχθούν, διότι μεγάλο μέρος της διαμόρφωσης του υλικού, όπως η δικτύωση και η αποθήκευση έχουν ήδη καλυφθεί από το λειτουργικό σύστημα. Κάθε φορά που μία εικονική μηχανή εκτελεί μια ανάγνωση δίσκου, μια λειτουργία δικτύου ή οποιαδήποτε

άλλη αλληλεπίδραση με το υλικό, περνά αυτό το αίτημα στον Υπερεπόπτη ακριβώς όπως στο περιβάλλον του Τύπου 1. Αντίθετα από αυτό το περιβάλλον, ο Υπερεπόπτης Τύπου 2 πρέπει να περάσει το αίτημα στο λειτουργικό σύστημα το οποίο διαχειρίζεται τα I/O αιτήματα. Το λειτουργικό σύστημα στέλνει την πληροφορία πίσω στον Υπερεπόπτη και μετά στο φιλοξενούμενο, προσθέτοντας έτσι δύο επιπρόσθετα βήματα, χρόνο και επεξεργασία σε κάθε διενέργεια.



Εικόνα 9: Υπερεπόπτης Τύπου 2

Οι Υπερεπόπτες Τύπου 2 είναι λιγότερο αξιόπιστοι, διότι υπάρχουν περισσότερα σημεία αποτυχίας, οτιδήποτε επηρεάζει την διαθεσιμότητα του υποκείμενου λειτουργικού μπορεί να έχει αντίκτυπο στον Υπερεπόπτη και στους φιλοξενούμενους που υποστηρίζει. Τα λογισμικά VMware Player, VMware Workstation και Microsoft Virtual Server αποτελούν παραδείγματα Υπερεπόπτη Τύπου 2.

2.4 Το λογισμικό Openstack

Εικόνα 10: Λογότυπο Openstack

Το Openstack[17] είναι ένα λογισμικό ανοιχτού κώδικα, το οποίο επιτρέπει την δημιουργία υποδομών Υπολογιστικού Νέφους. Δημιουργήθηκε από τη Rackspace Hosting και τη NASA, όταν το 2010 αποφάσισαν να δουλέψουν πάνω σε ένα κοινό έργο για την υλοποίηση μιας υποδομής Υπολογιστικού Νέφους. Σήμερα το Openstack είναι διαδεδομένο σε όλο τον κόσμο, ενώ περισσότερες από διακόσιες εταιρείες το χρησιμοποιούν (Yahoo , Intel , Oracle κ.α.). Τα συστήματα που είναι σχεδιασμένα στα πρότυπα του Openstack αποτελούνται από μια κεντρική αρχιτεκτονική, συνοδευόμενη από επιμέρους μικρότερα κομμάτια κώδικα που είναι υπεύθυνα για τον έλεγχο του μεγάλου όγκου υπολογιστικών πόρων που διαχειρίζονται.

[17] https://www.openstack.org/software/

Ο χρήστης μπορεί να ελέγξει τις λειτουργίες του Openstack, μέσω είτε γραμμής εντολών είτε της γραφικής διεπαφής που του παρέχεται, ωστόσο μπορεί να κάνει και χρήση APIs ώστε να έχει πρόσβαση στις low level λειτουργίες του συστήματος μέσα από τις ίδιες τις εφαρμογές.

2.4.1 Βασικές υπηρεσίες του OpenStack

Το OpenStack έχει μια αρθρωτή αρχιτεκτονική με διάφορα ονόματα κωδικών για τα συστατικά του:

Υπολογισμός (Nova): Το OpenStack Compute (Nova) αποτελεί το κύριο μέρος ενός συστήματος IaaS. Έχει σχεδιαστεί για να διαχειρίζεται και να αυτοματοποιεί τους πόρους του υπολογιστή, παρέχοντας μας έτσι μια πλατφόρμα στην οποία θα μπορούμε να τρέξουμε τις φιλοξενούμενες εικονικές μας μηχανές. Μπορεί να λειτουργήσει με ευρέως διαθέσιμες τεχνολογίες εικονικοποίησης, καθώς και με bare metal διαμορφώσεις υπολογιστών υψηλής απόδοσης (High Performance Computing - HPC). Τα KVM, VMware και Xen είναι διαθέσιμες επιλογές για τεχνολογία hypervisor, μαζί με τεχνολογία Hyper-V και Linux container, όπως LXC. Είναι γραμμένο σε Python και χρησιμοποιεί πολλές εξωτερικές βιβλιοθήκες όπως Eventlet (για παράλληλο προγραμματισμό), Kombu (για επικοινωνία AMQP) και SQLAlchemy (για πρόσβαση σε βάση δεδομένων).

Δικτύωση (Neutron): Το OpenStack Networking (Neutron) είναι ένα σύστημα διαχείρισης εικονικών δικτύων μέσω Software-Defined Networking[18] (SDN), παρέχοντας μας με αυτόν τον τρόπο το Δίκτυο ως Υπηρεσία (Networking as a Service). Δίνει στους χρήστες του ΥΝ ένα σύνθετο API για να δημιουργήσουν προηγμένες τοπικές δικτυακές τοπολογίες και να διαμορφώσουν τις παραμέτρους αυτών. Προσφέρει επίσης καινοτόμες επεκτάσεις (plugins) τόσο ανοικτού (open source) όσο και κλειστού (closed source) κώδικα ώστε να μπορούμε να κατασκευάζουμε και να διαχειριζόμαστε εικονικούς διακόπτες (switches) και δρομολογητές (routers).

Αποθήκευση Αντικειμένων (Swift): Το Swift προσφέρει λογισμικό αποθήκευσης στο ΥΝ (cloud storage software), η λειτουργία του οποίου μοιάζει με το Dropbox ή το Google Drive. Είναι κατασκευασμένο για μεγάλης κλίμακας δεδομένα με στόχο την ανθεκτικότητα, τη διαθεσιμότητα και την ταυτόχρονη πρόσβαση σε ολόκληρο το σύνολο δεδομένων αυτών. Το Swift είναι ιδανικό για την αποθήκευση μη δομημένων δεδομένων, ενώ παρέχει ένα άθροισμα ελέγχων ασφαλείας για τα αρχεία (checksum). Τα αρχεία αποθηκεύονται ως τμήματα και ένα αρχείο δήλωσης (manifest) είναι υπεύθυνο για να τα εντοπίζει.

Αποθήκευση Μπλοκ (Cinder): Το Cinder είναι επίσης ένα από τα δομικά στοιχεία αποθήκευσης του Openstack, η λειτουργία του οποίου μοιάζει με εξωτερικό σκληρό δίσκο ή συσκευή USB. Έχει τα χαρακτηριστικά απόδοσης ενός σκληρού δίσκου με χαμηλή

[18] https://www.opennetworking.org/sdn-definition/

καθυστέρηση (latency), ωστόσο είναι πολύ πιο αργό από το Swift. Η χρήση του ενδείκνυται σε σενάρια με αυστηρούς περιορισμούς επιδόσεων, όπως βάσεις δεδομένων και συστήματα αρχείων. Η πιο συνηθισμένη χρήση του Cinder είναι ο χώρος αποθήκευσης του Linux (Linux server storage), αλλά υπάρχουν και plug-ins για άλλες πλατφόρμες, όπως Ceph, NetApp, Nexenta και SolidFire. Οι χρήστες του ΥΝ μπορούν να διαχειριστούν τις μονάδες αποθήκευσης τους μέσω του πίνακα ελέγχου. Το σύστημα παρέχει διεπαφές για τη δημιουργία, την προσάρτηση και την αποσύνδεση συσκευών αποθήκευσης από / στους διακομιστές. Είναι επίσης δυνατό να δημιουργήσουμε αντίγραφα ασφαλείας για τους τόμους του Cinder χρησιμοποιώντας τη δυνατότητα στιγμιότυπων[19] (snapshots).

Ταυτότητα (Keystone): Το OpenStack Identity (Keystone) παρέχει έναν κεντρικό κατάλογο χρηστών με όλες υπηρεσίες OpenStack στις οποίες μπορούν να έχουν πρόσβαση. Λειτουργεί ως ένα κοινό σύστημα ελέγχου ταυτότητας σε όλο το λειτουργικό σύστημα νέφους και μπορεί να ενσωματωθεί με υπάρχουσες υπηρεσίες καταλόγου Back-End όπως το LDAP. Υποστηρίζει πολλαπλές μορφές επαλήθευσης ταυτότητας, συμπεριλαμβανομένων των τυπικών διαπιστευτηρίων ονόματος χρήστη και κωδικού πρόσβασης, συστημάτων που βασίζονται σε κουπόνι συνεδρίας (session token) ενώ είναι και συμβατό με τις μεθόδους επαλήθευσης της AWS (Amazon Web Services).

Εικόνα (Glance): Το OpenStack Image (Glance) παρέχει έναν κατάλογο για να βλέπουμε και να διαχειριζόμαστε αρχεία εικονικών μηχανών όπως guest (VM) images, εικονικούς δίκους (Disk Images) και στιγμιότυπα εκτέλεσης (snapshots). Ο κατάλογος του Glance προσφέρει επίσης και έτοιμες προ-ρυθμισμένες εικονικές μηχανές για να μπορούμε εύκολα να ξεκινήσουμε να τις χρησιμοποιούμε για την εργασία μας. Οι χρήστες έχουν τη δυνατότητα να ανεβάσουν και να αποθηκεύσουν στο Glance και τις δικές τους εικόνες (custom images), ενώ η εκκίνηση τους (boot) και η διαχείριση αυτών πραγματοποιείται εξολοκλήρου απομακρυσμένα, χωρίς να καταναλώνεται επιπλέον χώρος αποθήκευσης στον τοπικό δίσκο.

Πίνακας ελέγχου (Horizon): Το OpenStack Dashboard (Horizon) παρέχει στους διαχειριστές και τους χρήστες μια γραφική διεπαφή για την πρόσβαση, την παροχή και την αυτοματοποίηση της ανάπτυξης πόρων που βασίζονται στο ΥΝ. Ο πίνακας ελέγχου είναι ένας από τους πολλούς τρόπους με τους οποίους οι χρήστες μπορούν να αλληλεπιδρούν με τους πόρους του OpenStack. Οι προγραμματιστές μπορούν να αυτοματοποιήσουν την πρόσβαση ή να δημιουργήσουν εργαλεία για τη διαχείριση πόρων χρησιμοποιώντας το εγγενές API OpenStack ή το API συμβατότητας EC2.

Ενορχήστρωση (Heat): Η Heat είναι μια υπηρεσία για την ενορχήστρωση πολλαπλών σύνθετων εφαρμογών νέφους με χρήση προτύπων μέσω ενός REST API του OpenStack και ενός Query API συμβατό με το AWS CloudFormation[20]. Ένα πρότυπο (template) Heat

[19] https://stonefly.com/resources/what-is-storage-snapshot-technology
[20] http://docs.aws.amazon.com/AWSCloudFormation/latest/APIReference/Welcome.html?r=7078

περιγράφει την υποδομή για μια εφαρμογή νέφους σε ένα αρχείο κειμένου που είναι αναγνώσιμο και εγγράψιμο τόσο από τον άνθρωπο όσο και από τις μηχανές.

Τηλεμετρία (Ceilometer): Η Τηλεμετρία (Ceilometer) αποτελεί ένα ενιαίο σύστημα χρέωσης, παρέχοντας όλους τους μετρητές που χρειάζονται για να καθορίσουν την τιμολόγηση πελατών των επιμέρους υπηρεσιών του OpenStack. Το API της μπορεί να δημιουργεί χρεώσεις ανάλογα με τα στατιστικά χρήσης της κάθε υπηρεσίας, ενώ οι διαχειριστές μπορούν να θέσουν όρια μέγιστης χρήσης και συναγερμούς (alarms) σε περίπτωση υπέρβασης των ορίων αυτών.

2.5 Η Πλατφόρμα FIWARE για την Ανάπτυξη Εφαρμογών ΥΝ

Εικόνα 11: Λογότυπο Fiware

Το FIWARE[21] είναι μια πλατφόρμα middleware[22], βασιζόμενη στο λογισμικό Openstack για την ανάπτυξη και την παγκόσμια διανομή εφαρμογών στο ΥΝ. Η προδιαγραφή API του FIWARE είναι ανοικτή και απαλλαγμένη από δικαιώματα, όπου η συμμετοχή χρηστών και προγραμματιστών είναι κρίσιμη για να γίνει αυτή η πλατφόρμα μια τυποποιημένη και επαναχρησιμοποιήσιμη λύση. Ο στόχος του FIWARE είναι να διευκολύνει την οικονομικά αποδοτική δημιουργία και παροχή εφαρμογών και υπηρεσιών σε διάφορους τομείς, συμπεριλαμβανομένων των έξυπνων πόλεων, των βιώσιμων μεταφορών, της εφοδιαστικής, των ανανεώσιμων πηγών ενέργειας και της περιβαλλοντικής βιωσιμότητας.

2.5.1 Υπηρεσίες Γενικού και Ειδικού Σκοπού

Οι Υπηρεσίες Γενικού Σκοπού (Generic Enablers) διανέμονται μέσω διαδικτύου με τη μορφή SaaS από παρόχους Υπολογιστικού Νέφους για την δημιουργία εφαρμογών. Οι κανόνες για την δημιουργία τους υπόκεινται στην REST αρχιτεκτονική, ενώ πολλές φορές μέσω ενός κατάλληλου API αποκτάται πρόσβαση σε εξειδικευμένες λειτουργίες των υπηρεσιών. Τα βασικά χαρακτηριστικά μιας Υπηρεσίας Γενικού Σκοπού είναι η απλότητά στη χρήση της καθώς και η επαναχρησιμοποίησή της, με σκοπό την χρήση της ως συστατικό για την δημιουργία σύνθετων συστημάτων.

[21] https://www.fiware.org/
[22] https://en.wikipedia.org/wiki/Middleware

Η επικοινωνία και η σύνδεση παραπάνω της μιας Υπηρεσίας Γενικού Σκοπού δημιουργεί μια Υπηρεσία Ειδικού Σκοπού (Specific Enabler), που έχει ως στόχο την υλοποίηση μιας πιο σύνθετης λειτουργίας. Οι Υπηρεσίες Γενικού Σκοπού που συντελούν στην σύσταση μιας Υπηρεσίας Ειδικού Σκοπού είναι συνήθως αυτόνομες και ανεξάρτητες μεταξύ τους, γεγονός που καθιστά εύκολη την αντικατάστασή τους από άλλες υπηρεσίες αντίστοιχης λειτουργικότητας. Κάτι τέτοιο διευκολύνει αισθητά την αποσφαλμάτωση ενός συστήματος, καθώς αν εντοπιστεί πρόβλημα σε κάποια υπηρεσία, αυτή εύκολα μπορεί να αντικατασταθεί, χωρίς να χρειαστούν αλλαγές στα υπόλοιπα τμήματα της εφαρμογής.

2.5.2 Υπηρεσίες στο FIWARE

Οι πάροχοι Υπηρεσιών στο ΥΝ συνηθίζουν να διαθέτουν καταλόγους από υπηρεσίες που παρέχονται στους πελάτες, με σκοπό την διευκόλυνσή τους στην ανάπτυξη των δικτυακών τους εφαρμογών.

Ο κατάλογος του FIWARE[23] περιέχει μια πλούσια βιβλιοθήκη στοιχείων (Generic Enablers) με εφαρμογές αναφοράς που επιτρέπουν στους προγραμματιστές να υλοποιήσουν λειτουργίες όπως η σύνδεση στο Διαδίκτυο των Πραγμάτων (βλ. § 2.6) ή ανάλυση μεγάλων δεδομένων, διευκολύνοντας έτσι τον προγραμματισμό. Οι υπηρεσίες του FIWARE είναι όλες ανοικτού κώδικα και δωρεάν διαθέσιμες στο κοινό.

Για την καλύτερη ταξινόμηση και οργάνωση των υπηρεσιών, ο κατάλογος του FIWARE περιλαμβάνει τις επτά ακόλουθες κατηγορίες υπηρεσιών: Υπηρεσίες Διαχείρισης Συμβάντων και Δεδομένων, Υπηρεσίες Διαδικτύου των Πραγμάτων, Προηγμένες Διεπαφές Χρήστη Εφαρμογών Ιστού, Υπηρεσίες Ασφάλειας, Υπηρεσίες Δικτύων και Συσκευών, Εργαλεία Περιγραφής Υπηρεσιών για την Ανάπτυξη και Διανομή Εφαρμογών και Υπηρεσίες Cloud Hosting. Παρουσιάζονται ενδεικτικά στη συνέχεια κάποιες βασικές υπηρεσίες του FIWARE:

Publish/Subscribe Context Broker - Orion Context Broker: Το Orion Context Broker είναι μια εφαρμογή του NGSI9 και του NGIS10 (βλ. § 4.2.1.2) με μόνιμη αποθήκευση που βασίζεται στο MongoDB[24].

Ο Orion Context Broker είναι μια υλοποίηση Υπηρεσίας Διαχείρισης Συμβάντων και Συνδρομών, παρέχοντας τις διασυνδέσεις NGSI9 και NGSI10. Χρησιμοποιώντας αυτές τις διεπαφές, οι πελάτες μπορούν να κάνουν διάφορες λειτουργίες, όπως:

- Εγγραφή οντοτήτων για παρακολούθηση των συμβάντων αυτών, π.χ. έναν αισθητήρα θερμοκρασίας μέσα σε ένα δωμάτιο.
- Ενημέρωση πληροφοριών των καταχωρημένων οντοτήτων, π.χ. αποστολή ενημερώσεων θερμοκρασίας.

[23] https://catalogue.fiware.org/
[24] https://www.mongodb.com/

- Ενημέρωση όταν πραγματοποιούνται αλλαγές στις πληροφορίες περιβάλλοντος (π.χ. η θερμοκρασία έχει αλλάξει) ή με μια δεδομένη συχνότητα (π.χ. να πάρει τη θερμοκρασία κάθε λεπτό).
- Αποθήκευση των παραπάνω πληροφοριών σε καταλόγους αναζήτησης, ώστε τα ερωτήματα (queries) να επιλύονται με βάση αυτές τις πληροφορίες.

Complex Event Processing (CEP) - Proactive Technology Online: Το CEP GE αναλύει δεδομένα συμβάντων σε πραγματικό χρόνο, δημιουργεί άμεση γνώση και επιτρέπει την άμεση ανταπόκριση στις μεταβαλλόμενες συνθήκες. Η επεξεργασία συμβάντων είναι μια μέθοδος ανίχνευσης και ανάλυσης ροών δεδομένων σχετικά με τα γεγονότα που συμβαίνουν (events) και την εξαγωγή συμπερασμάτων από αυτά. Η σύνθετη επεξεργασία συμβάντων, ή CEP, είναι η επεξεργασία συμβάντων που συνδυάζει δεδομένα από πολλαπλές πηγές για να συμπεράνει γεγονότα ή μοτίβα που υποδεικνύουν πιο περίπλοκες περιστάσεις. Ο στόχος της πολύπλοκης επεξεργασίας συμβάντων είναι να εντοπιστούν σημαντικά γεγονότα και να απαντηθούν σε αυτά όσο το δυνατόν γρηγορότερα. Ένα συμβάν μπορεί επίσης να οριστεί ως "αλλαγή κατάστασης", όταν μια μέτρηση υπερβαίνει ένα προκαθορισμένο κατώτατο όριο (π.χ. χρόνου, θερμοκρασίας ή άλλης τιμής).

Identity Management (IdM) – KeyRock: Η διαχείριση ταυτότητας καλύπτει μια σειρά πτυχών που αφορούν την πρόσβαση των χρηστών σε δίκτυα, υπηρεσίες και εφαρμογές, συμπεριλαμβανομένης της ασφάλειας και της ιδιωτικής πιστοποίησης από χρήστες σε συσκευές, δίκτυα και υπηρεσίες, διαχείριση εξουσιοδότησης και εμπιστοσύνης, διαχείριση προφίλ χρηστών, διατήρηση προσωπικών δεδομένων, Single Sign-On[25] (SSO) προς εξυπηρετητές υπηρεσιών και παροχής αδειών προς εφαρμογές. Ο Διαχειριστής ταυτότητας είναι το κεντρικό στοιχείο που παρέχει μια γέφυρα μεταξύ συστημάτων IdM σε επίπεδο συνδεσιμότητας και επίπεδο εφαρμογής. Επιπλέον, η Διαχείριση Ταυτότητας χρησιμοποιείται για την εξουσιοδότηση ξένων υπηρεσιών για πρόσβαση σε προσωπικά δεδομένα που είναι αποθηκευμένα σε ασφαλές περιβάλλον. Με τον τρόπο αυτό συνήθως ο ιδιοκτήτης των δεδομένων πρέπει να δώσει τη συγκατάθεσή τους για την πρόσβαση στα δεδομένα.

2.6 Διαδίκτυο των Πραγμάτων

Με τον όρο Διαδίκτυο των πραγμάτων (Internet of Things - IoT) αναφερόμαστε στη διασύνδεση φυσικών συσκευών, οχημάτων (επίσης γνωστών ως "συνδεδεμένων συσκευών" και "έξυπνων συσκευών"), κτηρίων και άλλων αντικειμένων, που μέσω ενσωματωμένων ηλεκτρονικών μικροσυστημάτων, λογισμικού, αισθητήρων (ενεργητικών και παθητικών) παρέχουν δικτυακή συνδεσιμότητα με σκοπό την μεταξύ τους ανταλλαγή και την συλλογή δεδομένων.

[25] https://auth0.com/docs/sso/current

Το 2013 η Πρωτοβουλία για τα Παγκόσμια Πρότυπα για το Ίντερνετ των Πραγμάτων (Global Standards Initiative on Internet of Things (IoT-GSI)) καθόρισε το IoT ως "μια παγκόσμια υποδομή για την κοινωνία της πληροφορίας, επιτρέποντας προηγμένες υπηρεσίες με διασύνδεση (φυσικών και εικονικών) πραγμάτων βασισμένα στις υπάρχουσες και εξελισσόμενες διαλειτουργικές τεχνολογίες πληροφοριών και επικοινωνιών. Για το σκοπό αυτό ως "πράγμα" νοείται "ένα αντικείμενο του φυσικού κόσμου (φυσικό πράγμα) ή του κόσμου της πληροφορίας (εικονικό πράγμα), το οποίο είναι ικανό να ταυτοποιηθεί και να ενσωματωθεί στα δίκτυα επικοινωνίας".

Το IoT επιτρέπει στα αντικείμενα να ανιχνεύονται ή να ελέγχονται εξ' αποστάσεως σε όλη την υπάρχουσα υποδομή δικτύου, δημιουργώντας με αυτόν τον τρόπο ευκαιρίες για πιο άμεση ενσωμάτωση του φυσικού κόσμου σε συστήματα βασισμένα σε υπολογιστές και οδηγώντας σε βελτιωμένη αποτελεσματικότητα, ακρίβεια και οικονομικό όφελος, με μειωμένη την ανθρώπινη παρέμβαση.

Όταν το IoT συμπληρώνεται με ενεργητικούς αισθητήρες, η τεχνολογία αποτελεί μέρος μιας γενικότερης κλάσης κυβερνο-φυσικών συστημάτων, τα οποία περιλαμβάνουν επίσης τεχνολογίες όπως τα έξυπνα δίκτυα, οι εικονικές μονάδες παραγωγής ενέργειας, τα έξυπνα σπίτια, οι έξυπνες μεταφορές και οι έξυπνες πόλεις. Οι ειδικοί εκτιμούν ότι το IoT θα αποτελείται από περίπου 30 δισεκατομμύρια αντικείμενα έως το 2020.

2.7 Non-IP Πρωτόκολλα Τοπικής Δικτύωσης

Όπως αναφέραμε το Διαδίκτυο των Πραγμάτων αποτελείται από "συνδεδεμένες συσκευές" που επικοινωνούν διαρκώς μεταξύ τους ανταλλάσσοντας πληροφορία. Παρακάτω θα δούμε συνοπτικά κάποια από τα πιο γνωστά Non-IP πρωτόκολλα που επιτρέπουν την ασύρματη μεταφορά δεδομένων σε δίκτυα IoT συσκευών:

RFID (Radio-Frequency IDentification)

Η αναγνώριση ραδιοσυχνοτήτων (RFID) χρησιμοποιεί ηλεκτρομαγνητικά πεδία για την αυτόματη αναγνώριση και παρακολούθηση ετικετών που συνδέονται με αντικείμενα. Οι ετικέτες περιέχουν ηλεκτρονικά αποθηκευμένες πληροφορίες. Οι παθητικές ετικέτες συλλέγουν ενέργεια από τα ασύρματα ραδιοκύματα κάποιου κοντινού αναγνώστη RFID, συνεπώς δεν διαβάζονται από μεγάλη απόσταση ούτε υποστηρίζουν αμφίδρομη επικοινωνία. Οι ενεργές ετικέτες έχουν μια τοπική πηγή ενέργειας (όπως μια μπαταρία) και μπορεί να λειτουργούν εκατοντάδες μέτρα από τον αναγνώστη RFID. Σε αντίθεση με έναν γραμμωτό κώδικα (barcode), η ετικέτα δεν χρειάζεται να βρίσκεται εντός της οπτικής επαφής του αναγνώστη, επομένως μπορεί να ενσωματωθεί στο αντικείμενο που παρακολουθείται.

NFC (Near Field Communication)

Ασύρματο πρωτόκολλο μικρής εμβέλειας που επιτρέπει σε δύο ηλεκτρονικές συσκευές να επικοινωνούν όταν πλησιάζουν πολύ κοντά μεταξύ τους. Υπάρχουν τρεις λειτουργίες στις οποίες μια NFC συσκευή μπορεί να λειτουργήσει:

1. **Προσομοίωση φυσικών καρτών:** Σε αυτή τη λειτουργία οι NFC συσκευές συμπεριφέρονται ως φυσικές κάρτες (π.χ. χρήση του τηλεφώνου για την πραγματοποίηση συναλλαγών αντί πιστωτικής κάρτας, πρόσβαση σε χώρους αντί για κάρτα – κλειδί κλπ.)
2. **Αναγνώστης ετικετών:** Η συσκευή NFC μπορεί να διαβάζει πληροφορία από συμβατές ετικέτες πάνω σε αντικείμενα.
3. **NFC peer-to-peer:** Όπου δύο συσκευές μπορούν να επικοινωνήσουν μεταξύ τους αν έρθουν κοντά.

ZigBee

Το Zigbee είναι μια προδιαγραφή που βασίζεται στο IEEE 802.15.4 για μια σειρά πρωτοκόλλων επικοινωνίας υψηλού επιπέδου που χρησιμοποιούνται για τη δημιουργία δικτύων ιδιωτικών περιοχών με μικρά, χαμηλής ισχύος ψηφιακά ραδιόφωνα, όπως για οικιακό αυτοματισμό, συλλογή δεδομένων ιατρικών συσκευών και άλλες χαμηλής ισχύος ανάγκες χαμηλού εύρους ζώνης, σχεδιασμένα για έργα μικρής κλίμακας που χρειάζονται ασύρματη σύνδεση. Ως εκ τούτου, το Zigbee είναι ασύρματο δίκτυο ad hoc (δηλ. δεν χρησιμοποιεί κάποια εξωτερική συσκευή δρομολόγησης ή σταθμό βάσης) χαμηλής κατανάλωσης, χαμηλής ταχύτητας μεταφοράς δεδομένων και κοντινής εμβέλειας.

Η τεχνολογία που καθορίζεται από την προδιαγραφή Zigbee προορίζεται να είναι απλούστερη και λιγότερο δαπανηρή από άλλα ασύρματα δίκτυα, όπως το Bluetooth ή το Wi-Fi. Οι εφαρμογές του περιλαμβάνουν ασύρματους διακόπτες φωτισμού, οθόνες οικιακής ενέργειας, συστήματα διαχείρισης κυκλοφορίας και άλλον καταναλωτικό και βιομηχανικό εξοπλισμό που απαιτεί ασύρματη μεταφορά δεδομένων μικρής εμβέλειας.

Η χαμηλή κατανάλωση ενέργειας περιορίζει τις αποστάσεις μετάδοσης σε οπτική επαφή 10-100 μέτρων, ανάλογα με την ισχύ εξόδου και τα περιβαλλοντικά χαρακτηριστικά. Οι συσκευές Zigbee ωστόσο μπορούν να μεταδίδουν δεδομένα και σε μεγάλες αποστάσεις μεταφέροντας τα μέσω ενός δικτύου ενδιάμεσων συσκευών για να καλυφτεί η απόσταση. Το Zigbee χρησιμοποιείται συνήθως σε εφαρμογές χαμηλού ρυθμού δεδομένων που απαιτούν μεγάλη διάρκεια ζωής της μπαταρίας και ασφαλή δικτύωση (τα δίκτυα Zigbee είναι ασφαλισμένα με συμμετρικά κλειδιά κρυπτογράφησης 128 bit). Το Zigbee έχει καθορισμένο ρυθμό 250 kbit / s, κατάλληλο για διακοπτόμενες μεταδόσεις δεδομένων από αισθητήρα ή συσκευή εισόδου.

DASH7

Το πρωτόκολλο κάνει χρήση ραδιοσυχνοτήτων, ωστόσο κύριο χαρακτηριστικό που το διαφοροποιεί από αντίστοιχες τεχνολογίες, είναι η εξαιρετικά χαμηλή κατανάλωση ενέργειας

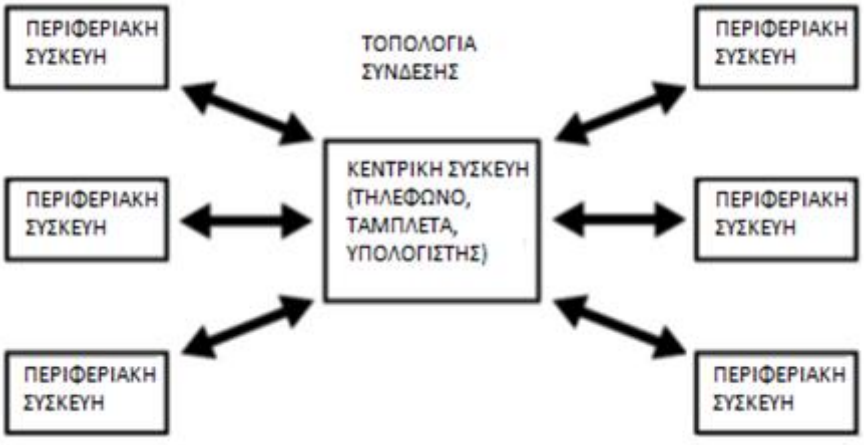
σε συνδυασμό με πολύ μεγάλη εμβέλεια επικοινωνίας, της τάξης του 1km. Η χαμηλή κατανάλωση ενέργειας επιτυγχάνεται μέσω της αδρανοποίησης του πομποδέκτη για όσο χρονικό διάστημα αυτός δεν επικοινωνεί, ενώ η αφύπνιση του πραγματοποιείται με την λήψη κατάλληλου ερεθίσματος (wake up signal).

Wi-Fi

Επιτρέπει στις συσκευές να επικοινωνούν μεταξύ τους μέσω ενός ασύρματου τοπικού δικτύου (WLAN[26]) κάνοντας χρήση των μπαντών συχνοτήτων 2.4GHz UHF και 5GHz SHF. Κατόπιν μέσω ενός σημείου πρόσβασης (wireless access point) οι συσκευές ενός WLAN μπορούν να συνδεθούν με το διαδίκτυο. Χαρακτηρίζεται από υψηλό ρυθμό μεταφοράς δεδομένων (300MBps) αλλά και από υψηλή κατανάλωση ενέργειας, καθώς απαιτεί μόνιμη σύνδεση σε πηγή τροφοδοσίας, γεγονός που το καθιστά μη πρακτικό για χρήση σε απομακρυσμένους εξωτερικούς χώρους.

Bluetooth

Το πρωτόκολλο Bluetooth μεταδίδει UHF ραδιοκύματα στο εύρος ζώνης 2.4 – 2.485GHz και χρησιμοποιείται για την ασύρματη μεταφορά δεδομένων σε σχετικά κοντινές αποστάσεις. Η εμβέλεια του πρωτοκόλλου, στις πιο σύγχρονες εκδοχές του, προσεγγίζει σε ιδανικές συνθήκες τα 100 μέτρα σε ανοικτό χώρο. Ο ρυθμός μετάδοσής του ανέρχεται στα 2.1Mbps, με την επικοινωνία μεταξύ συσκευών να είναι από σημείο σε σημείο[27] (point to point). Η συγκεκριμένη τοπολογία επιτρέπει την ανταλλαγή πληροφοριών ανάμεσα σε δύο μόνο συσκευές Bluetooth τη φορά, με την πρώτη να έχει το ρόλο του εξυπηρετητή (server), ενώ η δεύτερη του (πελάτη) client. Ωστόσο είναι εφικτό να δημιουργηθούν και δίκτυα αποτελούμενα από πολλές Bluetooth συσκευές, με μία όμως από αυτές να αποτελεί την κεντρική συσκευή (coordinator device), ενώ όλες οι υπόλοιπες επικοινωνούν με αυτήν, η καθεμιά ξεχωριστά με τη σειρά της (Εικόνα 12).



Εικόνα 12: Τοπολογία Bluetooth συνδεσιμότητας με την χρήση κεντρικής συσκευής

[26] https://en.wikipedia.org/wiki/Wireless_LAN
[27] http://wiki.p2pfoundation.net/Network_Topology

2.7.1 Bluetooth χαμηλής κατανάλωσης[28] (Bluetooth v4.0 Low Energy - BLE)

Ως εξέλιξη του κλασικού Bluetooth πρωτοκόλλου, η συγκεκριμένη έκδοση σχεδιάστηκε για χρήση από συσκευές στο Διαδίκτυο των Πραγμάτων, με κύριο χαρακτηριστικό της την πολύ χαμηλή κατανάλωση ενέργειας. Ένας BLE αισθητήρας έχει πολύ μικρό κόστος αγοράς, ενώ η μπαταρία του, συνήθως σε μέγεθος κέρματος, έχει αυτονομία έως και για δύο χρόνια. Το BLE πρωτόκολλο έχει υιοθετηθεί από το σύνολο σχεδόν των κατασκευαστών IoT μικροσυσκευών – και όχι άδικα – αφού προσφέρει μεγάλη οικονομία, αλλά και αυξημένη ασφάλεια μέσω κρυπτογράφησης 128bit AES.

Το Attribute Protocol (ATT) και το Generic Attribute Profile (GATT) είναι τα σημαντικότερα τμήματα της στοίβας πρωτοκόλλων του Bluetooth Low Energy, σε ό,τι αφορά τον προγραμματισμό BLE εφαρμογών. Πάνω στα τμήματα αυτά είναι χτισμένα τα APIs στις περισσότερες πλατφόρμες που υποστηρίζουν το BLE και με τα οποία έρχονται σε επαφή κατά κύριο λόγο οι προγραμματιστές. Τα ATT και GATT δηλαδή, αποτελούν το μοντέλο δεδομένων του BLE, πάνω στο οποίο μπορούν να δημιουργηθούν οι διάφορες εφαρμογές και περιπτώσεις χρήσης.

Είναι σημαντικό να επισημάνουμε την διαφορά ανάμεσα στα GAP και GATT. Το GAP ορίζει τη γενική τοπολογία της στοίβας ενός BLE δικτύου. Το GATT περιγράφει λεπτομερώς πως τα χαρακτηριστικά (attributes) των δεδομένων μεταφέρονται αφού επιτευχθεί η σύνδεση μεταξύ των συσκευών. Το GATT επικεντρώνεται ειδικά στον τρόπο μορφοποίησης, συσκευασίας και αποστολής δεδομένων σύμφωνα με τους κανόνες που περιγράφονται. Στη στοίβα δικτύου BLE, το Πρωτόκολλο Χαρακτηριστικών (ATT) είναι στενά αντιστοιχισμένο με το GATT, όπου η GATT βρίσκεται ακριβώς πάνω από το ATT. Το GATT χρησιμοποιεί στην πραγματικότητα ATT για να περιγράψει τον τρόπο ανταλλαγής δεδομένων από δύο συνδεδεμένες συσκευές.

2.7.1.1 Γενικό Προφίλ Πρόσβασης - Generic Access Profile (GAP)

Υπάρχουν δύο μηχανισμοί που η συσκευή BLE μπορεί να χρησιμοποιήσει για να επικοινωνήσει με τον έξω κόσμο: εκπομπή (broadcasting) ή σύνδεση. Αυτοί οι μηχανισμοί υποβάλλονται στις οδηγίες γενικού προφίλ πρόσβασης (GAP). Το GAP καθορίζει τον τρόπο με τον οποίο οι συσκευές με δυνατότητα BLE μπορούν να γίνουν διαθέσιμες για επίτευξη σύνδεσης και τον τρόπο με τον οποίο δύο συσκευές μπορούν να επικοινωνούν απευθείας μεταξύ τους. Μια συσκευή μπορεί να ενταχθεί σε ένα δίκτυο BLE υιοθετώντας τους ρόλους που καθορίζονται στο GAP.

Έτσι κατά το στάδιο της εκπομπής (broadcasting) πριν την επίτευξη σύνδεσης μια συσκευή μπορεί να έχει το ρόλο του εκπομπού (broadcaster), όπου διαφημίζει τον εαυτό της

[28] https://www.bluetooth.com/what-is-bluetooth-technology/how-it-works/le-p2p

μεταδίδοντας περιοδικά πακέτα διαφήμισης, είτε του παρατηρητή (observer), όπου ακούει τα πακέτα διαφήμισης που εκπέμπονται στο περιβάλλον της. Αμέσως μετά την επίτευξη της σύνδεσης, προκειμένου να αρχίσει η μεταφορά των δεδομένων, μια συσκευή θα πρέπει να πάρει το ρόλο είτε της κεντρικής συσκευής είτε της περιφερειακής συσκευής.

- **Περιφερειακή**: Μια περιφερειακή συσκευή διαφημίζει την παρουσία της σε κεντρικές συσκευές μέχρι να δημιουργήσει μια σύνδεση. Μετά τη σύνδεση, τα περιφερειακά δεν μεταδίδουν πλέον δεδομένα σε άλλες κεντρικές συσκευές και παραμένουν συνδεδεμένα στη συσκευή που δέχτηκε αίτηση σύνδεσης. Τα περιφερειακά είναι χαμηλής ισχύος επειδή πρέπει να στέλνουν πακέτα περιοδικά. Οι κεντρικές συσκευές είναι υπεύθυνες για την έναρξη της επικοινωνίας με περιφερειακά. Το beacon είναι ένα παράδειγμα περιφερειακού BLE.
- **Κεντρική**: Μια συσκευή που ξεκινά μια σύνδεση με μια περιφερειακή συσκευή, αρχικά ακούγοντας τα διαφημιστικά πακέτα. Μια κεντρική συσκευή μπορεί να συνδεθεί με πολλές άλλες περιφερειακές συσκευές. Όταν η κεντρική συσκευή θέλει να συνδεθεί, στέλνει ένα πακέτο δεδομένων σύνδεσης αίτησης στην περιφερειακή συσκευή. Εάν η περιφερειακή συσκευή δέχεται το αίτημα από την κεντρική συσκευή, δημιουργείται μια σύνδεση. Ο υπολογιστής μας είναι ένα παράδειγμα κεντρικής συσκευής BLE όταν συνδέεται με το beacon.

Οι κεντρικές συσκευές μπορούν να ενημερώσουν τις παραμέτρους σύνδεσης: Αφού επιτευχθεί η σύνδεση, η κεντρική συσκευή τυπικά καθορίζει τις παραμέτρους σύνδεσης μεταξύ της περιφερειακής συσκευής και της ίδιας. Ωστόσο, η περιφερειακή συσκευή μπορεί να ζητήσει από την κεντρική συσκευή να αλλάξει τις παραμέτρους σύνδεσης.

Οι περιφερειακές ή κεντρικές συσκευές μπορούν να τερματίσουν τις συνδέσεις: Οι συνδέσεις ενδέχεται να τερματίζονται για διάφορους λόγους: η μπαταρία μιας συσκευής μπορεί να τελειώσει ή οι παρεμβολές δικτύου ενδέχεται να προκαλέσουν βλάβη στη σύνδεση. Οι συσκευές μπορούν επίσης να αποσυνδεθούν σκόπιμα από τις άλλες.

2.7.1.2 Γενικό Προφίλ Χαρακτηριστικών - Generic Attribute Profile (GATT)

Παρόμοια με το GAP, υπάρχουν ορισμένοι ρόλοι που μπορούν να υιοθετήσουν οι συσκευές:

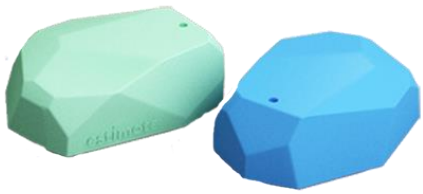
- **Πελάτης (Client)**: Συνήθως στέλνει αίτημα στον εξυπηρετητή GATT. Ο πελάτης μπορεί να διαβάσει και / ή να γράψει χαρακτηριστικά που βρίσκονται στο διακομιστή.
- **Εξυπηρετητής (Server)**: Ένας από τους κύριους ρόλους του διακομιστή είναι η αποθήκευση των χαρακτηριστικών. Μόλις ο πελάτης υποβάλει ένα αίτημα, ο διακομιστής πρέπει να κάνει τα χαρακτηριστικά διαθέσιμα.

Σχέσεις πελάτη-διακομιστή: Ένα παράδειγμα σχέσης πελάτη-διακομιστή έχει ως εξής: Έχουμε ένα beacon και θέλουμε ο υπολογιστής να διαβάσει αυτές τις πληροφορίες που αυτό αποστέλλει. Το beacon λειτουργεί ως διακομιστής και παρέχει πληροφορίες. Ο υπολογιστής λειτουργεί ως πελάτης, διαβάζοντας αυτές τις πληροφορίες.

Οι ρόλοι GAP και GATT είναι ουσιαστικά ανεξάρτητοι μεταξύ τους. Οι περιφερειακές ή κεντρικές συσκευές μπορούν αμφότερες να ενεργούν ως διακομιστές ή πελάτες, ανάλογα με τον τρόπο ροής των δεδομένων. Σε αντίθεση με το παραπάνω παράδειγμα, εάν θέλαμε να στείλουμε μια ενημέρωση από τον υπολογιστή προς το beacon, τότε ο υπολογιστής λειτουργεί ως διακομιστής και το beacon ενεργεί ως πελάτης.

2.7.2 Αισθητήρες Bluetooth της Estimote

Πρόκειται για BLE αισθητήρες μικρών διαστάσεων (stickers και beacons) με ενσωματωμένη μπαταρία μεγάλης αυτονομίας. Η εταιρία διαθέτει μοντέλα με διαφορές στο είδος των μετρήσεων που μπορούν να πραγματοποιήσουν (π.χ. απόσταση, θερμοκρασία, επιτάχυνση, φωτισμό) αλλά και στα λοιπά τεχνικά τους χαρακτηριστικά (διάρκεια μπαταρίας, ρυθμός αποστολής, ταχύτητα επεξεργασίας, κρυπτογράφηση δεδομένων). Επιπλέον μέσω του ειδικού API που συνοδεύει τους αισθητήρες, η όλη συνδεσιμότητα, η λήψη, αλλά και η επεξεργασία των μετρήσεων τους από εξωτερικές εφαρμογές απλουστεύεται αρκετά. Το API περιέχει μεθόδους για την αυτόματη επίτευξη σύνδεσης της κινητής συσκευής με τους αισθητήρες, αλλά την ανάγνωση όλων των μετρήσεων (π.χ. θερμοκρασία, απόσταση) και των χαρακτηριστικών (π.χ. διεύθυνση MAC, επίπεδο μπαταρίας) αυτών. Υπάρχουν διαθέσιμα Estimote APIs για iOS, Android αλλά και γενικής χρήσης σε JavaScript, του οποίου έγινε και χρήση για την ανάπτυξη της εφαρμογής μέσω του περιβάλλοντος Cordova. Για τους σκοπούς της εργασίας επιλέχτηκαν beacons ανίχνευσης προσέγγισης, με την μέτρηση που μας ενδιαφέρει να είναι η απόσταση των αισθητήρων από τη συσκευή που τρέχει η εφαρμογή πλοήγησης.



Εικόνα 13: Αισθητήρες τύπου beacon Bluetooth Low Energy της Estimote

2.8 Βασική Αρχιτεκτονική IoT Υπηρεσιών

Βασιζόμενοι στην Υπηρεσιοκεντρική Αρχιτεκτονική, οι πάροχοι υπηρεσιών Νέφους αναπτύσσουν εφαρμογές και υπηρεσίες, ο καθένας ακολουθώντας τη δικιά του υλοποίηση και τα δικά του μέσα (λειτουργικό σύστημα, γλώσσα προγραμματισμού, φυσικούς πόρους κλπ.). Οι υπηρεσίες αυτές προσφέρονται μέσω διαδικτύου με σκοπό την χρησιμοποίησή τους για την ανάπτυξη σύνθετων υπηρεσιών.

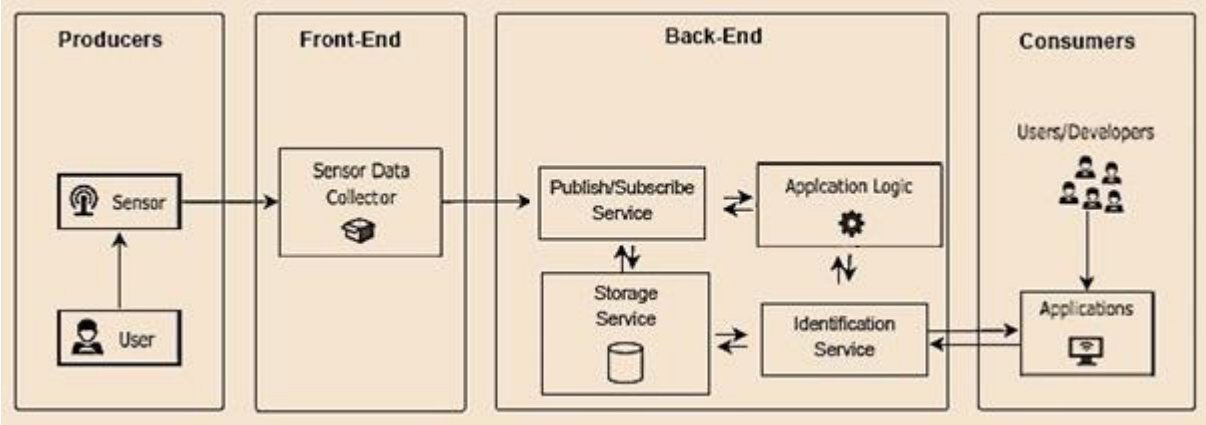
Σύμφωνα με το κλασικό πρότυπο ανάπτυξης IoT εφαρμογών στο ΥΝ, οι παρεχόμενες υπηρεσίες, αλλά και οι χρήστες ενός IoT συστήματος διακρίνονται σε 4 κατηγορίες με βάση το είδος του χρήστη, τις συσκευές ή την χρήση που εξυπηρετούν. Ειδικότερα, διακρίνονται σε υπηρεσίες για (βλ. και Εικόνα 14) Παραγωγούς, (Producers), Υπηρεσίες Διεπαφής Χρήστη (Front End), Υπηρεσίες Διεπαφής Συστήματος (Back End) και Υπηρεσίες για Καταναλωτές (Consumers).

Παραγωγοί (Producers): Είναι οι αισθητήρες (Εικόνα 14 – Sensors) που παράγουν την πληροφορία. Η πληροφορία συνήθως είναι μια μέτρηση που πραγματοποιούν οι αισθητήρες και έχει ενδιαφέρον για εμάς. Τέτοιες μετρήσεις για παράδειγμα μπορεί να είναι η θερμοκρασία και η υγρασία σε ένα χώρο ή αρτηριακή πίεση και οι καρδιακοί παλμοί ενός ασθενή. Στην εφαρμογή μας η μέτρηση που μας ενδιαφέρει είναι η απόσταση των αισθητήρων από μία άλλη συσκευή με την οποία συνδέονται.

Διεπαφή Χρήστη (Front-End): Περιλαμβάνεται η υπηρεσία που παίζει το ρόλο μιας πύλης δικτύου ανάμεσα στα δεδομένα που στέλνει ο αισθητήρας και τα δεδομένα που διαχειρίζεται η εφαρμογή μέσω της «Διεπαφής Διαχείρισης Συστήματος» (Back-End). Στο Front-End περιλαμβάνεται επίσης η γραφική διεπαφή χρήστη της κινητής συσκευής, καθώς και το API (βλ. § 2.7.2) που είναι υπεύθυνο για τη συλλογή μετρήσεων από τους αισθητήρες (Εικόνα 14 – Sensor Data Collector)

Διεπαφή Διαχείρισης Συστήματος (Back End): Περιλαμβάνονται όλες οι υπηρεσίες γενικού σκοπού (για αναλυτική περιγραφή των υπηρεσιών back-end βλ. § 4.2.1) για την ταυτοποίηση του χρήστη (Εικόνα 14 – Identification Service), τη δημιουργία συνδρομών (Εικόνα 14 – Publish/Subscribe Service) και την αποθήκευση των δεδομένων (Εικόνα 14 – Storage Service). Οι υπηρεσίες αυτές επικοινωνούν με την Λογική Συστήματος (Εικόνα 14 – Application Logic), μέσω της οποίας τα δεδομένα διανέμονται στα επί μέρους τμήματα της υπολογιστικής υποδομής για την περεταίρω επεξεργασία τους. Η Λογική Συστήματος αποτελεί δηλαδή τον ενορχηστρωτή του συστήματός στο νέφος, αφού μέσω των οδηγιών αυτής καθορίζονται οι ενέργειες που επιτελούν όλες οι υπόλοιπες υπηρεσίες.

Καταναλωτές (Consumers): Αποτελούν τους τελικούς χρήστες (Εικόνα 14 – Users/Developers), είτε άλλες εφαρμογές (Εικόνα 14 – Applications) που έχουν την δυνατότητα να αλληλοεπιδρούν με κάθε ενότητα του συστήματος.



Εικόνα 14: Πρότυπη απεικόνιση της IoT Αρχιτεκτονικής για την ανάπτυξη εφαρμογών στο ΥΝ

2.9 Ανάπτυξη IoT Εφαρμογών με τη βοήθεια του Apache Cordova

Εικόνα 15: Λογότυπο Apache Cordova

Το Apache Cordova[29] (πρώην PhoneGap) είναι ένα πλαίσιο (framework) ανάπτυξης εφαρμογών για κινητά που δημιουργήθηκε αρχικά από τη Nitobi. Η Adobe Systems αγόρασε τη Nitobi το 2011, το ξανασχεδίασε ως PhoneGap και αργότερα κυκλοφόρησε μια έκδοση ανοιχτού κώδικα του λογισμικού. Το Apache Cordova επιτρέπει στους προγραμματιστές να δημιουργούν εφαρμογές για κινητές συσκευές χρησιμοποιώντας CSS3, HTML5 και JavaScript, αντί να βασίζονται σε API συγκεκριμένα για πλατφόρμες, όπως σε Android, iOS ή Windows Phone. Επιτρέπει την συμπλήρωση του κώδικα CSS, HTML και JavaScript ανάλογα με την πλατφόρμα της συσκευής, ενώ παράλληλα επεκτείνει τις λειτουργίες για μεγαλύτερη συμβατότητα. Οι εφαρμογές που προκύπτουν είναι υβριδικές, πράγμα που σημαίνει ότι δεν είναι ούτε πραγματικά εγγενείς (native) εφαρμογές για κινητά (επειδή όλες οι γραφικές διεπαφές γίνονται μέσω προβολών web αντί για το περιβάλλον του UI της πλατφόρμας), αλλά ούτε και καθαρά web-based (επειδή δεν είναι απλά web εφαρμογές, αλλά κάνουν χρήση και native APIs της πλατφόρμας για άμεση επικοινωνία με το λειτουργικό σύστημα).

Ο πυρήνας των εφαρμογών που σχεδιάζονται με αυτή τη μέθοδο χρησιμοποιεί CSS3 και HTML5 για την γραφική διεπαφή και JavaScript για τη λογική τους. Η CCS3 είναι η τελευταία έκδοση της CSS, έχοντας εμπλουτιστεί με λειτουργίες για τη καλύτερη διαχείριση κινούμενων εικόνων και γραφικών (2D/3D Μετασχηματισμοί, animations κλπ.). Η HTML5 διαθέτει

[29] https://cordova.apache.org/

επίσης νέα χαρακτηριστικά σε σχέση με την απλή HTML, προσφέρει πλήρη υποστήριξη για CSS3, ήχο, βίντεο, υποστήριξη για γραφικά 2D και 3D, βελτιωμένο χειρισμό σφαλμάτων, επιτρέπει τοπική αποθήκευση στην συσκευή του πελάτη, προσφέρει τοπική βάση δεδομένων SQL και μπορεί να τρέξει σε οποιαδήποτε συσκευή. Το Apache Cordova μπορεί να επεκταθεί με εγγενείς (native) προσθήκες για την συσκευή για την οποία προορίζεται η εφαρμογή, επιτρέποντας στους προγραμματιστές να προσθέσουν περισσότερες λειτουργίες που μπορούν να καλούνται από τη JavaScript. Καθίσταται έτσι εφικτή η επικοινωνία απευθείας μεταξύ του λειτουργικού συστήματος της συσκευής και της σελίδας HTML5. Αυτά τα πρόσθετα επιτρέπουν την πρόσβαση στο επιταχυνσιόμετρο, τη φωτογραφική μηχανή, την πυξίδα, το σύστημα αρχείων, το μικρόφωνο και πολλά άλλα. Ωστόσο, η χρήση τεχνολογιών βασισμένων στην HTML και JavaScript οδηγεί σε ορισμένες εφαρμογές του Apache Cordova να τρέχουν πιο αργά από τις εγγενείς εφαρμογές με παρόμοια λειτουργικότητα. Αυτό μπορεί να είναι ένα ζήτημα για κάποιες εφαρμογές που απαιτούν γρήγορη απόκριση και μικρούς χρόνους φόρτωσης (π.χ. παιχνίδια).

Κεφάλαιο 3 – Απαιτήσεις Συστήματος και Σχεδιασμός

3.1 Σενάριο Χρήσης

Η εφαρμογή έχει ως στόχο να διευκολύνει την πλοήγηση επισκεπτών σε μεγάλους εσωτερικούς χώρους, με τη χρήση αισθητήρων ανίχνευσης προσέγγισης που βρίσκονται τοποθετημένοι σε κάθε χώρο του κτηρίου. Μέσω της Bluetooth συνδεσιμότητας της κινητής συσκευής του χρήστη, είναι δυνατόν να εντοπίζονται αισθητήρες που βρίσκονται στο χώρο και να ταξινομούνται με βάση την απόσταση τους. Η τρέχουσα τοποθεσία του χρήστη και ο χώρος μέσα στο κτήριο που επιθυμεί να επισκεφτεί αποστέλλονται στο Back-End, όπου και εκτελείται ο αλγόριθμος προσδιορισμού της επιθυμητής διαδρομής. Αρχικά γίνεται αντιστοίχιση του κάθε αισθητήρα σε μία και μοναδική περιοχή, βάσει του αναγνωριστικού του. Αφού προσδιοριστεί η θέση του χρήστη στο χώρο, υπολογίζεται με βάση τον επιλεγμένο προορισμό η επόμενη τοποθεσία που ανήκει στην διαδρομή. Σε περίπτωση που η τοποθεσία του χρήστη δεν ανήκει σε αυτή, τότε και μόνο εξετάζονται αν υπάρχει γειτονική τοποθεσία η οποία αποτελεί μέρος της, ώστε να δοθεί νέα κατευθυντήρια οδηγία. Σε αντίθετη περίπτωση ο χρήστης θεωρούμε ότι έχει χάσει τον προσανατολισμό του, οπότε στέλνουμε αίτημα για βοήθεια στον διαχειριστή του συστήματος. Υπάρχει ένα αρχικό σημείο εισόδου και ένας ή περισσότεροι προορισμοί. Υποθέτουμε ότι ο επισκέπτης έχει εκδηλώσει τον ενδιαφέρον του να επισκεφθεί μία συγκεκριμένη τοποθεσία μέσα στο κτήριο, έχει πάρει άδεια για αυτό τον σκοπό και παράλληλα ο διαχειριστής του κτηρίου του έχει στείλει στο λογαριασμό του (π.χ. email) τις οδηγίες κατεύθυνσης που πρέπει να ακολουθήσει. Η διαδρομή για κάθε προορισμό έχει οριστεί εκ των προτέρων από τον διαχειριστή του συστήματος. Αυτή δεν είναι απαραίτητα η πιο σύντομη διαδρομή, αλλά μπορεί να είναι η πιο εύκολη ή η επιτρεπόμενη διαδρομή για επισκέπτες. Οι οδηγίες πλοήγησης που αποστέλλονται ικανοποιούν την παραπάνω συνθήκη.

Ωστόσο η μονομερής πλοήγηση του χρήστη δεν είναι το μόνο θέμα που μας απασχολεί. Η παρακολούθηση της πορείας των επισκεπτών σε κάθε χώρο του κτηρίου είναι λειτουργικότητα που περιλαμβάνεται στην εφαρμογή διαχείρισης. Αυτό είναι και το κύριο πλεονέκτημα που προσφέρει η ανάπτυξη της εφαρμογής στο ΥΝ. Ο διαχειριστής της υποδομής μπορεί μέσω του διαδικτύου σε πραγματικό χρόνο να παρακολουθεί την εξέλιξη της πλοήγησης κάθε επισκέπτη εξατομικευμένα και να ορίζει για αυτόν σε ποιους χώρους του κτηρίου επιτρέπεται να πάει. Σε περίπτωση που ο χρήστης χαθεί μέσα στο κτήριο ή βρεθεί σε τοποθεσία που απαγορεύεται η πρόσβαση, τότε ενημερώνεται αυτόματα τόσο ο ίδιος όσο και ο διαχειριστής του συστήματος. Η παραπάνω λειτουργικότητα συμπληρώνεται από μία ολοκληρωμένη υπηρεσία επικοινωνίας, που επιτρέπει την ανταλλαγή μηνυμάτων βοηθείας ανάμεσα στους χρήστες και τους διαχειριστές της υποδομής. Τέλος, το σύστημά μας στο ΥΝ δύναται να κρατάει ιστορικό πλοήγησης για κάθε χρήστη, στατιστικά πληρότητας χώρων κλπ.

3.2 Λειτουργικές και Μη Λειτουργικές Απαιτήσεις Συστήματος

Προτού μεταβούμε στο σχεδιασμό της εφαρμογής είναι χρήσιμο να ορίσουμε τις βασικές ομάδες χρηστών στους οποίους απευθύνεται και να αναλύσουμε τις απαιτήσεις τους. Ειδικότερα, διακρίνουμε δύο ομάδες χρηστών: **(1) τον διαχειριστή της εταιρείας και (2) τους επισκέπτες της**. Στη συνέχεια ορίζουμε τις απαιτήσεις του συστήματος, κάνοντας τον διαχωρισμό τους σε λειτουργικές και μη.

3.2.1 Λειτουργικές Απαιτήσεις

Ως λειτουργικές απαιτήσεις ορίζουμε τις διαδικασίες που πρέπει απαραίτητα να ικανοποιεί το σύστημα και συνδέονται άρρηκτα με την υλοποίησή του. Οι απαιτήσεις αυτές πιθανόν να είναι διαφορετικές για κάθε ομάδα χρηστών, συνεπώς θα πρέπει να ικανοποιούνται όλες προκειμένου να θεωρηθεί πλήρως χρηστικό το σύστημά μας. Αναλυτικότερα περιγράφουν τις λειτουργίες που αφορούν κάθε μια από τις παραπάνω δύο κατηγορίες χρηστών.

Διαχειριστής της εταιρίας στο υπολογιστικό νέφος

1. Εισαγωγή / Επεξεργασία / Διαγραφή χρήστη
2. Εισαγωγή / Επεξεργασία / Διαγραφή τοποθεσιών και διαδρομών
3. Διαχείριση δικαιωμάτων πρόσβασης ανά χρήστη και τοποθεσία
4. Προβολή τρέχουσας θέσης επισκεπτών
5. Προβολή στοιχείων δραστηριότητας χώρων
6. Προβολή ιστορικού χρηστών και τοποθεσιών
7. Αποστολή ειδοποιήσεων και επικοινωνία με τους επισκέπτες

Επισκέπτες

1. Είσοδος στο σύστημα
2. Επιλογή επιθυμητού προορισμού από τη λίστα των διαθέσιμων επιτρεπτών
3. Λήψη οδηγιών πλοήγησης
4. Προβολή ιστορικού πρόσφατων τοποθεσιών
5. Δυνατότητα επικοινωνίας με το διαχειριστή

3.2.2 Μη Λειτουργικές Απαιτήσεις

Η ικανοποίηση των απαιτήσεων αυτών δεν είναι αναγκαία για να επιτελέσει μια εφαρμογή την βασική της λειτουργικότητα. Ωστόσο ο βαθμός πλήρωσής τους επηρεάζει ανάλογα και την ποιότητα του τελικού προϊόντος, ιδιαίτερα αν πρόκειται για εμπορική εφαρμογή. Σε αυτές περιλαμβάνονται:

Απόδοση: Αφορά την ταχύτητα απόκρισης του συστήματος, υπό συνθήκες υψηλού φόρτου.

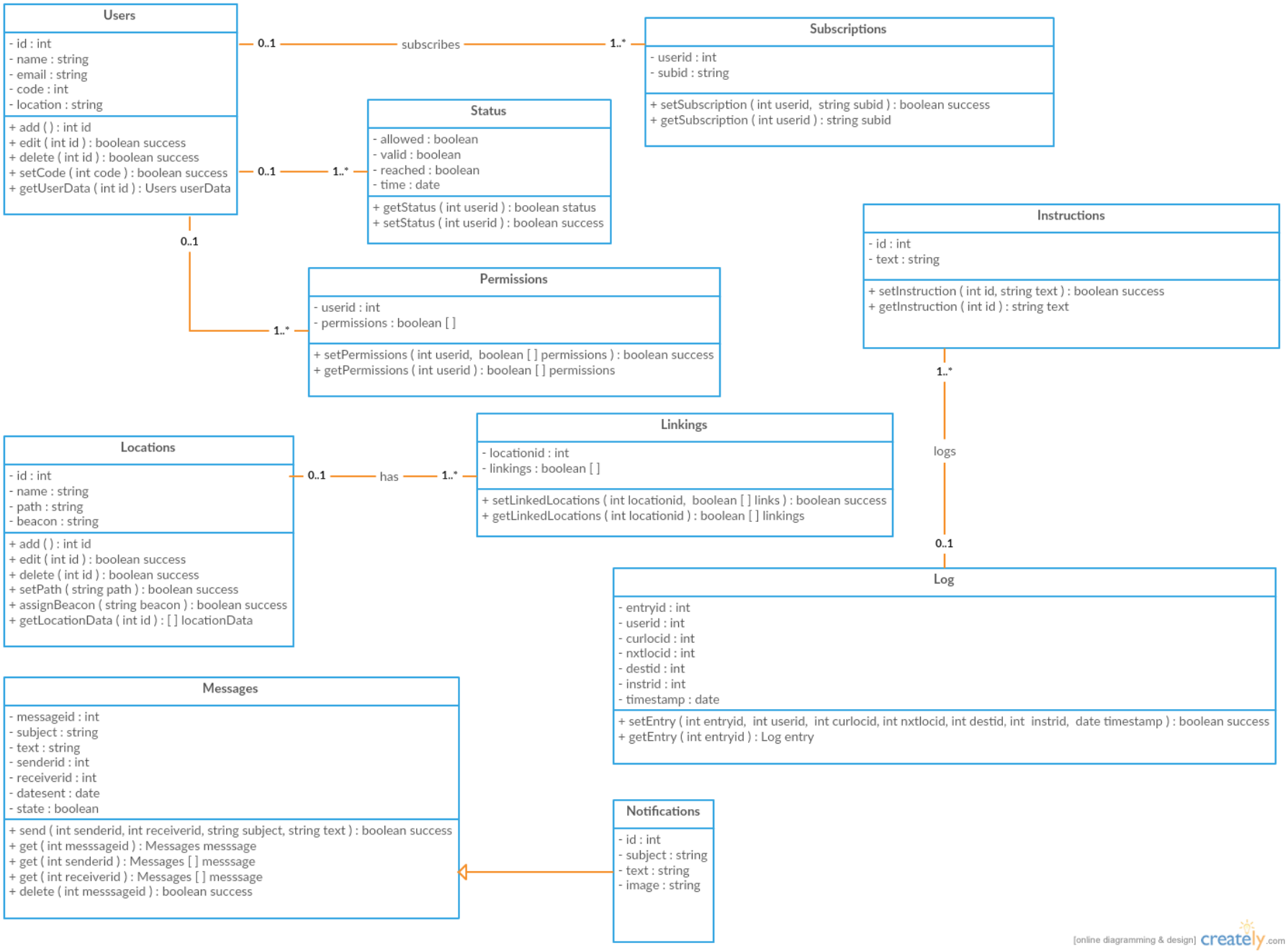
Επεκτασιμότητα: Αφορά την δυνατότητα της εφαρμογής να λαμβάνει μελλοντικές αναβαθμίσεις ώστε να βελτιώνεται και να επεκτείνεται η λειτουργικότητά της

Ασφάλεια: Αφορά την ασφάλεια χρηστών, δηλαδή την ασφαλή πρόσβασή τους στο σύστημα και την προστασία από μη εξουσιοδοτημένους χρήστες, της ταυτότητάς τους, των προσωπικών τους δεδομένων αλλά και του ιστορικού χρήσης της εφαρμογής. Παράλληλα, αφορά και την προστασία της εφαρμογής συνολικά και την αποτροπή πρόσβασης σε υπηρεσίες και δεδομένα, καθώς του δικτύου από μη εξουσιοδοτημένους χρήστες. Μέθοδοι μέσω των οποίων επιτυγχάνεται ασφάλεια δεδομένων και δικτύου περιλαμβάνουν την κρυπτογράφηση των δεδομένων της εφαρμογής τοπικά στην συσκευή, ενώ ασφάλεια δικτύου επιτυγχάνεται με την χρήση αφαλών μεθόδων επικοινωνίας για την ανταλλαγή πληροφορίας μέσω του διαδικτύου. Η ασφάλεια χρηστών και εφαρμογής επιτυγχάνεται με την χρήση ασφαλών πρωτοκόλλων πρόσβασης σε υπηρεσίες όπως Oauth 2.0.

Χρηστικότητα: Καθορίζει πόσο εύκολο είναι στη χρήση του το σύστημα. Στην κατηγορία αυτή ανήκουν χαρακτηριστικά όπως γραφικές διεπαφές, μενού βοηθείας, προηγμένες λειτουργίες ελέγχου μέσω αφής είτε φωνητικών εντολών και οτιδήποτε άλλο έχει ως στόχο να βελτιώσει την εμπειρία χρήσης της εφαρμογής.

Χρόνος λειτουργικότητας: Στις εφαρμογές ιστού είναι σημαντικό το σύστημα να είναι πάντοτε σε ετοιμότητα (online) και διαθέσιμο προς χρήση, συνεπώς θέλουμε όσο το δυνατόν μικρότερα διαστήματα μη απόκρισης αυτού (downtime). Κάτι τέτοιο ενδέχεται να συμβεί λόγω εργασιών συντήρησης είτε από την πλευρά του παρόχου του υπολογιστικού νέφους, είτε από τους προγραμματιστές της εφαρμογής ιστού λόγω αλλαγών στον κώδικα του back end.

Όπως έχουμε ήδη αναφέρει στο προηγούμενο κεφάλαιο, η χρήση της Υπηρεσιοκεντρικής Αρχιτεκτονικής επιτρέπει την ικανοποίηση των παραπάνω απαιτήσεων, καθώς διευκολύνει σημαντικά την συντήρηση και την επέκταση των εφαρμογών μέσω της χρήσης ανεξάρτητων υπηρεσιών. Επιπλέον, η συγκεκριμένη αρχιτεκτονική επιτρέπει τον διαμοιρασμό του φόρτου εργασίας σε διαφορετικές εικονικές μηχανές, καθώς οι υπηρεσίες δεν είναι απαραίτητο να φιλοξενούνται στην ίδια εικονική μηχανή, αυξάνοντας τη σταθερότητα του συστήματος.



Εικόνα 16: Διάγραμμα κλάσεων (Class Diagram)

3.3 Διάγραμμα Κλάσεων

Προκειμένου να κατανοήσουμε καλύτερα τις λειτουργίες του συστήματός μας καθώς και τις συσχετίσεις των υπηρεσιών που το απαρτίζουν, θα παραθέσουμε μια σύντομη περιγραφή των κλάσεων που δομούν την εφαρμογή, όπως αυτές αποτυπώνονται και στην Εικόνα 16. Πιο συγκεκριμένα:

Κλάση Χρηστών (Class Users)

Η κλάση αυτή περιλαμβάνει τα στοιχεία που συνθέτουν το προφίλ κάθε χρήστη της εφαρμογής (π.χ. όνομα, email, password), καθώς και τις απαραίτητες μεθόδους για την διαχείρισή τους.

Κλάση Συνδρομών (Class Subscriptions)

Η κλάση παρέχει βοηθητική λειτουργικότητα κατά την δημιουργία χρηστών, αποθηκεύοντας πληροφορίες της διεπαφής NGSI-9 για το χειρισμό των συμβάντων αυτών από τον Context Broker. Η διεπαφή NGSI-9 χρησιμοποιείται για πληροφορίες διαθεσιμότητας σχετικά με τις οντότητες και τα χαρακτηριστικά τους. Όταν ένας νέος χρήστης εγγράφεται στο σύστημα από τον διαχειριστή, τότε δημιουργείται και μια νέα οντότητα στην Υπηρεσία Διαχείρισης Συμβάντων και Συνδρομών (βλ. § 3.4), ενώ παράλληλα επιστρέφεται ένα μοναδικό αναγνωριστικό συνδρομής (subscription id). Με βάση το αναγνωριστικό συνδρομής, θα ενημερώνεται από εκείνη τη στιγμή και στο εξής το σύστημα για οποιαδήποτε αλλαγή στην θέση ή τον επιλεγμένο προορισμό του χρήστη που χειρίζεται την εφαρμογή στο τηλέφωνο και έχει αντιστοιχιστεί με το συγκεκριμένο αναγνωριστικό συνδρομής.

Κλάση Αδειών (Class Permissions)

Περιλαμβάνει την απαραίτητη πληροφορία για την διαχείριση των δικαιωμάτων πρόσβασης που έχει ο κάθε χρήστης ανά χώρο. Η πληροφορία αναπαρίσταται μέσω ενός πίνακα, όπου το κλειδί κάθε γραμμής αντιπροσωπεύει το αναγνωριστικό κάθε τοποθεσίας, ενώ η τιμή αυτού το αντίστοιχο δικαίωμα πρόσβασης, δηλαδή αν επιτρέπεται η πρόσβαση στο χώρο αυτό ή όχι. Κατά την πλοήγηση του επισκέπτη, κάθε φορά που αυτός εισέρχεται σε ένα νέο χώρο, ελέγχεται η παραπάνω συνθήκη και σε περίπτωση που δεν επιτρέπεται η πρόσβαση του στο χώρο αυτό, τότε αμέσως ενημερώνεται ο διαχειριστής μέσω κατάλληλης γραφικής ειδοποίησης που εμφανίζεται στη σελίδα διαχείρισης χρηστών.

Κλάση Κατάστασης Πορείας Χρήστη (Class Status)

Οι μέθοδοι της κλάσης αυτής καλούνται όποτε παρατηρείται αλλαγή στην τοποθεσία του χρήστη, επομένως και εκτελείται ο αλγόριθμος προσδιορισμού της επόμενης τοποθεσίας ως προς την επιθυμητή διαδρομή. Σκοπός είναι να ενημερώνεται κάθε στιγμή τόσο ο χρήστης όσο και ο διαχειριστής για την κατάσταση της πλοήγησης. Πιθανές περιπτώσεις κατάστασης

είναι: ο χρήστης να έχει χαθεί, να βρίσκεται σε μη επιτρεπτή τοποθεσία ή να έχει φτάσει στον προορισμό του.

Κλάση Τοποθεσιών (Class Locations)

Στην κλάση κρατάται όλη η πληροφορία για τις τοποθεσίες της υποδομής και τη διαχείριση αυτών (π.χ. όνομα και αναγνωριστικό τοποθεσίας, αναγνωριστικό αισθητήρα, προκαθορισμένη διαδρομή), καθώς και οι αντίστοιχες βοηθητικές μέθοδοι.

Κλάση Συνδέσεων Τοποθεσιών (Class Linkings)

Περιλαμβάνει πληροφορία για τις τοποθεσίες που συνορεύουν με μια δεδομένη περιοχή. Η πληροφορία αναπαρίσταται μέσω ενός πίνακα, όπου το κλειδί κάθε γραμμής αντιπροσωπεύει το αναγνωριστικό κάθε τοποθεσίας, ενώ η τιμή αυτού την ύπαρξη της αντίστοιχης σύνδεσης, εφόσον αυτή είναι ορισμένη από τον διαχειριστή.

Κλάση Οδηγιών Πλοήγησης (Class Instructions)

Εφόσον αναγνωριστεί η τοποθεσία του επισκέπτη στο κτήριο, τότε μέσω της λογικής της εφαρμογής για την πλοήγηση, θα υπολογιστεί η αμέσως επόμενη τοποθεσία που θα πρέπει κάποιος να κατευθυνθεί, η οποία και θα συνορεύει με την τρέχουσα τοποθεσία του. Συνεπώς υπάρχει μια σχέση και μία αλληλεξάρτηση που συνδέει την τρέχουσα τοποθεσία με την επόμενη. Αν για παράδειγμα είμαστε στη θέση Α και θα πρέπει να πάμε στην θέση Β, τότε θα υπάρχει μία οδηγία που μας δώσει την απαραίτητη πληροφορία για το πως θα πάμε από την θέση Α στην Β, π.χ. να πάμε ευθεία. Τέτοιες λεκτικές οδηγίες, έχουν την μορφή τριπλέτας π.χ. [Α→Β, Ευθεία] και έχουν οριστεί από την διαχειριστή του συστήματος για κάθε πιθανή αλληλουχία διαδοχικών τοποθεσιών μέσα στο κτήριο. Οι μέθοδοι της Κλάσης Οδηγιών Πλοήγησης διαχειρίζονται την επικοινωνία με τη βάση δεδομένων, όπου βρίσκονται αποθήκευες οι παραπάνω οδηγίες.

Κλάση Μηνυμάτων (Class Messages)

Η εφαρμογή μας έχει μια απλή υλοποίηση υπηρεσίας ανταλλαγής μηνυμάτων από τους χρήστες προς τους διαχειριστές του συστήματος. Πέραν των αυτοματοποιημένων ειδοποιήσεων συστήματος, επιτρέπεται και η αποστολή προσωπικών μηνυμάτων προς κάποιο χρήστη. Τα μηνύματα αποθηκεύονται στη βάση δεδομένων και περιέχουν πληροφορία για τον αποστολέα, τον παραλήπτη, τον τίτλο και το κείμενο αυτών. Μέσω AJAX Long Polling από τα Front End, ερευνάται αν υπάρχουν νέα μηνύματα στη βάση με παραλήπτη κάποιον συγκεκριμένο χρήστη ώστε να προωθηθεί το μήνυμα στη συνέχεια προς την συσκευή του χρήστη αυτού. Οι μέθοδοί της Κλάσης Μηνυμάτων διευκολύνουν τη διαχείριση και ανάκτηση των μηνυμάτων από τη βάση δεδομένων, επιτρέποντας τη σύνθεση, αλλά και αναζήτηση αυτών βάσει τίτλου, περιεχομένου, ονόματος παραλήπτη, ημερομηνίας αποστολής κλπ.

Κλάση Ειδοποιήσεων Συστήματος (Class Notifications)

Αποτελεί υπο-κλάση των «Μηνυμάτων» (Class Messages) επεκτείνοντας κάποιες από τις λειτουργίες τους. Οι ειδοποιήσεις είναι ουσιαστικά μηνύματα που αποστέλλονται αυτόματα από το σύστημα είτε στους χρήστες είτε στο διαχειριστή, ανάλογα με ξαφνικά περιστατικά που ενδέχεται να προκύψουν (π.χ. αν ο χρήστης χαθεί τότε αποστέλλεται αυτόματα μία ειδοποίηση στο διαχειριστή).

Κλάση Εγγραφών (Class Log)

Σκοπός της κλάσης είναι η αποθήκευση στη βάση δεδομένων εγγραφών, κάθε φορά που πραγματοποιείται κάποια αλλαγή στην τοποθεσία των χρηστών. Έτσι αποθηκεύονται αναλυτικά στοιχεία όπως το αναγνωριστικό του χρήστη, η χρονική στιγμή που συνέβη η αλλαγή θέσης του, το όνομα αυτής και η εγκυρότητά της, ο επιλεγμένος προορισμός, η επόμενη αποδεκτή τοποθεσία στην διαδρομή, η οδηγία που εστάλη στην εφαρμογή κ.α. Τα στοιχεία αυτά χρησιμοποιούνται αργότερα για την ανάκτηση του ιστορικού χρηστών και τοποθεσιών και για τον έλεγχο της κινητικότητας ανά χώρο.

3.4 Διαγράμματα Περιπτώσεων Χρήσης

Έχοντας περιγράψει στην Ενότητα 3.1.1 τις λειτουργικές απαιτήσεις χρηστών ανά κατηγορία χρήστη και προσπαθώντας να τις ομαδοποιήσουμε, κάνουμε χρήση Διαγραμμάτων Περιπτώσεων Χρήσης της UML[30] (Use Case Diagrams) για την καλύτερη αποτύπωσή τους. Αναλυτικότερα, για το διαχειριστή ιδιωτικού νέφους, παρατηρούμε απαιτήσεις αφορούν τη διαχείριση χρηστών είναι οι εξής:

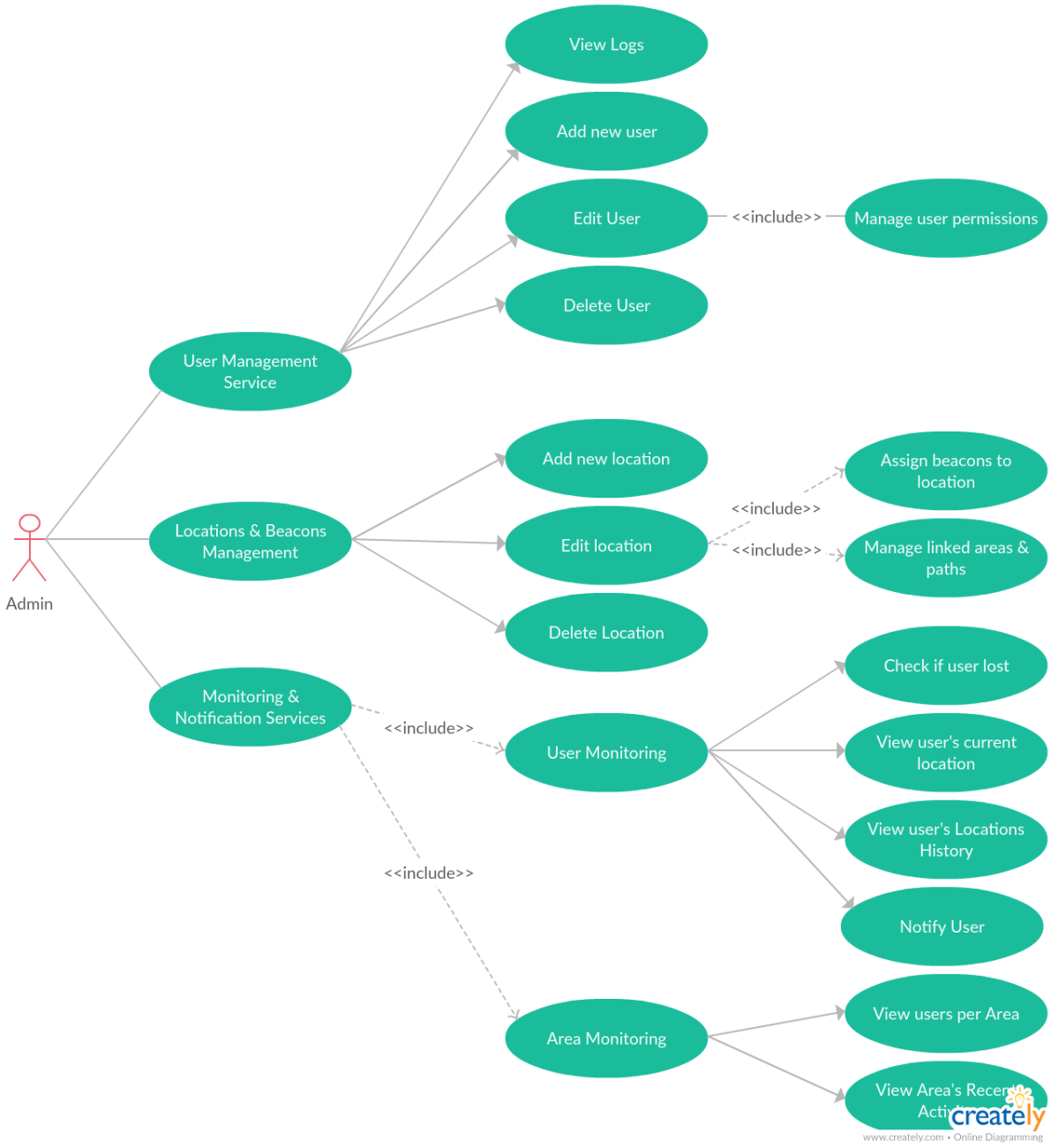
- Εισαγωγή/Επεξεργασία/Διαγραφή
- Διαχείριση Δικαιωμάτων Πρόσβασης
- Προβολή Τρέχουσας Θέσης
- Προβολή Ιστορικού
- Αποστολή Ειδοποιήσεων

Επομένως τις εντάσσουμε σε μία Υπηρεσία Διαχείρισης Χρηστών. Αντίστοιχα αποφασίσαμε να έχουμε την Υπηρεσία Διαχείρισης Χώρων που θα καλύπτει τις παρακάτω απαιτήσεις:

- Εισαγωγή/Επεξεργασία/Διαγραφή
- Καθορισμός Αποδεκτών Διαδρομών
- Συνδέσεις με Γειτονικούς Χώρους
- Συσχετισμός Χώρου με Αισθητήρα

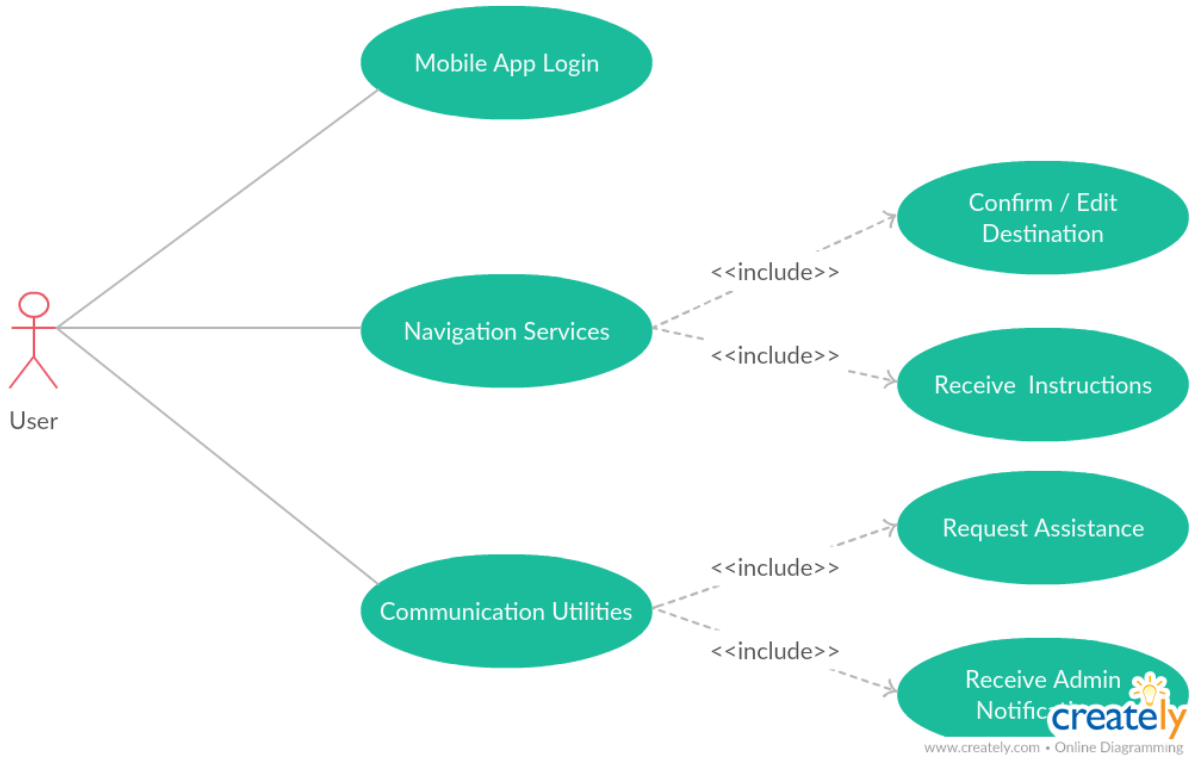
[30] https://msdn.microsoft.com/en-us/library/dd409432.aspx

Μία τρίτη κατηγορία υπηρεσιών θα μπορούσε να θεωρηθεί η Παρακολούθηση της Δραστηριότητας των Χώρων και των Επισκεπτών τους. Σε αυτήν περιλαμβάνονται ειδοποιήσεις συστήματος που αφορούν τα δικαιώματα των χρηστών σε σχέση με την τοποθεσία τους. (π.χ. αν ο χρήστης έχει χαθεί ή βρίσκεται σε μη επιτρεπτή τοποθεσία). Παρά την απεικόνιση τους ως ξεχωριστή υπηρεσία στο διάγραμμα, οι ενέργειες αυτές στην υλοποίηση της λογικής της εφαρμογής, περιλαμβάνονται στο ευρύτερο πλαίσιο της διαχείρισης Χρηστών και Τοποθεσιών. Με βάση τα παραπάνω, καταλήγουμε στο διάγραμμα της Εικόνας 17:



Εικόνα 17: Διάγραμμα περιπτώσεων χρήσης (Use case diagram) για το διαχειριστή χρήστη.

Αντίστοιχη λογική εφαρμόζουμε και για την απεικόνιση (Εικόνα 18) των λειτουργικών απαιτήσεων των χρηστών – επισκεπτών, όπως αυτές περιγράφησαν στην Ενότητα 3.1.1.

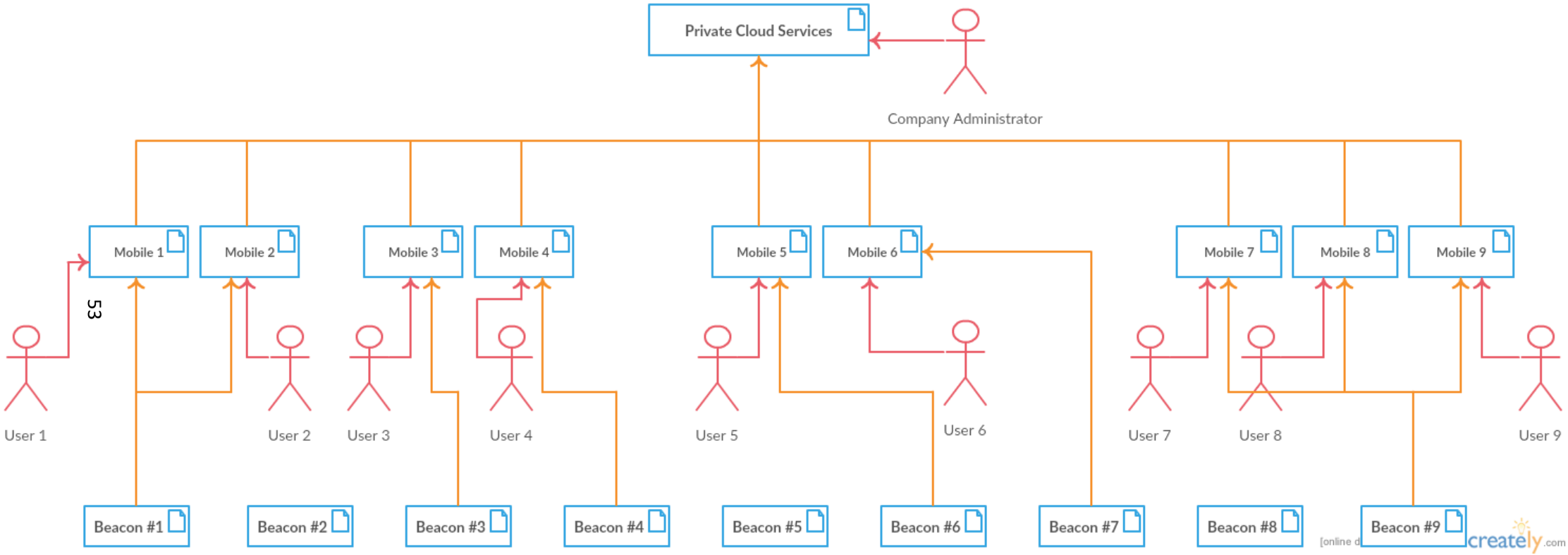


Εικόνα 18: Διάγραμμα περιπτώσεων χρήσης (Use case diagram) για τον επισκέπτη χρήστη.

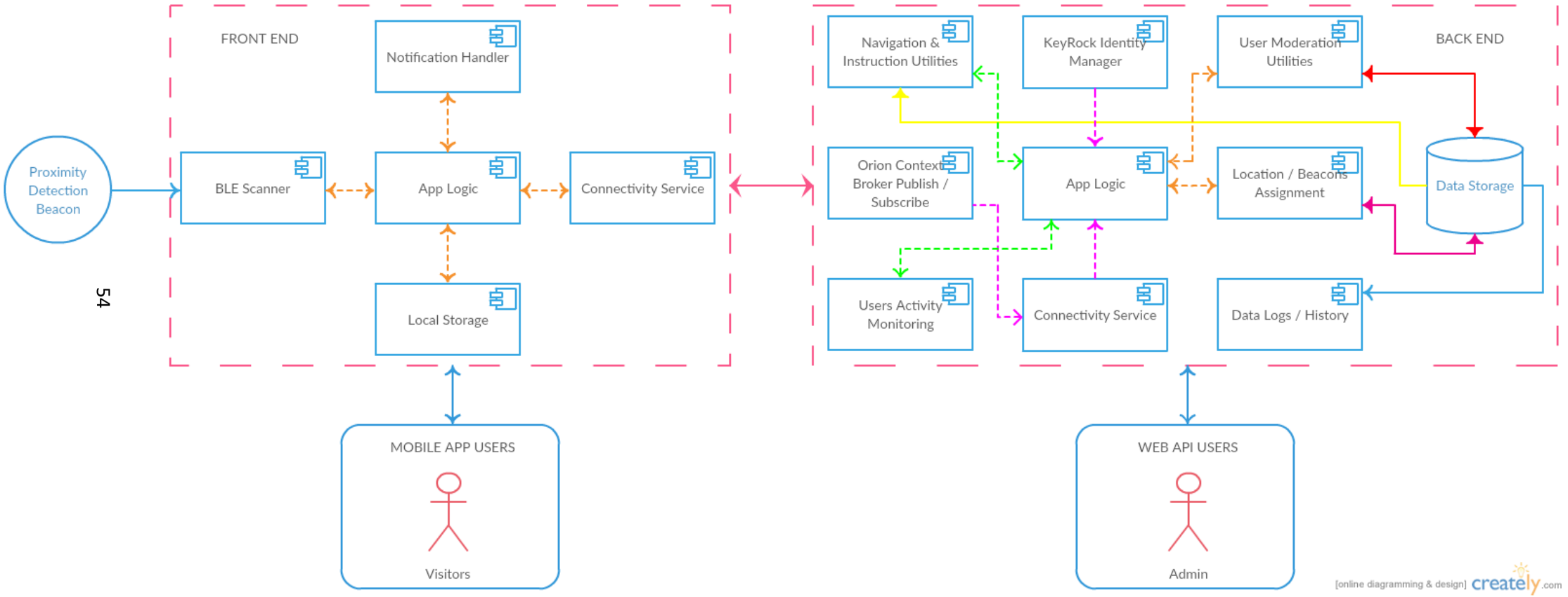
3.5 Διάγραμμα Τοπολογίας

Το επόμενο βήμα είναι η κατασκευή του διαγράμματος τοπολογίας της εφαρμογής (Deployment Diagram). Το διάγραμμα αυτό μας δίνει ένα γενικό πλάνο των φυσικών οντοτήτων της εφαρμογής και τον τρόπο αλληλεπίδρασής τους, με βάση το σενάριο χρήσης που περιγράψαμε στην Εισαγωγή (§ 1.2). Έτσι λοιπόν βλέπουμε, στην Εικόνα 18, τους αισθητήρες (beacons) ανίχνευσης προσέγγισης να είναι τοποθετημένοι στους διάφορους χώρους. Τα κινητά τηλέφωνα των επισκεπτών μέσω της εφαρμογής, με τη βοήθεια της Bluetooth συνδεσιμότητας, εντοπίζουν ένα ή περισσότερα beacons ανάλογα με το αν βρίσκονται στην εμβέλειά τους. Κατόπιν το αναγνωριστικό του πλησιέστερου αισθητήρα αποστέλλεται στην υποδομή μας στο υπολογιστικό νέφος. Με αυτό τον τρόπο, ο μεν επισκέπτης λαμβάνει στην εφαρμογή την επιθυμητή οδηγία κατεύθυνσης, ο δε διαχειριστής είναι σε θέση να παρακολουθεί σε πραγματικό χώρο τη μεταβολή της θέση κάθε επισκέπτη στο κτήριο.

Εικόνα 19: Διάγραμμα τοπολογίας συστήματος (Deployment diagram)



Εικόνα 20: Διάγραμμα αρχιτεκτονικής συστήματος (Architectural Diagram)



3.6 Διάγραμμα Αρχιτεκτονικής

Στη συνέχεια (Εικόνα 20) παρουσιάζουμε ένα πιο εξειδικευμένο διάγραμμα που αφορά την αρχιτεκτονική της εφαρμογής, ενώ κάνουμε και μια σύντομη περιγραφή των τμημάτων αυτής. Ουσιαστικά ακολουθούμε την γενική αρχιτεκτονική που παρουσιάσαμε στο προηγούμενο κεφάλαιο (βλ. § 2.8) και την εξειδικεύουμε ακόμα περισσότερο για να ανταποκρίνεται στις απαιτήσεις της εφαρμογής μας. Αναλυτική περιγραφή των επιμέρους υπηρεσιών θα κάνουμε στο επόμενο κεφάλαιο.

3.6.1 Σύστημα Διεπαφής Τελικού Χρήστη (Front-End)

Το σύστημα διεπαφής τελικού χρήστη επιτρέπει την επικοινωνία της γραφικής διεπαφής για το τηλέφωνο, με τον πυρήνα της εφαρμογής που υλοποιείται στο υπολογιστικό νέφος. Οι κύριες συνιστώσες της είναι οι ακόλουθες:

Λογική εφαρμογής (Application Logic)

Περιέχει τον κώδικα, ο οποίος χρησιμοποιείται για την επικοινωνία με τον πυρήνα του συστήματός μας. Το τμήμα αυτό της εφαρμογής είναι υπεύθυνο για την ενορχήστρωση των επιμέρους τμημάτων της. Πιο αναλυτικά η λογική της εφαρμογής επικοινωνεί με όλα τα υπόλοιπα τμήματα του Front-End με προκαθορισμένη σειρά, επιτελώντας τις ακόλουθες ενέργειες:

- Κατά την εκκίνηση της εφαρμογής πραγματοποιείται έλεγχος στον τοπικό χώρο αποθήκευσης της συσκευής, προκειμένου να ανακτηθούν αν υπάρχουν τα στοιχεία πρόσβασης του χρήστη.
- Αμέσως μετά, μέσω της υπηρεσίας διασύνδεσης επιτυγχάνεται μια συνεδρία (session) ανάμεσα στην εφαρμογή του χρήστη και τον εξυπηρετητή στο Back-End.
- Παράλληλα, ενεργοποιείται και ο Ανιχνευτής BLE συσκευών, ο οποίος σαρώνει συνεχώς το χώρο γύρω του για BLE συσκευές. Αν κάποια από αυτές βρεθεί στην εμβέλεια του και έχει επιτευχθεί και σύνδεση του χρήστη κατά το προηγούμενο βήμα, τότε και μόνο μέσω της υπηρεσίας διασύνδεσης, αποστέλλονται τα δεδομένα των BLE αισθητήρων στο Back-End.

Τοπικός Χώρος Αποθήκευσης (Local Storage)[31]

Μέσω αυτής της υπηρεσίας, η εφαρμογή αποθηκεύσει τα απαραίτητα δεδομένα πρόσβασης για αυτοματοποιημένη είσοδο. Όπως έχουμε ήδη αναφέρει η εφαρμογή μας για

[31] http://www.w3schools.com/html/html5_webstorage.asp

κινητές συσκευές είναι γραμμένη στο μεγαλύτερο μέρος της σε HTML5, ενώ χρησιμοποιήθηκε το Apache Cordova (§ 2.9) για να μετατραπεί σε ανεξάρτητη εφαρμογή. Η HTML5 λοιπόν διαθέτει μία δική της μικρή βάση δεδομένων που μπορεί να αποθηκεύσει τοπικά στη συσκευή διάφορες πληροφορίες που μας ενδιαφέρουν. Όταν ο χρήστης εισάγει κάποιον κωδικό πρόσβασης, αυτός αποθηκεύεται τοπικά στην εφαρμογή μας, ώστε την επόμενη φορά να γίνει προσπάθεια αυτοματοποιημένης εισόδου χωρίς να ζητηθούν ξανά τα στοιχεία του. Φυσικά αν η επαλήθευση των στοιχείων αποτύχει στο Back-End τότε αυτά θα ζητηθούν εκ νέου άσχετα αν υπάρχουν ήδη αποθηκευμένα στη συσκευή.

Ανιχνευτής συσκευών BLE (BLE Scanner)

Είναι μια βιβλιοθήκη που επιτρέπει την αναγνώριση BLE συσκευών από την εφαρμογή. Χρησιμοποιήθηκε το Estimote[32] API για Apache Cordova, παρέχοντάς μας εξειδικευμένες λειτουργίες για τους αισθητήρες αυτής της εταιρίας. Το API αυτό μεταξύ των άλλων κάνει και τα εξής:

- Ενεργοποιεί αυτόματα ή μετά από προτροπή του χρήστη τον πομποδέκτη Bluetooth μιας κινητής συσκευής.
- Σαρώνει κατ' εξακολούθηση την γύρω περιοχή για BLE συσκευές εντός εμβέλειας.
- Ταξινομεί αυτόματα τα αποτελέσματα σάρωσης κατά αύξουσα απόσταση.
- Για κάθε αισθητήρα που εντοπίστηκε μπορεί και διαβάζει την Διεύθυνση MAC, το επίπεδο μπαταρίας του, την απόστασή του από την συσκευή κατά προσέγγιση βάσει της ισχύος σήματος, καθώς και οποιαδήποτε άλλη μέτρηση πραγματοποιεί ο αισθητήρας (π.χ. θερμοκρασία, επιτάχυνση, φωτεινότητα κλπ.)

Υπηρεσία Διασύνδεσης (Connectivity Service)

Η υπηρεσία είναι υπεύθυνη για κάθε μορφής επικοινωνία της εφαρμογής για κινητές συσκευές με το Back-End. Τα δεδομένα που ανταλλάσσονται πακετάρονται σε μορφή JSON, ενώ η επικοινωνία με το Back-End γίνεται ασύγχρονα μέσω κλήσεων AJAX. Επικοινωνία της εφαρμογής μας με το Back-End γίνεται στις εξής τρεις περιπτώσεις:

- Κατά την σύνδεση στην εφαρμογή ώστε να επαληθευτεί η ταυτότητα του χρήστη: Τα δεδομένα που αποστέλλονται είναι το μοναδικό αναγνωριστικό χρήστη το οποίο επιτελεί ταυτόχρονα και ρόλο κωδικού πρόσβασης. Η πληροφορία που επιστρέφεται στη συσκευή είναι τα το ονοματεπώνυμο του χρήστη, οι επιτρεπόμενες διαδρομές, όπως τις όρισε ο διαχειριστής για αυτόν και το ιστορικό τοποθεσιών αυτού. Τα παραπάνω στοιχεία χρησιμοποιούνται στην γραφική διεπαφή της εφαρμογής πλοήγησης.

[32] https://github.com/evothings/phonegap-estimotebeacons/tree/master/examples/beacon-finder

- Κατά την σάρωση νέου αισθητήρα από τον Ανιχνευτή BLE συσκευών, ο οποίος είναι πλέον και ο πιο κοντινός προς τη συσκευή: Σε αυτή την περίπτωση αποστέλλονται στο Back-End το αναγνωριστικό του κοντινότερου αισθητήρα και ο επιλεγμένος προορισμός που όρισε ο χρήστης, ενώ ως απόκριση λαμβάνεται η οδηγία πλοήγησης.
- Κατά την λήψη λοιπών δεδομένων, ύστερα από απαίτηση του χρήστη ή του διαχειριστή (π.χ. φόρτωση δεδομένων ιστορικού πλοήγησης, αλλαγή προορισμού, ανταλλαγή μηνυμάτων μέσω της φόρμας επικοινωνίας με το διαχειριστή, λήψη ειδοποιήσεων κλπ.).

Υπηρεσία Ενημέρωσης (Notification Handler)

Υπηρεσία, μέσω της οποίας το τηλέφωνό μας είναι σε θέση να λαμβάνει ειδοποιήσεις από τον διαχειριστή σε πραγματικό χρόνο (push notifications), ακόμα και αν η εφαρμογή εκτελείται στο παρασκήνιο. Οι ειδοποιήσεις αυτές ενδέχεται να είναι αυτοματοποιημένες από το σύστημα ανάλογα με την πορεία που ακολουθεί ο χρήστης (π.χ. αν έχει χαθεί, αν έχει απομακρυνθεί από την προκαθορισμένη διαδρομή κλπ.). Ενδέχεται ωστόσο να είναι και μηνύματα που στέλνει ο ίδιος ο διαχειριστής. Για την λήψη των ειδοποιήσεων η υπηρεσία πραγματοποιεί AJAX Long Polling, στέλνει δηλαδή αιτήματα σε τακτά χρονικά διαστήματα προς τον εξυπηρετητή ερευνώντας αν υπάρχουν νέες μη αναγνωσμένες ενημερώσεις για λήψη, ενώ μετά την λήψη και ανάγνωση μιας ειδοποίησης το Back-End ενημερώνεται πως η ενημέρωση έχει αναγνωστεί. Προκειμένου να μπορούν να λαμβάνονται ειδοποιήσεις κατευθείαν στην συσκευή ακόμα και αν η εφαρμογή τρέχει στο παρασκήνιο χρησιμοποιήθηκε μία επέκταση (plug-in) του Apache Cordova το οποίο επιτρέπει τη λήψη push notifications από τη συσκευή, δηλαδή ειδοποιήσεων αρχικής οθόνης του Android με συγκεκριμένο εικονίδιο, ήχο και κείμενο όπως αυτά έχουν οριστεί από την εφαρμογή.

3.6.2 Υπηρεσίες Εκτελέσιμες στο Υπολογιστικό Νέφος (Back-End)

Περιλαμβάνει όλες εκείνες τις υπηρεσίες που συνθέτουν τον πυρήνα του συστήματός μας και εκτελούνται στο υπολογιστικό νέφος. Παραθέτουμε τις πιο σημαντικές παρακάτω:

Λογική εφαρμογής (Application Logic)

Κατ' αντιστοιχία με την λογική εφαρμογής στο Front-End, έτσι και εδώ πρόκειται για «την καρδιά» της εφαρμογής μας, αυτή την φορά του μέρους της που εκτελείται απομακρυσμένα στο υπολογιστικό νέφος. Στόχος της είναι η ενορχήστρωση των διάφορων υπηρεσιών και η ορθή μεταξύ τους επικοινωνία. Όποτε λαμβάνεται κάποιο αίτημα από το

Back-End, η Λογική της Εφαρμογής είναι υπεύθυνη για την δρομολόγησή του στην κατάλληλη υπηρεσία για την απάντηση του αιτήματος αυτού. Η υπηρεσία συνδέεται άρρηκτα με την Υπηρεσία Συνδέσεων (βλ. παρακάτω) στο Back-End, αφού τη χρησιμοποιεί σαν δίαυλο για την ανταλλαγή δεδομένων ανάμεσα στις υπόλοιπες υπηρεσίες στο ΥΝ. Για παράδειγμα εάν κάποιο αίτημα που λαμβάνεται αφορά την διαχείριση των χρηστών της εφαρμογής, μέσω της Λογικής της Εφαρμογής το αίτημα θα δρομολογηθεί προς την Υπηρεσία Διαχείρισης Χρηστών (βλ. παρακάτω). Εάν το αίτημα αφορά τη διαχείριση χώρων, ή την αποστολή οδηγιών πλοήγησης, τότε αυτό θα δρομολογηθεί στις αντίστοιχες υπηρεσίες. Τέλος η Λογική της Εφαρμογής εκτελεί και κάποιους βασικούς ελέγχους που αφορούν την ασφαλή χρήση της εφαρμογής από τους χρήστες και τον διαχειριστή του συστήματος. Συγκεκριμένα ελέγχεται εάν έχει επιτευχθεί συνεδρία (session) ανάμεσα στον χρήστη από τον οποίο προέρχεται κάποιο αίτημα και στην εφαρμογή στο Back-End. Έτσι αγνοούνται και δεν εξυπηρετούνται αιτήματα που προέρχονται από εξωτερικές μη ασφαλείς πηγές.

Υπηρεσία Ταυτοποίησης και Εξουσιοδότησης Χρηστών (KeyRock Identity Manager)

Πρόκειται για υπηρεσία παρεχόμενη από το FIWARE με σκοπό την εύκολη εγγραφή και σύνδεση στο σύστημα, μέσω χρήσης του πρωτοκόλλου OAuth2. Ουσιαστικά η υπηρεσία αυτή μας επιτρέπει να συνδεόμαστε στην δικτυακή εφαρμογή μέσω του FIWARE λογαριασμού μας, μη αποκαλύπτοντας τα στοιχεία πρόσβασής μας κατευθείαν στους εξυπηρετητές της δικτυακής εφαρμογής μας (βλ. § 4.2.1.1). Αντιθέτως το FIWARE είναι αυτό που δρα ως μεσάζοντας, παρέχοντας στην εφαρμογή μας τα στοιχεία χρήστη στα οποία αυτή επιτρέπεται να έχει πρόσβαση (email, όνομα, ρόλος π.χ. απλός χρήστης ή διαχειριστής κλπ.).

Υπηρεσία Διαχείρισης Χρηστών (User Moderation Utilities)

Η υπηρεσία παρέχει λειτουργικότητα για όλες τις απαραίτητες ενέργειες που αφορούν τη διαχείριση των χρηστών της εφαρμογής, συμπεριλαμβανομένων της δημιουργίας, επεξεργασίας, διαγραφής χρηστών, την αποθήκευση του ιστορικού πλοήγησής τους αλλά και τις επιτρεπόμενες τοποθεσίες που δύναται να επισκεφτεί ο χρήστης. Η υπηρεσία επικοινωνεί με μια κοινόχρηστη MySQL βάση δεδομένων όπου και αποθηκεύει όλες τις παραπάνω πληροφορίες, ενώ μέσω της ευρύτερης λογικής της εφαρμογής, οι πληροφορίες αυτές μπορούν και διαμοιράζονται και στις υπόλοιπες υπηρεσίες. Κατά την διαδικασία της πλοήγησης, για παράδειγμα, η εφαρμογή ανατρέχει στη βάση δεδομένων εξετάζοντας εάν η τρέχουσα τοποθεσία του επισκέπτη εντάσσεται στην λίστα των επιτρεπόμενων τοποθεσιών του προφίλ του (βλ. § 4.2.1.3 για τον αναλυτικό κατάλογο REST κλήσεων της υπηρεσίας).

Υπηρεσία Διαχείρισης Χώρων (Area / Beacon Assignment)

Η υπηρεσία παρέχει λειτουργικότητα για τη δημιουργία, επεξεργασία, διαγραφή τοποθεσιών, την συσχέτισή τους με αισθητήρες, τον ορισμό περιοχών που συνορεύουν με αυτές, καθώς και την δημιουργία προκαθορισμένων διαδρομών που οδηγούν στις συγκεκριμένες τοποθεσίες. Οι παραπάνω πληροφορίες αποθηκεύονται σε MySQL βάση δεδομένων και χρησιμοποιούνται κατά την διαδικασία της πλοήγησης (βλ. § 4.2.1.4 για τον αναλυτικό κατάλογο REST κλήσεων της υπηρεσίας).

Υπηρεσία Ελέγχου Δραστηριότητας (Users Activity Monitoring)

Η υπηρεσία παρέχει την δυνατότητα παρακολούθησης σε πραγματικό χρόνο της πορείας κάθε χρήστη μέσα στο κτήριο. Με χρήση κατάλληλων γραφικών ειδοποιήσεων που εμφανίζονται τόσο στην συσκευή του χρήστη όσο και στην εφαρμογή διαχείρισης, διαχειριστής και επισκέπτης ενημερώνονται σε περίπτωση εσφαλμένης πορείας ή πρόσβασης σε μη αποδεκτή τοποθεσία του τελευταίου, με βάση τα δικαιώματα που έχουν παραχωρηθεί από πριν. Η υπηρεσία αντλεί τις πληροφορίες αυτές (προφίλ χρήστη, δικαιώματα πρόσβασης, ονόματα τοποθεσιών, προκαθορισμένες διαδρομές κλπ.) από την κοινόχρηστη MySQL βάση δεδομένων, όπως αυτές αποθηκευτήκαν από τις αντίστοιχες υπηρεσίες Διαχείρισης Χρηστών και Χώρων. Αποθηκεύει επίσης πίσω σε αυτήν, πίνακες (logs) με το ιστορικό πλοήγησης κάθε χρήστη, καταγράφοντας ουσιαστικά τις διαδοχικές τοποθεσίες από τις οποίες αυτός πέρασε.

Υπηρεσία αποθήκευσης Δεδομένων (Data Storage)

Πρόκειται για μια κοινόχρηστη MySQL βάση δεδομένων, όπου διατηρούνται τα δεδομένα όλου του συστήματος, συμπεριλαμβανομένων αυτών των χρηστών και των τοποθεσιών, όπως αυτά αναλύθηκαν παραπάνω. Η βάση ακολουθεί το σχεσιακό μοντέλο, πράγμα που σημαίνει ότι τηρούνται όλες οι απαραίτητες συσχετίσεις μεταξύ των πινάκων της, ενώ με την χρήση κατάλληλων μεθόδων και εναυσμάτων (triggers) πραγματοποιούνται έλεγχοι εγκυρότητας κατά την εισαγωγή, επεξεργασία και διαγραφή καταχωρήσεων από αυτήν. Τα ερωτήματα προς αυτήν τίθενται από τις υπόλοιπες υπηρεσίες, ενώ για την απάντησή τους απαιτούνται συχνά συνδυαστικά αποτελέσματα από διαφορετικούς πίνακες. Για παράδειγμα κατά την πλοήγηση ενός επισκέπτη χρειαζόμαστε να συσχετίσουμε τις επιτρεπόμενες τοποθεσίες που έχουν οριστεί για το χρήστη αυτό, με την προκαθορισμένη διαδρομή που οδηγεί στον επιθυμητό του προορισμό (στοιχεία δηλαδή που υπάρχουν στους πίνακες "Χρήστες", "Δικαιώματα" και "Τοποθεσίες").

Υπηρεσία Διαχείρισης Συμβάντων και Συνδρομών (Orion Context Broker Publish-Subscribe Service)

Η υπηρεσία αυτή του FIWARE δρα ως μεσολαβητής για τις μετρήσεις των αισθητήρων θέσης που αποστέλλονται από το Front-End στο Back-End. Η αποστολή αλλά και η ανάγνωση των δεδομένων λαμβάνουν τη μορφή κλήσεων RESTful για τη διασυνδέση NGSI-10. Η διεπαφή NGSI-10 χρησιμοποιείται για την ανταλλαγή πληροφοριών σχετικά με τις οντότητες και τα χαρακτηριστικό τους, δηλαδή αξίες χαρακτηριστικών και μεταδεδομένα. Πιο συγκεκριμένα τα δεδομένα που εισέρχονται στον Context Broker αφορούν το αναγνωριστικό του κοντινότερου αισθητήρα και τον επιθυμητό προορισμό, όπως αυτά αποστέλλονται από την κινητή συσκευή του χρήστη. Όταν υπάρξει κάποια αλλαγή στις τιμές των δεδομένων αυτών, η Υπηρεσία Διαχείρισης Συμβάντων και Συνδρομών του FIWARE τα προωθεί στην Υπηρεσία Συνδέσεων του Back-End μας (βλ. παρακάτω) ενημερώνοντάς παράλληλα, μέσω της λογικής της εφαρμογής, για την αλλαγή.

Υπηρεσία Συνδέσεων (Connectivity Service)

Η υπηρεσία είναι υπεύθυνη για την επικοινωνία μεταξύ του Front-End και Back-End της εφαρμογής. Η επικοινωνία γίνεται με REST κλήσεις ενώ τα δεδομένα είναι σε μορφή JSON. Επικοινωνία με το Back-End έχουμε στις εξής περιπτώσεις:

- Κατά την προώθηση των πληροφοριών αισθητήρων από τον Context Broker στο Back-End. Τα δεδομένα αυτά είναι της μορφής NGSI-10, ενός JSON που αν αποκωδικοποιηθεί περιέχει πληροφορία της μορφής:

  ```
  $params['contextResponses'][index]['contextElement']['attributes'][
  index]['value']
  ```

  Η αναπαράσταση αυτή αντιστοιχεί σε ένα διάνυσμα, η κάθε διάσταση του οποίου δηλώνει τη βαθμίδα στην ιεραρχία των δεδομένων του Context Broker, όπως αυτά δομούνται σύμφωνα με το NGSI-10. Ακολουθεί παράδειγμα:

```
{
    "contextResponses": [
        {
            "contextElement": {
                "attributes": [
                    {
                        "name": "temperature",
                        "type": "centigrade",
                        "value": "23"
                    },
                    {
                        "name": "pressure",
                        "type": "mmHg",
                        "value": "720"
                    }
                ],
                "id": "Room1",
                "isPattern": "false",
                "type": "Room"
            },
            "statusCode": {
                "code": "200",
                "reasonPhrase": "OK"
            }
        }
    ]
}
```

Το παραπάνω τμήμα κώδικα αναφέρεται στις περιβαλλοντικές συνθήκες (θερμοκρασία και πίεση αέρα) που επικρατούν στα δωμάτια ενός ξενοδοχείου. Σύμφωνα με την ιεραρχία των 'contextResponses' → 'contextElement' → 'attributes' που έχουμε στο παράδειγμά μας, μπορούμε να διαβάσουμε τις τιμές της θερμοκρασίας και της πίεσης στο δωμάτιο με id = "Room1" χρησιμοποιώντας τις εξής εκφράσεις:

['contextResponses'][0]['contextElement']['attributes'][0]['value'] (θερμοκρασία)
['contextResponses'][0]['contextElement']['attributes'][1]['value'] (πίεση)

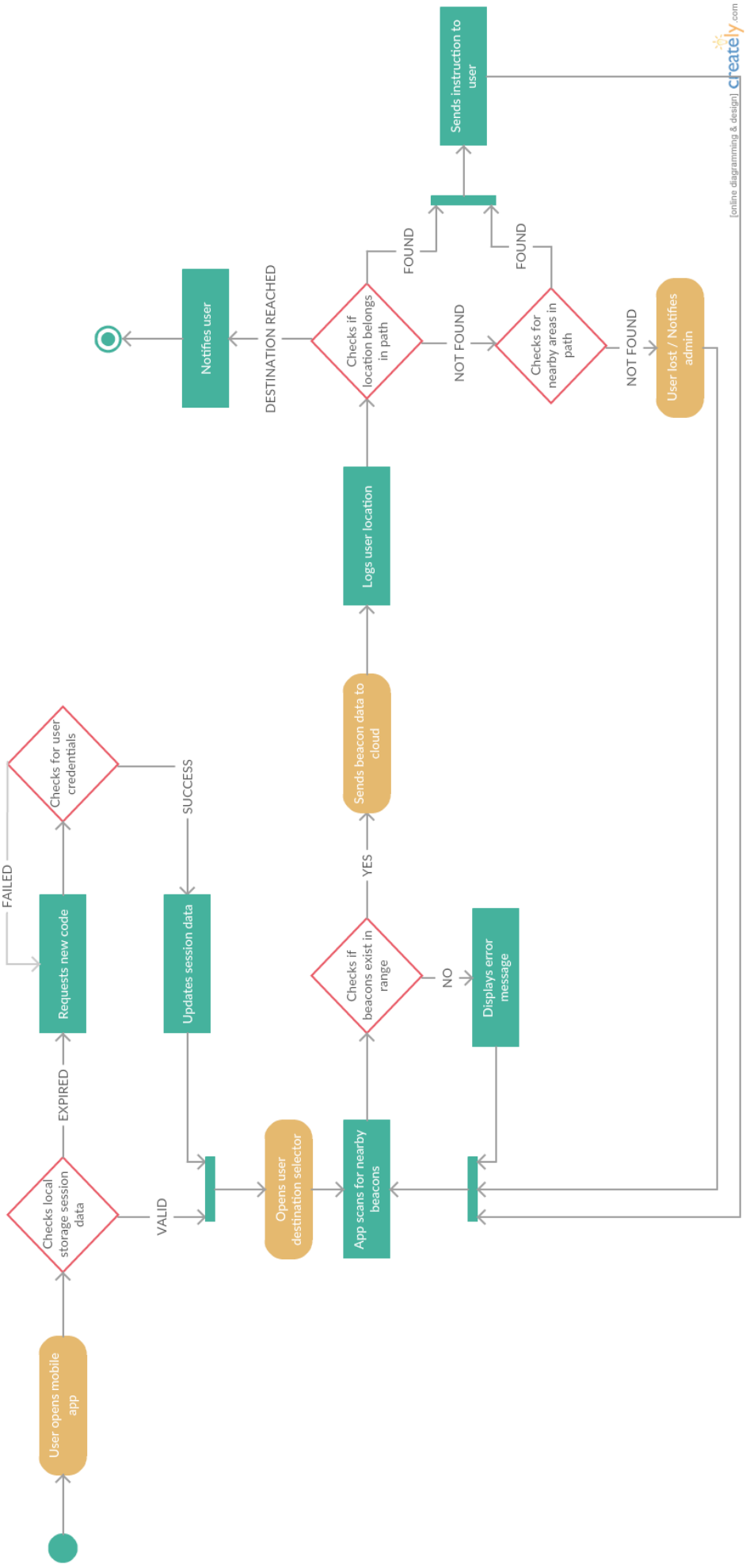
- Κατά την επικοινωνία των λοιπών υπηρεσιών με το Front-End, όπως εμφάνιση πληροφοριών χρηστών και τοποθεσιών στην εφαρμογή διαχείρισης, αποστολή οδηγιών πλοήγησης προς τον χρήστη, αποστολή ειδοποιήσεων κλπ.

3.7 Διαγράμματα Δραστηριοτήτων

Ακολουθούν τα διαγράμματα δραστηριοτήτων (Activity Diagrams) για τους χρήστες και τον διαχειριστή του συστήματος. Στόχο έχουν να παρουσιάσουν τα σημαντικότερα σενάρια χρήσης της εφαρμογής για την κάθε κατηγορία χρηστών. Στο διάγραμμα της Εικόνας 21 παρουσιάζεται η γενική ιδέα χρήσης της εφαρμογής από το τηλέφωνο ενός επισκέπτη. Για την καλύτερη κατανόηση των ενεργειών που απεικονίζονται, θα τις απαριθμήσουμε με τη σειρά που αυτές λαμβάνουν χώρα:

Διαδικασία σύνδεσης στην εφαρμογή

1. Ο χρήστης ανοίγει την εφαρμογή για κινητές συσκευές.
2. Η εφαρμογή αρχικά ελέγχει αν βρίσκονται αποθηκευμένα τοπικά στη συσκευή στοιχεία πρόσβασης από προηγούμενη είσοδο του χρήστη.
3. Αν υπάρχουν αποθηκευμένα στοιχεία πρόσβασης, τότε αυτά αποστέλλονται στην υποδομή μας στο υπολογιστικό νέφος για εξακρίβωση.
4. Εφόσον τα στοιχεία πρόσβασης που απεστάλησαν επαληθευτούν και πράγματι ισχύουν, ο χρήστης αποκτά πρόσβαση στις υπηρεσίες της εφαρμογής.
5. Διαφορετικά, του ζητείται να εισάγει έναν καινούριο κωδικό πρόσβασης. Ο κωδικός αυτός αποστέλλεται στο email του χρήστη κατόπιν συνεννόησης του με το διαχειριστή.
6. Εάν ο νέος κωδικός γίνει αποδεκτός ο χρήστης συνδέεται στην εφαρμογή, ενώ τα στοιχεία του αποθηκεύονται τοπικά στη συσκευή για τη διευκόλυνση μελλοντικής εισόδου.



Εικόνα 21: Σενάριο χρήσης της εφαρμογής για πλοήγηση από το τηλέφωνο

Διαδικασία πλοήγησης

1. Ο χρήστης καλείται να ενεργοποιήσει τον πομποδέκτη Bluetooth της συσκευής του.
2. Επιλέγει τον επιθυμητό προορισμό από το μενού της εφαρμογής, μέσα από μια λίστα με όλες τις επιτρεπόμενες τοποθεσίες που έχουν οριστεί από τον διαχειριστή.
3. Η εφαρμογή ελέγχει εάν βρίσκεται κάποιος αισθητήρας (beacon) εντός εμβέλειας, σε διαφορετική περίπτωση η σάρωση BLE συσκευών επαναλαμβάνεται.
4. Το αναγνωριστικό του αισθητήρα που εντοπίστηκε πιο κοντά αποστέλλεται στο νέφος.
5. Εφόσον ο αισθητήρας εξακριβωθεί πως ανήκει σε κάποιο χώρο της κτηριακής υποδομής, καταγράφεται ο χώρος αυτός ως η τρέχουσα τοποθεσία του χρήστη.
6. Εν συνεχεία ελέγχεται αν η τοποθεσία του χρήστη ανήκει στην διαδρομή που οδηγεί στον επιλεγμένο προορισμό.
7. Εάν πράγματι αποτελεί μέρος της, αποστέλλεται στο τηλέφωνο η οδηγία κατεύθυνσης προς την επόμενη τοποθεσία της διαδρομής, ενώ επιλαμβάνεται η παραπάνω διαδικασία (στάδια 3-7).
8. Διαφορετικά εξετάζεται εάν η τοποθεσία που βρίσκεται ο επισκέπτης συνορεύει με άλλες που ανήκουν στην διαδρομή.
9. Εάν βρεθούν έγκυρες γειτονικές τοποθεσίες αποστέλλονται οδηγίες πλοήγησης προς την κοντινότερη που οδηγεί στον τελικό προορισμό.
10. Διαφορετικά θεωρούμε ότι ο χρήστης έχει χαθεί, οπότε και αποστέλλονται ειδοποιήσεις συστήματος τόσο σε αυτόν όσο και στον διαχειριστή.

Τέλος παραθέτουμε το διάγραμμα δραστηριοτήτων για τον διαχειριστή του συστήματος, περιγράφοντας τα πιο συνηθισμένα σενάρια διαχείρισης. Η γραφική διεπαφή της εφαρμογής ιστού, όπως θα δούμε και στο Κεφάλαιο 4, παρέχει επιλογές για διαχείριση χρηστών, τοποθεσιών και μηνυμάτων. Στο διάγραμμα της Εικόνας 22 περιγράφονται οι πιο συνηθισμένες ενέργειες που αφορούν τη διαχείριση και παρακολούθηση της δραστηριότητας χρηστών και τοποθεσιών:

Προσθήκη / Επεξεργασία / Διαγραφή Χρήστη

1. Ο διαχειριστής μεταβαίνει στην καρτέλα «Χρήστες».
2. Καταχωρεί τα στοιχεία του προς επεξεργασία χρήστη (όνομα, email, επιτρεπόμενες τοποθεσίες) στο αντίστοιχο παράθυρο διαλόγου.

3. Επιβεβαιώνει την αντίστοιχη ενέργεια για Προσθήκη ή Επεξεργασία και τα νέα στοιχεία χρήστη καταχωρούνται στο σύστημα.
4. Εάν είχε επιλεγεί η ενέργεια για διαγραφή χρήστη τότε ο επιλεγμένος χρήστης διαγράφεται από το σύστημα.

Παρακολούθηση Ιστορικού Πλοήγησης Χρήστη

1. Ο διαχειριστής επιλέγει το όνομα του χρήστη μέσα από τη λίστα χρηστών.
2. Ανακτάται το ιστορικό για τον επιλεγμένο χρήστη, από τη βάση δεδομένων.
3. Ανοίγει ένα νέο παράθυρο διαλόγου όπου και εμφανίζονται οι πληροφορίες που ζητήθηκαν.

Αποστολή Ειδοποίησης σε Χρήστη

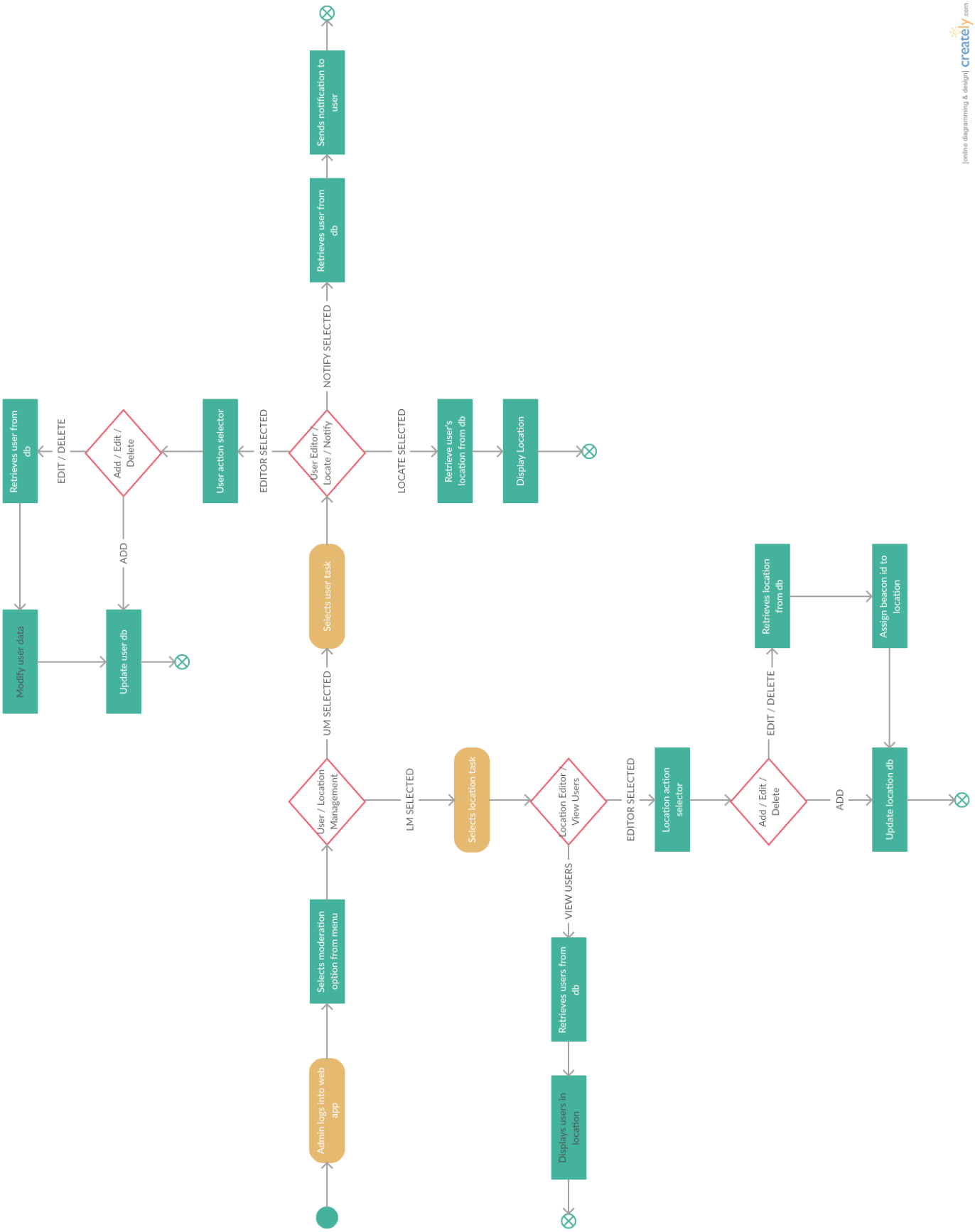
1. Ο διαχειριστής επιλέγει το όνομα του χρήστη μέσα από τη λίστα χρηστών.
2. Στις διαθέσιμες ενέργειες χρήστη, επιλέγει «Αποστολή Μηνύματος».
3. Ανοίγει ένα μια φόρμα σύνταξης μηνύματος.
4. Το μήνυμα αποστέλλεται στο χρήστη.

Προσθήκη / Επεξεργασία / Διαγραφή Τοποθεσίας

1. Ο διαχειριστής μεταβαίνει στην καρτέλα «Τοποθεσίες».
2. Καταχωρεί τα στοιχεία της προς επεξεργασία τοποθεσίας (όνομα, αναγνωριστικό αισθητήρα, τοποθεσίες που συνορεύουν) στο αντίστοιχο παράθυρο διαλόγου.
3. Επιβεβαιώνει την αντίστοιχη ενέργεια για Προσθήκη ή Επεξεργασία και τα νέα στοιχεία τοποθεσίας καταχωρούνται στο σύστημα.
4. Εάν είχε επιλεγεί η ενέργεια για διαγραφή τοποθεσίας τότε η επιλεγμένη τοποθεσία διαγράφεται από το σύστημα.

Παρακολούθηση Δραστηριότητας Χρηστών

1. Ο διαχειριστής επιλέγει το όνομα της τοποθεσίας μέσα από τη λίστα τοποθεσιών.
2. Ανακτάται το ιστορικό πρόσβασης χρηστών για την επιλεγμένη τοποθεσία, από τη βάση δεδομένων.
3. Ανοίγει ένα νέο παράθυρο διαλόγου όπου και εμφανίζονται οι πληροφορίες που ζητήθηκαν.



Εικόνα 22: Σενάριο χρήσης της εφαρμογής για λειτουργίες διαχείρισης

Κεφάλαιο 4 – Υλοποίηση Συστήματος

4.1 Σύστημα Διεπαφής Τελικού Χρήστη

Με τον όρο «Σύστημα Διεπαφής Τελικού Χρήστη (Front-End)» αναφερόμαστε στην γραφική διεπαφή στην οποία έχει πρόσβαση ο χρήστης από τη συσκευή του μαζί με όλο τον κώδικα που τη συνοδεύει και τρέχει τοπικά σε αυτή. Υπάρχουν δύο διαφορετικά Front-Ends στην υλοποίησή μας· ένα για τον επισκέπτη χρήστη που τρέχει στην κινητή του συσκευή και έναν για τον διαχειριστή χρήστη του συστήματος, το οποίο είναι προσβάσιμο στον ιστό από οποιαδήποτε συσκευή. Οι λειτουργίες που επιτελεί κάθε μία από τις παραπάνω διεπαφές χρήστη είναι διαφορετικές και υπόκεινται στις ανάγκες του καθενός ανάλογα με το ρόλο του στην εφαρμογή. Στο προηγούμενο κεφάλαιο, μέσα από τα διαγράμματα Περιπτώσεων Χρήσης και Δραστηριοτήτων αναλύθηκαν εκτενώς οι ενέργειες που επιτελεί κάθε ρόλος στο σύστημά μας. Συνοπτικά το Front-End του επισκέπτη περιορίζεται στην προβολή οδηγιών πλοήγησης, την επιλογή προορισμού και την λήψη ειδοποιήσεων από το διαχειριστή. Από την άλλη μεριά το Front-End του διαχειριστή εστιάζει περισσότερο σε λειτουργίες που αφορούν τον έλεγχο και τη διαχείριση των επισκεπτών και των χώρων του κτηρίου. Για την υλοποίηση των γραφικών διεπαφών χρήσης χρησιμοποιήθηκαν γνωστές τεχνολογίες ανάπτυξης τεχνολογιών ιστού:

HTML5

Η HTML5 είναι μια γλώσσα σήμανσης που χρησιμοποιείται για τη δομή και την παρουσίαση περιεχομένου στον Παγκόσμιο Ιστό. Πρόκειται για την πέμπτη και την τρέχουσα έκδοση του προτύπου HTML. Δημοσιεύθηκε τον Οκτώβριο του 2014 από την Κοινοπραξία World Wide Web (W3C) για τη βελτίωση της γλώσσας με την υποστήριξη των πιο πρόσφατων πολυμέσων, διατηρώντας την ταυτόχρονα κατανοητή από ανθρώπους αλλά και συσκευές, φυλλομετρητές και αναλυτές (parsers), ενώ διατηρεί πλήρη συμβατότητα με τις προηγούμενες εκδόσεις της HTML και XHTML. Η διαφορά της HTML5 από τις προηγούμενες εκδόσεις είναι ότι δίνει ιδιαίτερη έμφαση στο περιεχόμενο πολυμέσων υποστηρίζοντας τις νεότερες κωδικοποιήσεις βίντεο, εικόνας και ήχου, προσφέρει μεγαλύτερη συμβατότητα με εξωτερικές βιβλιοθήκες τρισδιάστατων γραφικών (WebGL), αφού επιτρέπει αλληλεπικαλυπτόμενους καμβάδες στην ίδια σελίδα κλπ. Η HTML5 διαθέτει επίσης ένα δικό της χώρο μόνιμης αποθήκευσης στην ίδια τη συσκευή (local storage), ενώ υποστηρίζει και επεκτάσεις για τοπικές βάσεις δεδομένων. Το τελευταίο μας διευκολύνει στην υλοποίησή μας ώστε να αποθηκεύουμε τα στοιχεία πρόσβασης του χρήστη κατά την είσοδο του για πρώτη φορά στην εφαρμογή.

CSS

Η CSS (Cascading Style Sheet) χρησιμοποιείται επιπρόσθετα με HTML, με στόχο την παροχή επιπλέον επιλογών σχετικά με τη μορφοποίηση του κειμένου και άλλες επιλογές που

αφορούν την τελική εμφάνιση μίας σελίδας που έχει δημιουργηθεί με HTML. Στην υλοποίησή μας χρησιμοποιήθηκε για να καθορίσουμε το μέγεθος των γραμματοσειρών της γραφικής διεπαφής, τα χρώματα, τα πλαίσια διαλόγου, την στοίχιση των επιλογών στα μενού κλπ.

JavaScript

Η JavaScript, συχνά συντομευμένη ως "JS", είναι μια γλώσσα προγραμματισμού υψηλού επιπέδου, δυναμική, μεταβλητών τύπων δεδομένων και αντικειμενοστραφής. Παράλληλα με την HTML και τη CSS, η JavaScript είναι μία από τις τρεις βασικές τεχνολογίες της παραγωγής περιεχομένου στον Παγκόσμιο Ιστό. Χρησιμοποιείται για να κάνει τις ιστοσελίδες διαδραστικές και να παρέχει ολοκληρωμένες εφαρμογές ιστπύ, συμπεριλαμβανομένων των βιντεοπαιχνιδιών. Η πλειοψηφία των ιστότοπων την χρησιμοποιούν και όλα τα σύγχρονα προγράμματα περιήγησης ιστού την υποστηρίζουν χωρίς την ανάγκη για plug-ins, μέσω ενσωματωμένης μηχανής JavaScript. Ενώ αρχικά εφαρμοζόταν μόνο ως "client-side" σε προγράμματα περιήγησης ιστού, οι μηχανές JavaScript ενσωματώνονται πλέον σε πολλούς άλλους τύπους λογισμικού, συμπεριλαμβανομένης της χρήσης τους ως "server-side" (γνωστή και ως Node.js[33]) σε διακομιστές και βάσεις δεδομένων, αλλά και σε προγράμματα εκτός του διαδικτύου, όπως επεξεργαστές κειμένου και αναγνώστες PDF. Στην εφαρμογή μας, η JavaScript αποτέλεσε τον πυρήνα της υλοποίησης των Front-Ends, αφού μέσω αυτής επιτεύχθηκε η λειτουργικότητα όλων των στοιχείων της γραφικής διεπαφής. Οποιαδήποτε επιλογή μενού, παράθυρο διαλόγου ή εικονίδιο «τρέχει» από πίσω κώδικα JavaScript που καθορίζει πως θα αλληλεπιδράσει με τον χρήστη. Πέραν όμως των σχεδιαστικών επιλογών, η JavaScript παρέχει στην εφαρμογή μας και πιο σύνθετη λειτουργικότητα για την διαχείριση των δεδομένων από τη συσκευή, συμπεριλαμβανομένης της κωδικοποίησης και αποκωδικοποίησής τους σε μορφή JSON, αλλά της ανταλλαγής αυτών προς άλλες υπηρεσίες ιστού μέσω κλήσεων AJAX.

jQuery Mobile

Το jQuery Mobile είναι μια βιβλιοθήκη της JavaScript, βελτιστοποιημένο για οθόνες αφής. Επικεντρώνεται στη δημιουργία ενός ενιαίου πλαισίου ανάπτυξης εφαρμογών, συμβατού με μια μεγάλη ποικιλία από έξυπνες κινητές συσκευές, υπολογιστές και ταμπλέτες, με σκοπό τη διαλειτουργικότητα των συσκευών αυτών. Το jQuery Mobile είναι συμβατό με άλλα πλαίσια ανάπτυξης εφαρμογών για κινητά και πλατφόρμες όπως PhoneGap, Worklight και πολλά άλλα. Στην εφαρμογή μας χρησιμοποιήθηκε παράλληλα με την HTML5 για την δημιουργία γραφικών διεπαφών, αυτόματα προσαρμόσιμων στις οθόνες οποιασδήποτε κινητής συσκευής ανεξάρτητα από το μέγεθος ή την ανάλυση αυτών. Επίσης το jQuery, και κατ' επέκταση το jQuery Mobile, παρέχει έτοιμες μεθόδους που διευκολύνουν τον χειρισμό συμβάντων από την εφαρμογή (π.χ. τι ενέργεια θα γίνει όταν φορτωθεί η σελίδα ή όταν αλλάξει η κατάσταση ενός αντικειμένου δηλ. on click event, on change state, on page load κλπ.). Ο εντοπισμός και διαχείριση τέτοιων συμβάντων είναι και μία από τις αδυναμίες της JavaScript, καθώς απαιτείται

[33] https://nodejs.org/en/

η συγγραφή μεγάλων τμημάτων κώδικα για τον αποτελεσματικό χειρισμό τους, γεγονός που καθιστά προτιμότερη τη χρήση εξωτερικών επεκτάσεων, όπως jQuery.

AJAX

Το AJAX ("asynchronous JavaScript and XML") είναι μια τεχνολογία που επιτρέπει τη δημιουργία ασύγχρονων εφαρμογών ιστού. Με το AJAX, οι εφαρμογές ιστού μπορούν να αποστέλλουν δεδομένα και να ανακτούν από έναν διακομιστή ασύγχρονα (στο παρασκήνιο) χωρίς να παρεμβαίνουν στην εμφάνιση και τη συμπεριφορά της υπάρχουσας σελίδας. Με την αποσύνδεση του στρώματος ανταλλαγής δεδομένων από το στρώμα παρουσίασης, το AJAX επιτρέπει σε ιστοσελίδες και κατ ' επέκταση σε δικτυακές εφαρμογές να αλλάζουν δυναμικά το περιεχόμενο χωρίς να χρειάζεται να επαναφορτωθεί ολόκληρη η σελίδα. Το JavaScript και το αντικείμενο XMLHttpRequest[34] παρέχουν μια μέθοδο ασύγχρονης ανταλλαγής δεδομένων μεταξύ του προγράμματος περιήγησης και του διακομιστή, για να αποφευχθεί η πλήρης επαναφόρτωση σελίδων. Το AJAX χρησιμοποιεί τις μεθόδους του πρωτοκόλλου HTTP (GET, PUT, POST, DELETE) για την ανταλλαγή δεδομένων με το διακομιστή, ενώ το σώμα των αιτημάτων περιέχει δεδομένα κωδικοποιημένα συνήθως στις μορφές JSON ή XML. Η JavaScript και, κατ' επέκταση η jQuery διαθέτουν μεθόδους για τον πλήρη χειρισμό ασύγχρονων κλήσεων AJAX. Ο χειρισμός αυτός μπορεί να αφορά στο τί θα γίνει από τι στιγμή που στάλθηκε το αίτημα στον εξυπηρετητή μέχρι να φορτωθούν τα δεδομένα στην εφαρμογή, για πόση ώρα θα περιμένουμε την απάντηση στο αίτημά μας (request timeout), τί είδους περιμένουμε να είναι τα δεδομένα που θα λάβουμε, πως θα χειριστούμε την απάντησή που θα λάβουμε, πως θα χειριστούμε αν υπάρξει απάντηση από τον διακομιστή με κωδικό σφάλματος κλπ.

Για να γίνουν πιο κατανοητά τα παραπάνω ας πάρουμε για παράδειγμα την ακόλουθη ασύγχρονη κλήση GET, στην πιο απλή της μορφή:

```
$.get( url, [data], [callback], [type] )
```

Οι παράμετροι που περιγράφονται στον ορισμό της μεθόδου είναι οι εξής –

- **url** – Η διεύθυνση του διακομιστή στον ιστό, στον οποίο στέλνουμε το αίτημα
- **data** – Εδώ έχουμε το σώμα δεδομένων του αιτήματος μας. Στην περίπτωση της GET η παράμετρος αυτή είναι προαιρετική, ωστόσο στις μεθόδους PUT, POST, DELETE περιέχονται τα δεδομένα προς αποστολή, τα οποία συνήθως κωδικοποιούνται πρώτα σε XML ή JSON.
- **callback** – Η παράμετρος αυτή είναι προαιρετική, ωστόσο είναι ιδιαίτερος ο ρόλος της, καθώς μας επιτρέπει να καθορίσουμε ποια μέθοδος της εφαρμογής μας θα εκτελεστεί αμέσως μόλις λάβουμε την απάντηση από τον διακομιστή.

[34] https://developer.mozilla.org/en-US/docs/Web/API/XMLHttpRequest

- **type** – Επίσης προαιρετική παράμετρος η οποία καθορίζει τον τύπο των δεδομένων που θα επιστραφεί στην callback μέθοδο: "xml", "html", "script", "json", "jsonp", or "text".

4.1.1 Υλοποίηση Εφαρμογής Πλοήγησης

Η εφαρμογή σχεδιάστηκε κάνοντας χρήση των τεχνολογιών HTML5, CSS, JavaScript και jQuery Mobile, ενώ για την μετατροπή της σε Android μορφή χρησιμοποιήσαμε το εργαλείο Apache Cordova. Το εργαλείο αυτό μπορεί να μετατρέψει εφαρμογές που σχεδιάστηκαν με τις παραπάνω τεχνολογίες ιστού, σε εκτελέσιμα αρχεία για iOS και Android. (βλ. § 4.1). Επίσης διαθέτει πληθώρα επεκτάσεων (plugins), οι οποίες μας επιτρέπουν να επικοινωνούμε άμεσα με το λειτουργικό σύστημα και μας παρέχουν πρόσβαση στα διάφορα μέρη της συσκευής (π.χ. κάμερα, πομποδέκτης Bluetooth κλπ.), μέσω χρήσης της JavaScript. Έτσι απαλλασσόμαστε από τις δυσκολίες που θα είχε η ανάπτυξη μιας αμιγώς Android εφαρμογής. Συνοπτικά οι τεχνολογίες που χρησιμοποιήθηκαν για τις ανάγκες της εφαμογής για κινητές συσκευές είναι οι ακόλουθες:

HTML5 Local Storage: Με την τοπική αποθήκευση, οι εφαρμογές ιστού μπορούν να αποθηκεύουν δεδομένα τοπικά στο πρόγραμμα περιήγησης του χρήστη. Πριν από την HTML5, τα δεδομένα εφαρμογών έπρεπε να αποθηκεύονται σε cookies, που περιλαμβάνονται σε κάθε αίτημα διακομιστή. Η τοπική αποθήκευση είναι πιο ασφαλής και μεγάλα ποσά δεδομένων μπορούν να αποθηκευτούν τοπικά. Σε αντίθεση με τα cookies, το όριο αποθήκευσης είναι πολύ μεγαλύτερο (τουλάχιστον 5MB) και οι πληροφορίες δεν μεταφέρονται ποτέ στο διακομιστή. Στην εφαρμογή μας τα στοιχεία που αποθηκεύονται είναι το αναγνωριστικό χρήστη και ο κωδικός πρόσβασής του, κατά την πρώτη είσοδο του σε αυτή. Έτσι, εφόσον τα στοιχεία αυτά εξακολουθούν να ισχύουν ο χρήστης δεν χρειάζεται να τα εισάγει ξανά αλλά συνδέεται αμέσως. Φυσικά τα στοιχεία αυτά εξακολουθούν και αποστέλλονται στο Back-End παρασκηνιακά κάθε φορά που ο χρήστης ανοίγει την εφαρμογή, ώστε να επαληθεύονται εκ νέου σε κάθε είσοδο.

BLE Scanner της Estimote: Πρόκειται για ένα plugin του Apache Cordova που επιτρέπει την σάρωση BLE συσκευών. Το API είναι γραμμένο σε jQuery και λόγω του ότι είναι κατασκευασμένο ειδικά για χρήση με τους αισθητήρες της εταιρίας Estimote, προσφέρει κάποιες επιπλέον λειτουργίες για την επικοινωνία ανάμεσα σε αυτούς και την κινητή συσκευή. Συγκεκριμένα το API, μπορεί και έχει πλήρη έλεγχο στον οδηγό Bluetooth της συσκευής μας, επομένως μπορεί να ορίσει αυτόματα τις βέλτιστες παραμέτρους σύνδεσης με BLE συσκευές (χρόνος ενεργοποίησης και απενεργοποίησης του πομποδέκτη, χρόνος σάρωσης και μέγιστη διάρκεια αναμονής για διακοπή της σύνδεσης (timeout), διάστημα μεσολάβησης ανάμεσα σε διαδοχικές απόπειρες σάρωσης κλπ.). Όταν οι συσκευές σαρωθούν, τότε ταξινομούνται αυτόματα από το API βάσει αύξουσας απόστασης από την συσκευή μας. Δημιουργείται λοιπόν ένας πίνακας από αντικείμενα JavaScript, όπου κάθε αντικείμενο

αποτελεί ουσιαστικά τον κάθε αισθητήρα που σαρώθηκε. Κάθε χαρακτηριστικό (attribute) των αντικειμένων αυτών αφορά και την αντίστοιχη μέτρηση ή πληροφορία του κάθε αισθητήρα. Στα χαρακτηριστικά αυτά περιλαμβάνονται (φυσική διεύθυνση, μέτρηση θερμοκρασίας, μέτρηση φωτεινότητας, μέτρηση απόστασης, επίπεδο μπαταρίας, έκδοση firmware του αισθητήρα κλπ.). Να σημειωθεί εδώ ότι το API είναι γενικό για διάφορα μοντέλα αισθητήρων της Estimote. Εάν κάποιος αισθητήρας δεν υποστηρίζει όλες τις παραπάνω μετρήσεις ή δεν είναι του ίδιου κατασκευαστή, πιθανόν να μην υπάρχουν τιμές σε όλα τα παραπάνω πεδία. Στην εφαρμογή μας φυσικά τα χαρακτηριστικά που μας ενδιαφέρουν είναι η φυσική διεύθυνση (MAC) και η απόσταση κάθε αισθητήρα από την συσκευή. Κάθε φορά που ολοκληρώνεται μία σάρωση το API μας επιτρέπει να καλέσουμε κατευθείαν μία δική μας callback μέθοδο ώστε να ορίσουμε ποια ενέργεια θα γίνει στα αντικείμενα που σαρώθηκαν και αποθηκεύτηκαν στον πίνακα. Στο σημείο αυτό είναι που παίρνουμε τα πεδία που μας ενδιαφέρουν από το πρώτο αντικείμενο στον πίνακα (εγγύτερος αισθητήρας) και τα κωδικοποιούμε σε μορφή JSON, προς αποστολή στο YN μέσω ασύγχρονης κλήσης AJAX. Αμέσως μετά τα αντικείμενα όλα διαγράφονται από τον πίνακα, προκειμένου να πραγματοποιηθεί νέα σάρωση και η διαδικασία επαναλαμβάνεται.

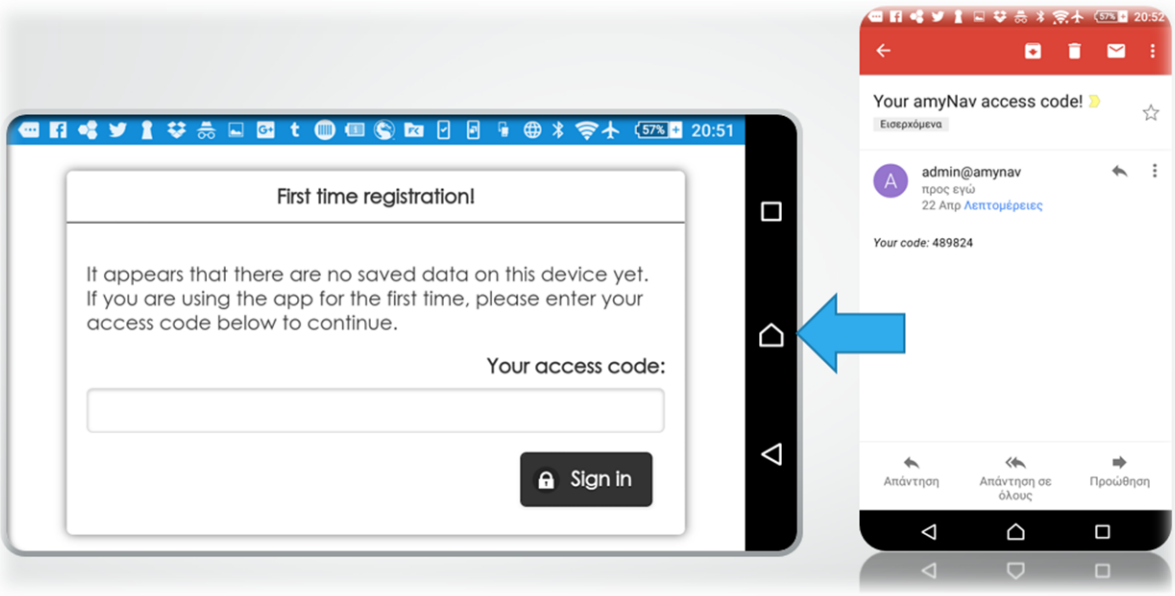
Συμβάντα εφαρμογής: Πρόκειται για ένα Cordova plugin που επιτρέπει στο λειτουργικό σύστημα (Android) να λαμβάνει συμβάντα από μια εφαρμογή, ακόμα και αν αυτή εκτελείται στο παρασκήνιο. Τα συμβάντα αυτά προβάλλονται με τη μορφή αναδυόμενων ειδοποιήσεων (push notifications) του Android και όχι μέσα από το παράθυρο στο οποίο τρέχει η εφαρμογή. Με αυτόν τον τρόπο η επικοινωνία του χρήστη και του διαχειριστή δεν περιορίζεται μονάχα στην ανταλλαγή μηνυμάτων εντός αυτής, αλλά επεκτείνεται και μέσω αυτοματοποιημένων ειδοποιήσεων συστήματος, ικανές να ενημερώνουν τον χρήστη ακόμα και με την οθόνη της συσκευής του κλειστή - μέσω ηχητικών ειδοποιήσεων του Android.

4.1.2 Γραφική Διεπαφή Χρήστη

Η εφαρμογή για Android είναι αρκετά απλή στη χρήση. Σχεδιάστηκε με τρόπο τέτοιο, ώστε ο οποιοσδήποτε κάτοχος ενός έξυπνου τηλεφώνου ή ταμπλέτας, να μπορεί εύκολα να την εγκαταστήσει και να την χρησιμοποιήσει, χωρίς να χρειάζονται ειδικές γνώσεις.

Είσοδος στην εφαρμογή

Αρχικά, όταν η εφαρμογή χρησιμοποιείται για πρώτη φορά, απαιτείται ένας μοναδικός κωδικός πρόσβασης. Ο κωδικός αυτός θεωρούμε πως έχει αποσταλεί εκ των προτέρων στο mail που είχε δηλωθεί στον διαχειριστή. Όπως θα δούμε και σε επόμενη ενότητα, ο κωδικός παράγεται αυτόματα από το σύστημα και αποστέλλεται ύστερα από εντολή του διαχειριστή, με την παραδοχή ότι η διεύθυνση ηλεκτρονικού ταχυδρομείου του παραλήπτη είναι γνωστή και έχει καταχωρηθεί κατά την δημιουργία του προφίλ χρήστη στην εφαρμογή διαχείρισης.



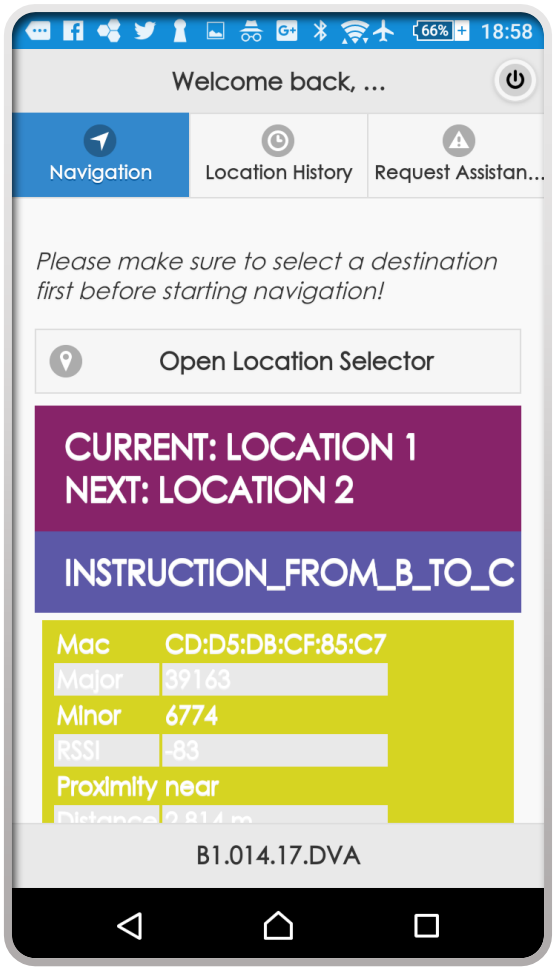
Εικόνα 23: Λήψη κωδικού πρόσβασης με email για σύνδεση στην εφαρμογή πλοήγησης

Οθόνη προβολής οδηγιών

Αποτελεί την κεντρική οθόνη της εφαρμογής, η οποία εμφανίζεται αμέσως μετά την σύνδεση μας σε αυτή. Σκοπό έχει να προβάλει οδηγίες πλοήγησης για το χώρο. Μόλις ο χρήστης συνδεθεί προτρέπεται από την εφαρμογή να ενεργοποιήσει τη σάρωση BLE συσκευών και να επιλέξει προορισμό, κατόπιν οι οδηγίες εμφανίζονται αυτόματα ανάλογα με την ύπαρξη beacons στην εμβέλεια, διαφορετικά εμφανίζονται στη θέση τους κατάλληλα μηνύματα σφάλματος.

Όπως μπορούμε να διακρίνουμε (Εικόνα 24) στην κεντρική οθόνη εμφανίζονται πληροφορίες για την τρέχουσα και την επόμενη τοποθεσία που πρέπει να πάει ο επισκέπτης (μωβ πλαίσιο), ενώ ακριβώς κάτω προβάλλεται η οδηγία που συνδέει τους δύο αυτούς χώρους (μπλε πλαίσιο).

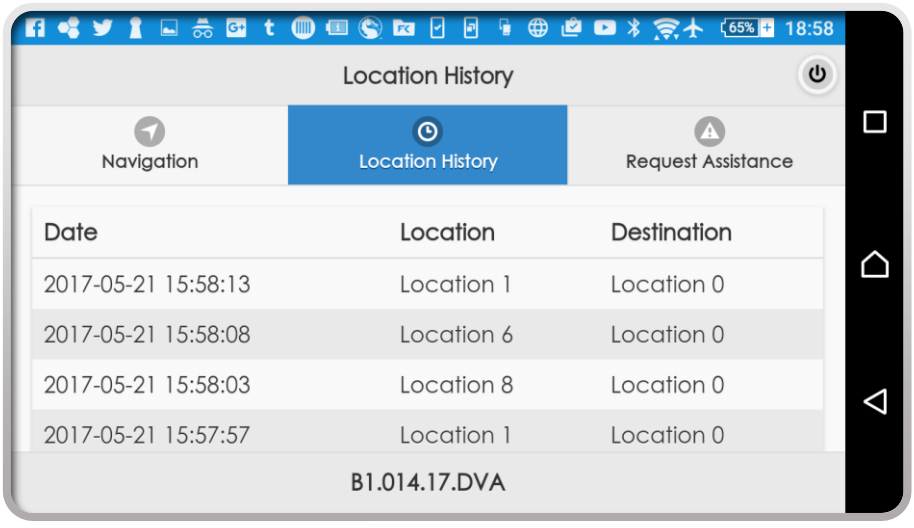
Αξίζει να σημειωθεί ότι κάτω από τις οδηγίες, εμφανίζονται σε μια λίστα (κίτρινο πλαίσιο) και όλες οι BLE συσκευές εντός εμβέλειας καθώς αυτές σαρώνονται από τη συσκευή. Παρόλο αυτή η πληροφορία φυσικά και δεν αφορά τον τελικό χρήστη, στην περίπτωση μας εμφανίζεται κυρίως για λόγους αποσφαλμάτωσης αλλά και επίδειξης της σωστής λειτουργίας του Estimote API που χρησιμοποιήθηκε.



Εικόνα 24: Οθόνη πλοήγησης

Οθόνη ιστορικού πλοήγησης

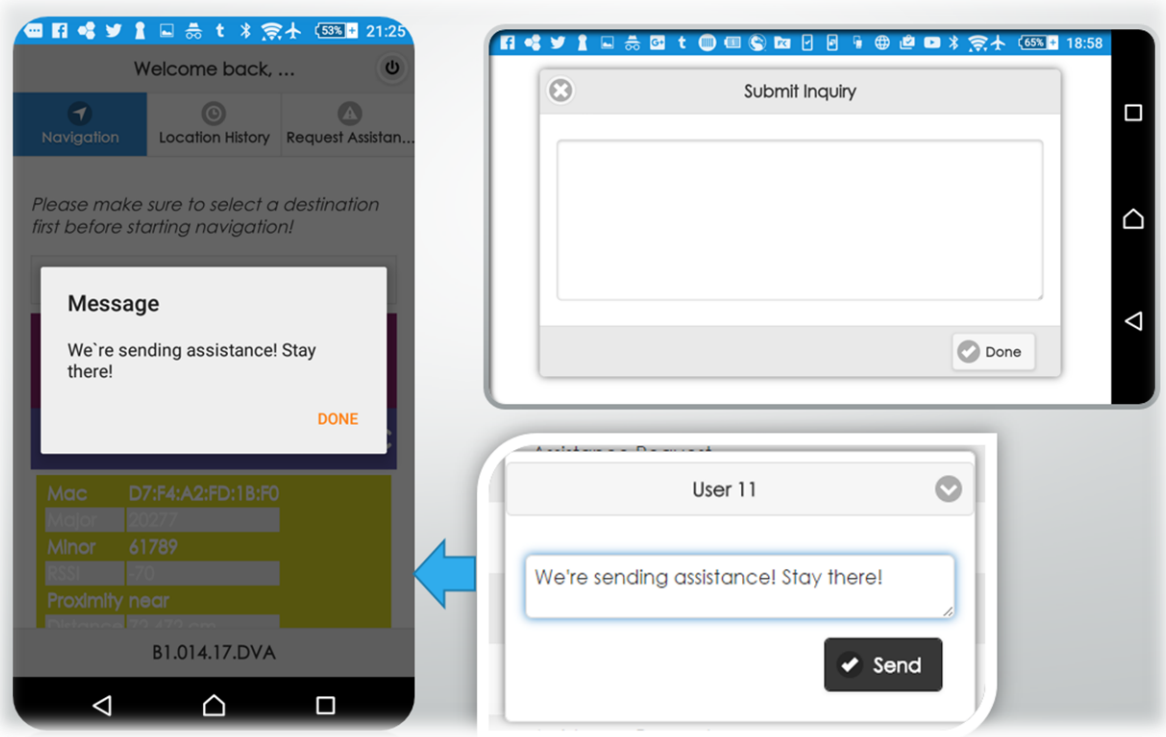
Ο χρήστης μπορεί να δει ένα σύντομο ιστορικό των πρόσφατων τοποθεσιών που έχει επισκεφτεί. Αντίστοιχη λειτουργία για ιστορικό χρηστών, αλλά και τοποθεσιών, υπάρχει και στην web εφαρμογή του διαχειριστή, όπως θα δούμε και παρακάτω.



Εικόνα 25: Προβολή Ιστορικού πρόσφατων τοποθεσιών

Οθόνη επικοινωνίας

Εκτός από τις αυτοματοποιημένες ειδοποιήσεις, ο επισκέπτης μπορεί να επικοινωνήσει και απευθείας με τον διαχειριστή μέσω μηνυμάτων. Για τα μηνύματα που λαμβάνει ο χρήστης στη συσκευή του ενημερώνεται τόσο μέσα από την ίδια την εφαρμογή (Εικόνα 26), όσο και μέσω αναδυόμενων ειδοποιήσεων (push notifications) του Android.



Εικόνα 26: Ανταλλαγή μηνυμάτων μεταξύ χρήστη και διαχειριστή

4.1.3 Γραφική Διεπαφή Διαχειριστή

Η διαδικτυακή εφαρμογή διαχείρισης σχεδιάστηκε χρησιμοποιώντας τις ίδιες τεχνολογίες με αυτή για τα τηλέφωνα που περιγράψαμε προηγουμένως (HTML5, CSS, JavaScript και jQuery Mobile). Η επιλογή της jQuery Mobile[35] δεν έγινε τυχαία ούτε σε αυτή την περίπτωση, παρόλο που η εφαρμογή τρέχει κατά κύριο λόγο μέσα από web browser. Συγκεκριμένα έχουμε τη δυνατότητα μέσω αυτής, να σχεδιάζουμε διαδραστικές γραφικές διεπαφές που προσαρμόζονται αυτόματα σε κάθε οθόνη, ανεξαρτήτως ανάλυσης, μεγέθους ή προγράμματος περιήγησης. Κάτι τέτοιο μας ενδιαφέρει ιδιαίτερα, καθώς θέλουμε η εφαρμογή διαχείρισης να μπορεί να τρέξει σωστά και από κινητά τηλέφωνα ή tablets, ενώ μέσω του

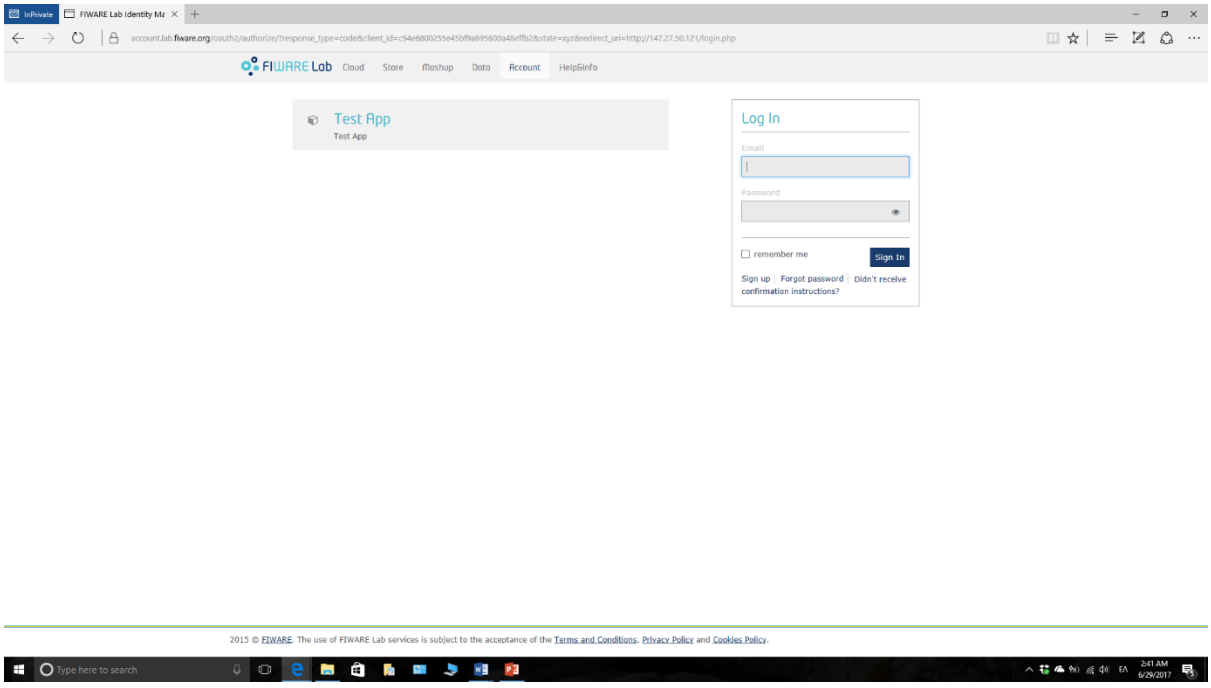
[35] http://demos.jquerymobile.com/1.4.0/rwd/

Apache Cordova είναι ιδιαίτερα εύκολο να μετατραπεί σε stand alone για Android κατ' αντιστοιχία με αυτή που έχει στη διάθεσή του ο χρήστης.

Η γραφική διεπαφή διαχείρισης είναι επίσης απλή στο σχεδιασμό της, αποτελούμενη από τέσσερις (4) καρτέλες στο πάνω μέρος της που αποτελούν ουσιαστικά και το κεντρικό μενού επιλογών: Γρήγορη Ενημέρωση, Χρήστες, Τοποθεσίες και Επικοινωνία.

Είσοδος στην εφαρμογή

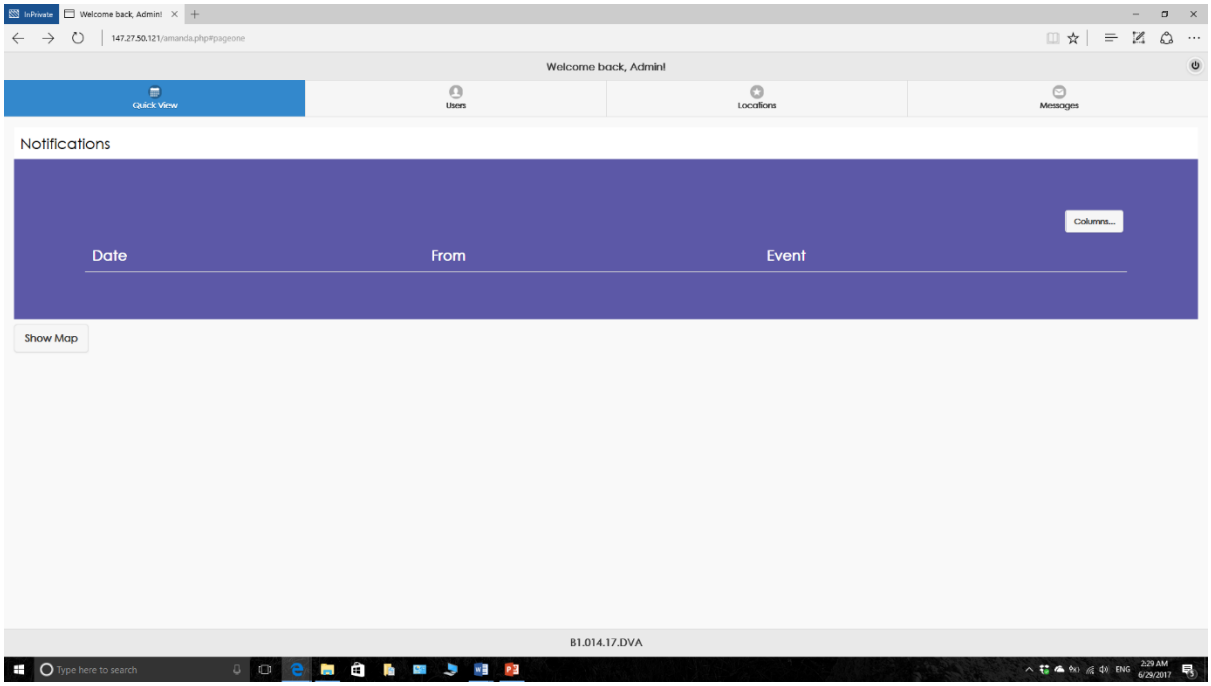
Για το διαχειριστή συστήματος, επιλέξαμε να συνδέεται στην εφαρμογή μέσω του FIWARE λογαριασμού που πρέπει να διαθέτει · συνεπώς κάθε φορά που θέλει να αποκτήσει πρόσβαση στη σελίδα διαχείρισης, ανακατευθύνεται πρώτα για σύνδεση στη σελίδα του KeyRock. Η πλήρης λειτουργικότητα της συγκεκριμένης υπηρεσίας αναλύεται σε επόμενη ενότητα.



Εικόνα 27: Σύνδεση στη σελίδα διαχείρισης μέσω KeyRock

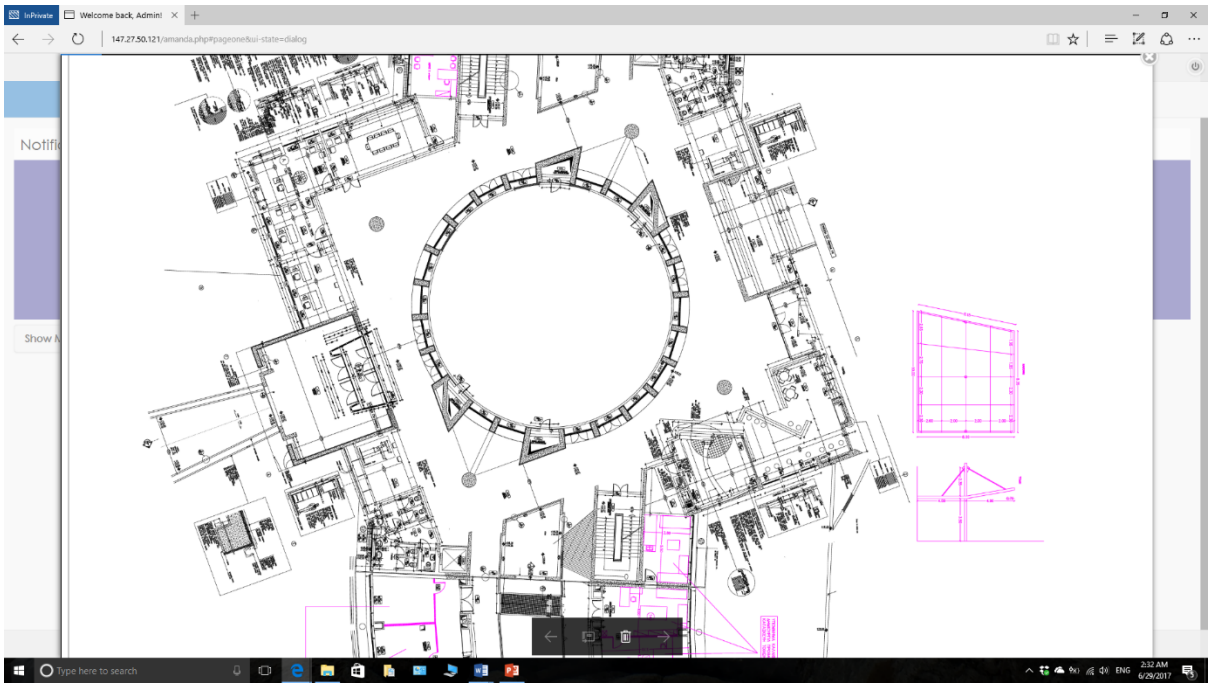
Οθόνη γρήγορης ενημέρωσης

Αποτελεί την αρχική σελίδα της διαδικτυακής εφαρμογής, την οποία και βλέπει ο διαχειριστής όταν συνδέεται. Στόχο έχει να προβάλει νέα συμβάντα και ειδοποιήσεις. (πχ. μη αναγνωσμένα μηνύματα από επισκέπτες, ειδοποιήσεις συστήματος κλπ.).



Εικόνα 28: Οθόνη γρήγορης ενημέρωσης

Από την ίδια οθόνη μπορούμε να δούμε και το χάρτη της κτηριακής υποδομής (επιλέγοντας "Show Map"), για να αποκτήσουμε με αυτό τον τρόπο μια καλύτερη αντίληψη για τους χώρους αυτής, αλλά και τις θέσεις που βρίσκονται τοποθετημένα τα beacons σε όλη την έκτασή της.



Εικόνα 29: Προβολή χάρτη

Οθόνη προβολής και διαχείρισης χρηστών

Προβάλει τους χρήστες που είναι εγγεγραμμένοι στο σύστημα μας, παρέχοντας παράλληλα και επιλογές για τη διαχείρισή τους. Συγκεκριμένα όπως φαίνονται και στην Εικόνα 30 (από αριστερά προς τα δεξιά) οι διαθέσιμες επιλογές είναι οι εξής:

Όνομα: *Εμφανίζει το όνομα του χρήστη.*

Τοποθεσία: *Εμφανίζει την τοποθεσία που βρίσκεται κάθε στιγμή, ή "null" αν ο χρήστης δεν έχει συνδεθεί στην εφαρμογή για το τηλέφωνο. Ανανεώνεται σε πραγματικό χρόνο, καθώς ο χρήστης μετακινείται μέσα στο κτήριο και εντοπίζεται από αισθητήρες. Δίπλα μάλιστα στην τοποθεσία εμφανίζονται και επιπλέον συμβολισμοί ανάλογα με την κατάσταση πλοήγησης κάθε χρήστη (Εικόνα 31).*

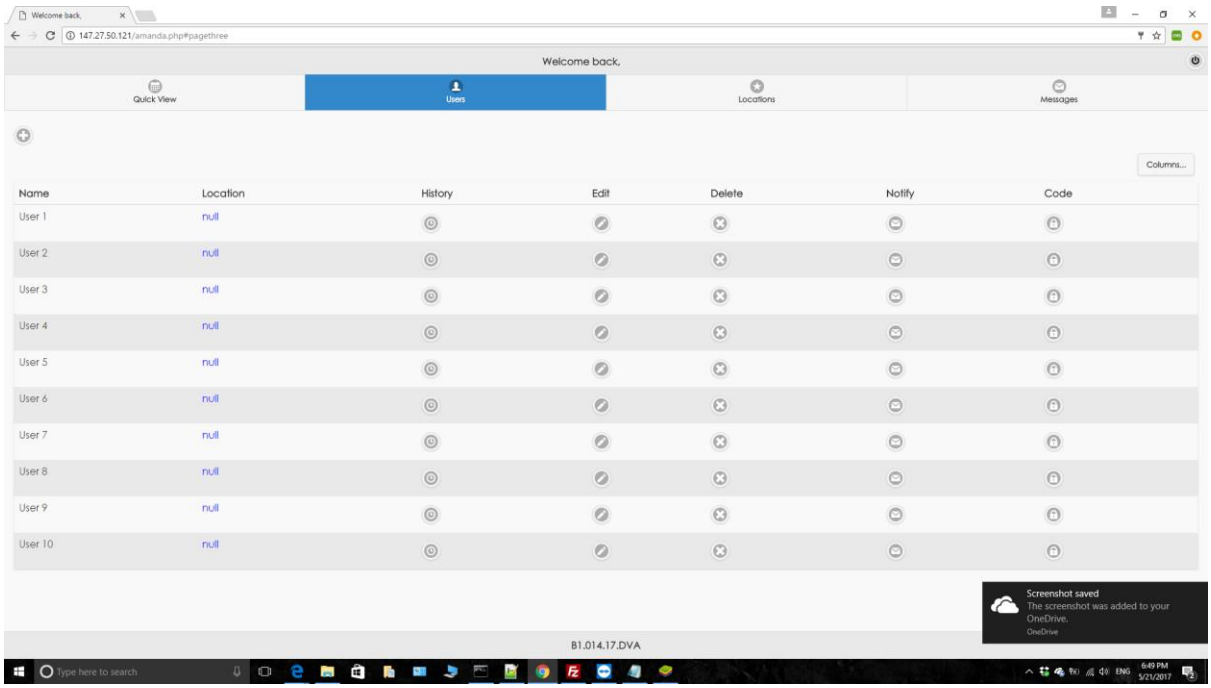
Ιστορικό: *Εμφανίζει το ιστορικό πλοήγησης ενός χρήστης με ανάλογο τρόπο όπως είδαμε και στην 4.1.2.*

Επεξεργασία: *Μας επιτρέπει να τροποποιήσουμε τις υπάρχουσες πληροφορίες για κάθε χρήστη, όπως όνομα, email και δικαιώματα πρόσβασης ανά χώρο (Εικόνα 32).*

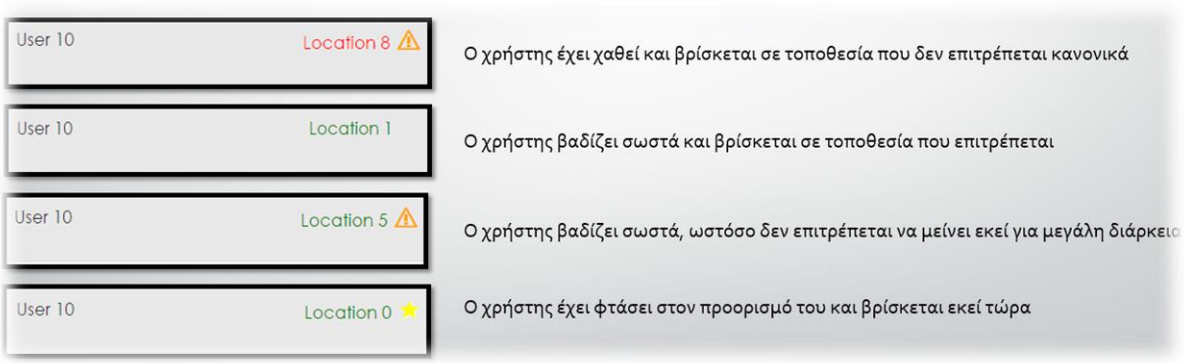
Διαγραφή: *Διαγράφουμε το χρήστη από το σύστημά μας, αποκλείοντας τον ταυτόχρονα και από κάθε χρήση της εφαρμογής για το τηλέφωνό του.*

Ειδοποίηση: *Αποστολή προσωπικού μηνύματος στον επιλεγμένο χρήστη.*

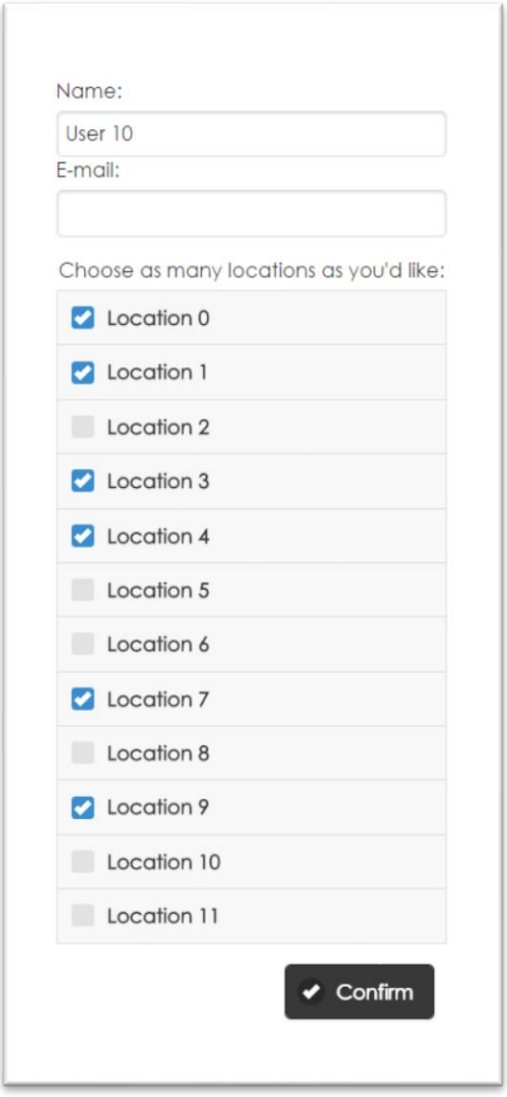
Κωδικός: *Αυτόματη παραγωγή νέου μοναδικού κωδικού εισόδου και αποστολή στη διεύθυνση email του χρήστη. Ο κωδικός αυτός χρησιμοποιείται για είσοδο στην mobile εφαρμογή (βλ. § 4.1.2).*



Εικόνα 30: Οθόνη προβολής χρηστών

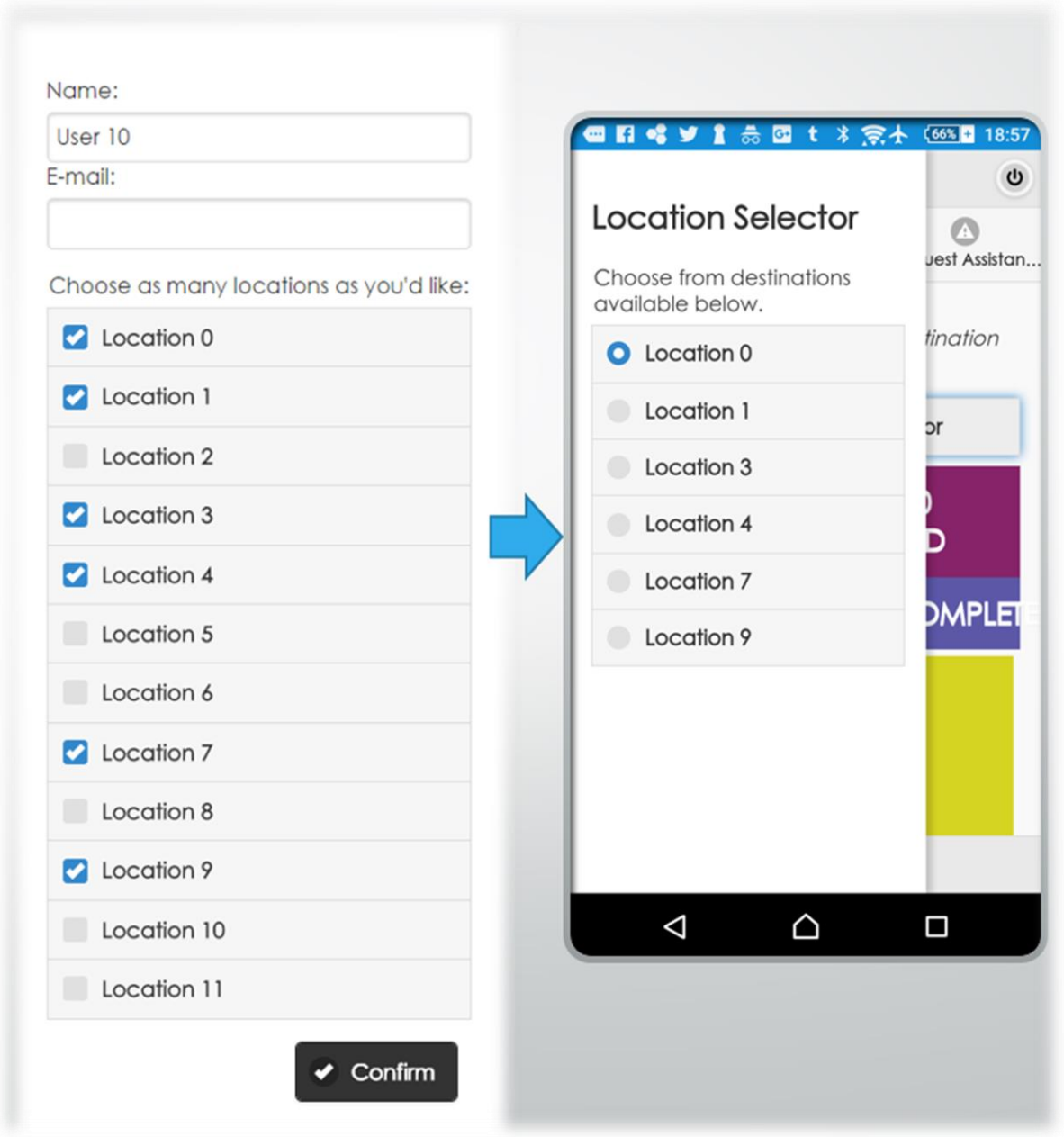


Εικόνα 31: Ειδοποιήσεις συστήματος ελέγχου χρηστών



Εικόνα 32: Πλαίσιο διαλόγου επεξεργασίας στοιχείων χρήστη και δικαιωμάτων πρόσβασης

Αξίζει να προσέξουμε στο σημείο αυτό την αντιστοιχία που υπάρχει ανάμεσα στην οθόνη διαχείρισης των δικαιωμάτων χρήστη ανά τοποθεσία και στο μενού επιλογής προορισμού που βλέπει ο ίδιος ο χρήστης στην εφαρμογή του στο τηλέφωνο. Όπως μπορούμε να διαπιστώσουμε (Εικόνα 33) η λίστα με τους διαθέσιμους προορισμούς, περιλαμβάνει μόνο όσους όρισε ο διαχειριστής. Το μενού αυτό φυσικά διαφοροποιείται κάθε φορά ανάλογα με τα δικαιώματα που έχουν παραχωρηθεί. Έτσι ο διαχειριστής έχει έναν επιπλέον έλεγχο στις τοποθεσίες τις οποίες θέλει να είναι ορατές στους επισκέπτες.



Εικόνα 33: Ορισμός επιτρεπτών τοποθεσιών χρήστη και εμφάνιση αυτών για διαθέσιμες επιλογές προορισμού

Οθόνη προβολής και διαχείρισης χώρων

Προβάλει τους χώρους της κτηριακής υποδομής, παρέχοντας παράλληλα και επιλογές για τη διαχείρισή τους. Συγκεκριμένα όπως φαίνονται και στην Εικόνα 34 (από αριστερά προς τα δεξιά) οι διαθέσιμες επιλογές είναι οι εξής:

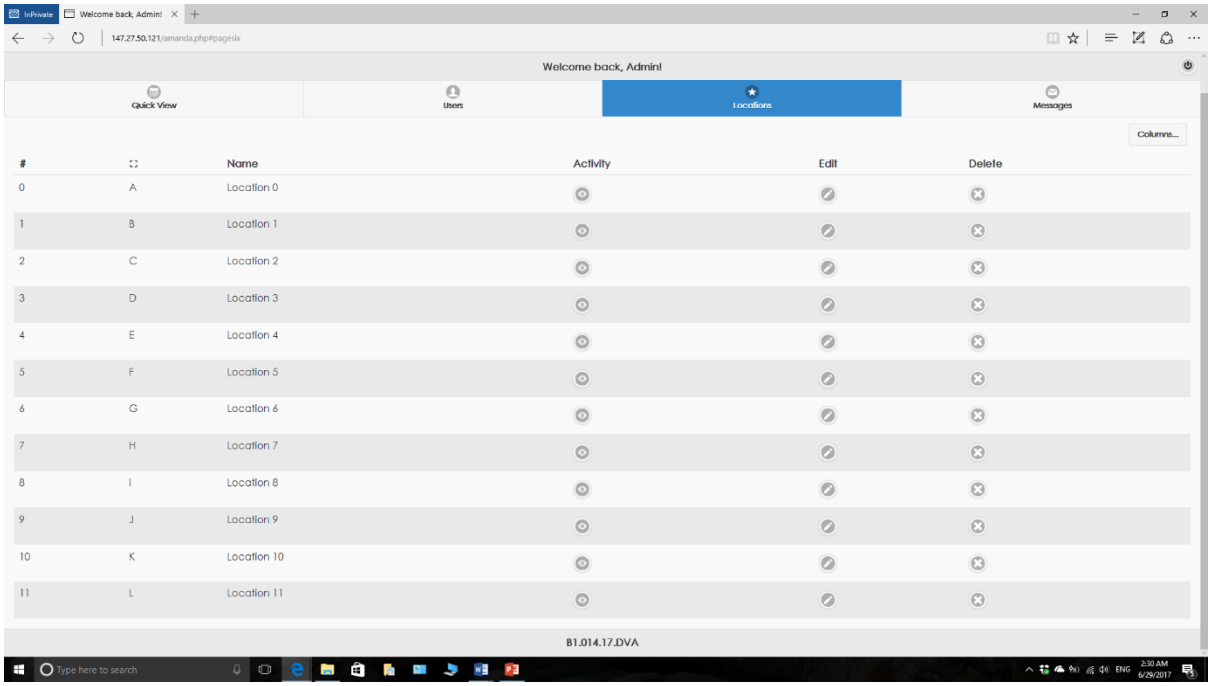
Α/Α: *Αύξων αριθμός τοποθεσίας. Χρησιμοποιείται για την ταυτοποίηση της σε περίπτωση που περισσότερες από μία τοποθεσίες έχουν κοινό όνομα..*

Όνομα: *Εμφανίζει το όνομα του χώρου.*

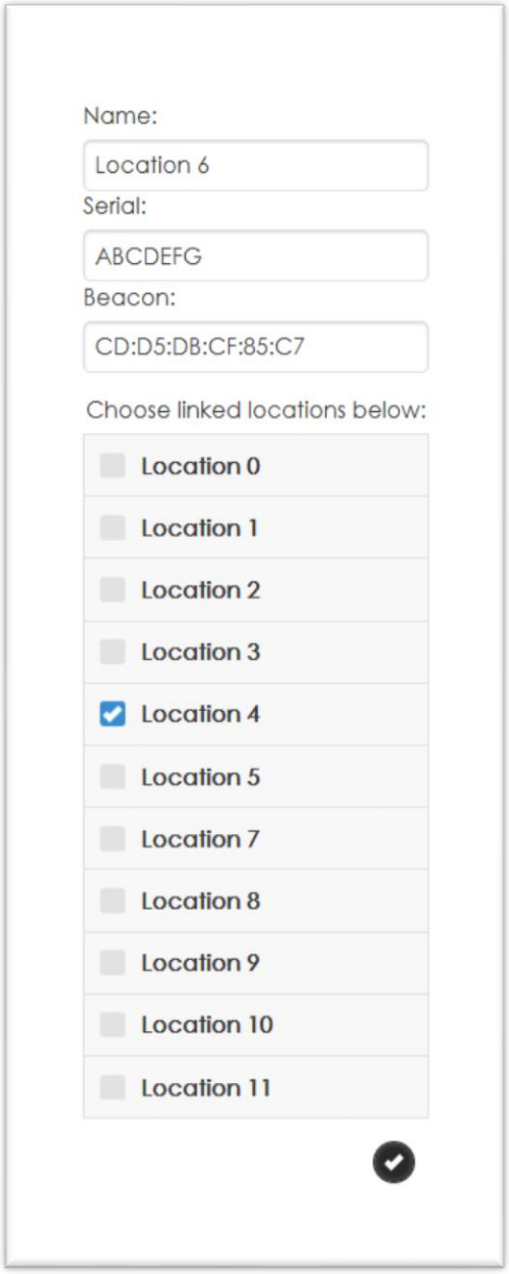
Δραστηριότητα: *Αντίστοιχη λειτουργία με το Ιστορικό πλοήγησης, μόνο πως τώρα εμφανίζονται όλοι οι χρήστες που βρέθηκαν στην συγκεκριμένη τοποθεσία μέσα στο χρονικό διάστημα της τελευταίας μιας ώρας.*

Επεξεργασία: *Μας επιτρέπει να αλλάξουμε το όνομα του χώρου, να αντιστοιχίσουμε ένα beacon, να ορίσουμε μια προκαθορισμένη διαδρομή που οδηγεί σε αυτόν καθώς και τις γειτονικές του τοποθεσίες (Εικόνα 35).*

Διαγραφή: *Διαγραφή χώρου.*



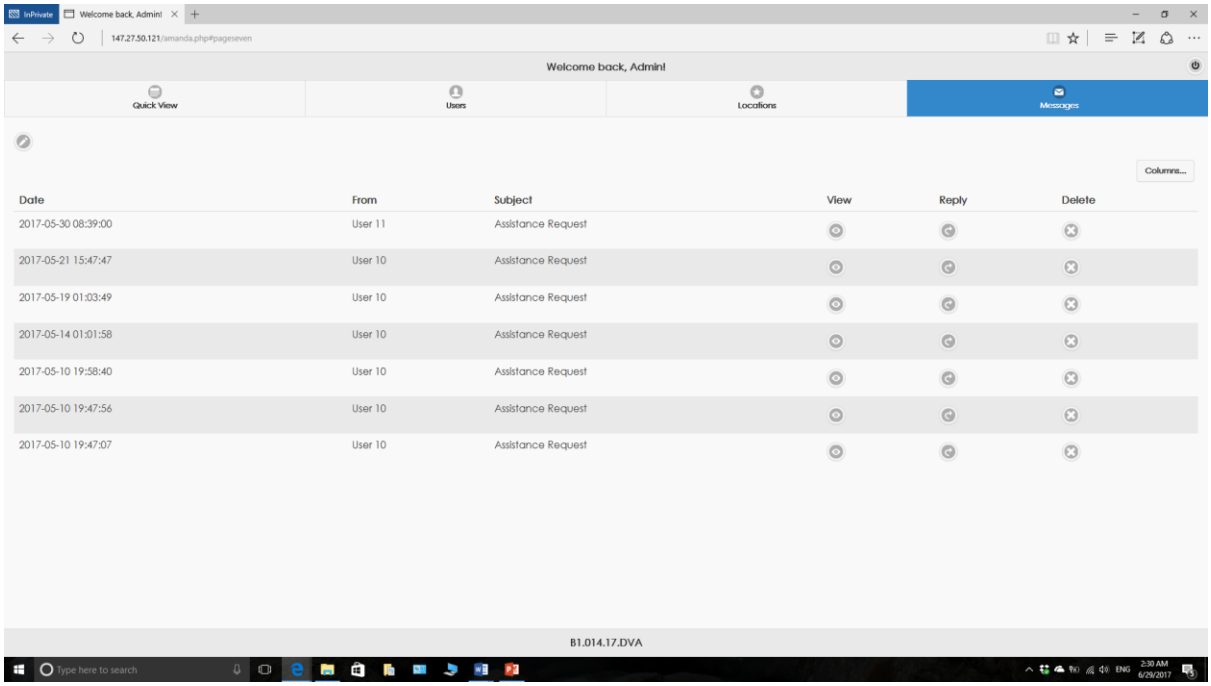
Εικόνα 34: Οθόνη προβολής χώρων



Εικόνα 35: Επεξεργασία στοιχείων χώρου

Οθόνη επικοινωνίας

Παρέχει λειτουργίες αποστολής και λήψης μηνυμάτων από μεταξύ του διαχειριστή και των χρηστών. *(βλ. και § 4.1.2, Εικόνα 26).* Μηνύματα που αποστέλλονται όταν ο χρήστης δεν έχει ανοιχτή την εφαρμογή για το τηλέφωνο, εμφανίζονται σε αυτήν αφότου την ανοίξει. Αντίστοιχα, μη αναγνωσμένες εκκλήσεις που δέχτηκε ο διαχειριστής από χρήστες εμφανίζονται και στην κεντρική οθόνη γρήγορης ενημέρωσης.



Εικόνα 36: Οθόνη επικοινωνίας

4.2 Επεξεργασία Δεδομένων στο Νέφος και Υλοποίηση Υπηρεσιών

Με τον όρο «Υπηρεσίες Νέφους Παρασκηνίου (Back-End)» αναφερόμαστε στο σύνολο των υπηρεσιών που "τρέχουν" στο υπολογιστικό νέφος με στόχο την επεξεργασία και απάντηση των αιτημάτων, προερχόμενα από τις εφαρμογές τελικών χρηστών (“Front-End ”). Για τον προγραμματισμό των υπηρεσιών νέφους έγινε χρήση των παρακάτω τεχνολογιών:

PHP: Η PHP (PHP: Hypertext Preprocessor) είναι μια γλώσσα προγραμματισμού για τη δημιουργία ιστοσελίδων με δυναμικό περιεχόμενο. Μια σελίδα PHP περνά από επεξεργασία από ένα συμβατό διακομιστή του Παγκόσμιου Ιστού (π.χ. Apache), ώστε να παραχθεί σε πραγματικό χρόνο το τελικό περιεχόμενο, που είτε θα σταλεί στο πρόγραμμα περιήγησης των επισκεπτών σε μορφή κώδικα HTML ή θα επεξεργασθεί τις εισόδους δίχως να προβάλλει την έξοδο στο χρήστη, αλλά θα τις μεταβιβάσει σε κάποιο άλλο PHP script. Στην υλοποίησή του Back-End χρησιμοποιήσαμε Apache εξυπηρετητή για την διαχείριση των αιτημάτων · επόμενο λοιπόν ήταν να γίνει χρήση και της PHP, αφού είναι η κύρια γλώσσα προγραμματισμού κατανοητή από τον συγκεκριμένο εξυπηρετητή. Λόγω του ότι όλες σχεδόν οι σελίδες της εφαρμογής μας είναι προκατασκευασμένες, με τα δεδομένα απλά να φορτώνονται ασύγχρονα με AJAX και η γραφική διεπαφή να σχεδιάζεται κατευθείαν στην πλευρά του πελάτη μέσω JavaScript, η PHP δεν χρησιμοποιήθηκε ιδιαίτερα ως εργαλείο για την κατασκευή δυναμικών ιστοσελίδων. Η χρήση που κυριάρχησε ήταν για την διαχείριση των REST αιτημάτων και για την επικοινωνία με την MySQL βάση όπου αποθηκεύονται όλα τα

δεδομένα του Back-End. Για να διευρυνθεί και να απλουστευτεί η παραπάνω λειτουργικότητα που μας ενδιέφερε, χρησιμοποιήθηκαν επιπλέον και οι εξής επεκτάσεις:

Slim: Το Slim είναι ένα micro framework για την PHP που διευκολύνει κατά πολύ τον χειρισμό και την απάντηση REST αιτημάτων. Επιτρέπει την δημιουργία εικονικών URI της μορφής REST, ενώ δρομολογεί και απαντά αυτόματα τα αιτήματα. Ας πάρουμε για παράδειγμα το παρακάτω τμήμα κώδικα που αφορά ένα αίτημα GET:

```
$app->get('/hello/{name}', function (Request $request, Response $response) {
    $name = $request->getAttribute('name');
    $response->getBody()->write("Hello, $name");

    return $response;
});
```

Στην πρώτη γραμμή του κώδικά έχουμε την διεύθυνση URL η οποία θα δέχεται το GET αίτημά μας, στο παράδειγμά μας `/hello/{name}`. Η παραπάνω διεύθυνση μπορεί να είναι οποιαδήποτε επιθυμούμε εμείς και δεν χρειάζεται να αντιστοιχεί σε πραγματικούς καταλόγους του συστήματος αρχείων μας. Για να γίνει η παραπάνω διαδικασία μέσω απλής PHP, θα έπρεπε αρχικά να τροποποιήσουμε χειροκίνητα το αρχείο `.httpaccess` του Apache ώστε να ορίσουμε τις δρομολογήσεις των αιτημάτων. Επιπρόσθετα μέσω κανονικών εκφράσεων θα έπρεπε να ελέγχουμε για την εγκυρότητα των URI και να διαχωρίζουμε τις πραγματικές από τις εικονικές δρομολογήσεις. Στη δεύτερη γραμμή, το Slim διαβάζει αμέσως τις μεταβλητές του GET αιτήματος (`name`) ενώ στην τρίτη γραμμή έχει ήδη ετοιμάσει το σώμα της απάντησης, αποθηκεύοντάς το στη μεταβλητή `$response`. Το παραπάνω θα χρειαζόταν πολύ περισσότερες γραμμές κώδικά αμιγούς PHP για να πραγματοποιηθεί και να λειτουργήσει σωστά.

cURL: Η cURL είναι μια βιβλιοθήκη της PHP που επιτρέπει την μεταφορά δεδομένων μεταξύ των υπηρεσιών χρησιμοποιώντας διάφορα πρωτόκολλα (DICT, FTP, FTPS, HTTP, HTTPS κ.α.). Στην εφαρμογή μας χρησιμοποιήθηκε για την κλήση μεθόδων του πρωτοκόλλου HTTP κατευθείαν μέσα από τον κώδικα της PHP.

MySQL: Η MySQL είναι ένα σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων. Το πρόγραμμα τρέχει έναν εξυπηρετητή (MySQL server) παρέχοντας πρόσβαση πολλών χρηστών σε ένα σύνολο βάσεων δεδομένων. Η MySQL χρησιμοποιήθηκε στην υλοποίησή μας λόγω της εγγενούς συμβατότητάς της με τον Apache και την PHP, αλλά και λόγω της εύκολής διαχείρισης του MySQL εξυπηρετητή μέσα από το γραφικό περιβάλλον phpMyAdmin[36].

[36] https://www.phpmyadmin.net/

4.2.1 Υπηρεσίες Υπολογιστικού Νέφους

Όλες οι υπηρεσίες υλοποιήθηκαν με βάση τη REST αρχιτεκτονική. Ακολουθείται συνήθως το κλασικό μοντέλο επικοινωνίας πελάτη - εξυπηρετητή, όπου λαμβάνεται ένα αίτημα (request), ανακτώνται από τη βάση μας τα δεδομένα που χρειάζονται για να διεκπεραιωθεί και στη συνέχεια αποστέλλεται πίσω μια απάντηση (response), συνοδευόμενη από ένα κωδικό κατάστασης (response code). Πιο περίπλοκη κάπως λογική έχει η υπηρεσία πλοήγησης, όπου και χρειάζεται να ανακτηθούν δεδομένα από διαφορετικούς πίνακες της βάσης μας (χρήστες, τοποθεσίες, δικαιώματα πρόσβασης και οδηγίες), για την χρήση αυτών στον αλγόριθμο πλοήγησης και παρακολούθησης της θέσης των επισκεπτών. Ακολουθεί η περιγραφή όλων των υπηρεσιών (βλ. και διάγραμμα αρχιτεκτονικής § 3.4) του Back-End:

4.2.1.1 Υπηρεσία Ταυτοποίησης και Εξουσιοδότησης Χρηστών KeyRock[37]

Η υπηρεσία παρέχεται από το FIWARE και χρησιμοποιεί το πρωτόκολλο εξουσιοδότησης OAuth2, ο τρόπος λειτουργίας του οποίου είναι ο εξής:

- Ο διαχειριστής του ΥΝ, αρχικά εγγράφει την εφαρμογή του στη λίστα εφαρμογών που θα αιτούν εξουσιοδότηση μέσω του KeyRock. Η εγγραφή νέων εφαρμογών πραγματοποιείται εύκολα μέσα από το γραφικό περιβάλλον που διαθέτει ο πίνακας ελέγχου του KeyRock. Μετά την ολοκλήρωση εγγραφής της νέας εφαρμογής μας ο KeyRock παράγει αυτόματα ένα URI στο οποίο περιέχονται ένα μοναδικό αναγνωριστικό για την συγκεκριμένη εφαρμογή (client_id) καθώς και η διεύθυνση ανακατεύθυνσης στο Back-End μας. Για παράδειγμα στο URI της εφαρμογής μας φαίνεται με ανοικτό γκρι χρώμα το αναγνωριστικό εφαρμογής (client_id) ενώ με πιο σκούρο χρώμα η διεύθυνση ανακατεύθυνσης (redirect_uri).

  ```
  https://account.lab.fiware.org/oauth2/authorize/?res
  ponse_type=code&client_id=c94e6800255e45bf9a695600a4
  8effb2&state=xyz&redirect_uri=http://147.27.50.121/l
  ogin.php/
  ```

- Κατά την είσοδο ενός χρήστη στην εφαρμογή, αυτή τον ανακατευθύνει στον εξυπηρετητή ταυτοποίησης - στην περίπτωση μας στον KeyRock - ενώ μέσω του αναγνωριστικού (client_id) που περιέχεται στο URI, ο εξυπηρετητής γνωρίζει για ποια εφαρμογή πρόκειται.

[37] https://catalogue.fiware.org/enablers/identity-management-keyrock

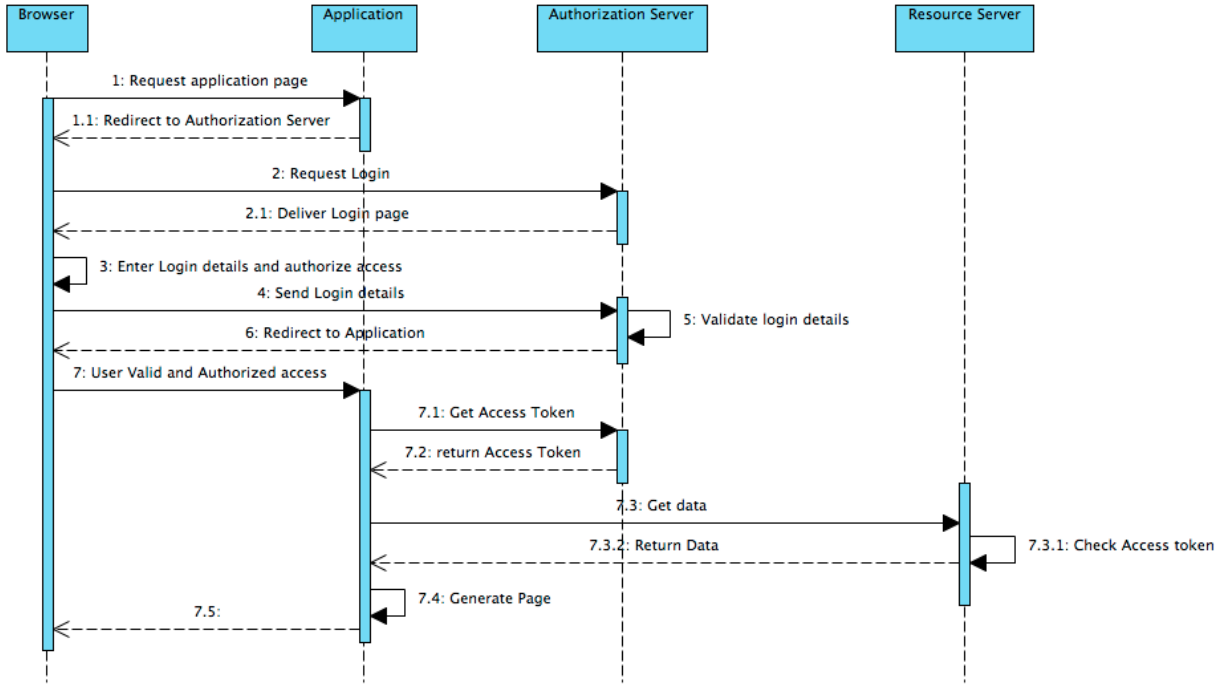
- Ο χρήστης εισάγει τα στοιχεία του FIWARE λογαριασμού του, όπου και εξακριβώνονται από τον KeyRock (authentication).

- Αν είναι η πρώτη φορά που συνδέεται στην εφαρμογή, ο KeyRock ρωτά το χρήστη σε ποια από τα στοιχεία του λογαριασμού του θα επιτρέψει στην εφαρμογή να έχει πρόσβαση (authorization).

- Σε περίπτωση που ο χρήστης δεχτεί να εξουσιοδοτήσει την εφαρμογή, γίνεται επιστροφή σε αυτή, μαζί με ένα κωδικό εξουσιοδότησης (authorization code), ενώ λαμβάνεται και ένα κουπόνι πρόσβασης[38] (access token), μέσω του οποίου η εφαρμογή έχει πρόσβαση στα στοιχεία του χρήστη που παρέχονται από το KeyRock.

Μετά την ολοκλήρωση της παραπάνω διαδικασίας η εφαρμογή έχει πλέον πρόσβαση στα εξής στοιχεία του FIWARE λογαριασμού του χρήστη: όνομα, διεύθυνση ηλεκτρονικού ταχυδρομείου και ρόλο - στην περίπτωσή μας «διαχειριστής».

Κατόπιν μπορούμε να δημιουργούμε μία συνεδρία (session) στον εξυπηρετητή μας, ώστε να θυμόμαστε τα στοιχεία αυτά για όση διάρκεια ορίσουμε, μέχρι να απαιτήσουμε εκ νέου σύνδεση με τον παραπάνω τρόπο.



Εικόνα 37: Διαδικασία σύνδεσης μέσω του πρωτοκόλλου OAuth2

Οι HTTP μέθοδοι της υπηρεσίας περιγράφονται στον παρακάτω πίνακα:

[38] https://auth0.com/docs/tokens/access-token

Πίνακας 1: Μέθοδοι HTTP αιτημάτων KeyRock Identity Manager

Μέθοδος	URL Μεθόδου	Σώμα αιτήματος	Περιγραφή μεθόδου
GET	http://account.lab.fiware.org/oauth2/authorize?response_type=code&client_id=bbe6a9794a47462aa3295bc1cb5b44b9&state=xyz&redirect_uri=http://147.27.60.151/login.php	-	Μετάβαση στη σελίδα του KeyRock για εξακρίβωση χρήστη. Το client_id αποτελεί αναγνωριστικό της εφαρμογής μας. Το redirect_uri ορίζει τη σελίδα που θα μεταφερθεί ο χρήστης μετά τη σύνδεσή του.
GET	https://account.lab.fiware.org/oauth2/token	HEADERS: 'Content-Type: application/x-www-form-urlencoded', 'Authorization: Basic '.base64_encode("client_id:client_secret") URL-encoded: grant_type=authorization_code&code="{code}"&redirect_uri="{redirect_uri}	Αφού ο χρήστης αποδεχτεί να παραχωρήσει τα στοιχεία του στην εφαρμογή, ανακατευθύνεται πίσω σε αυτή. Μέσω αυτού του αιτήματος, η εφαρμογή στέλνει τον κωδικό εξουσιοδότησης, που έλαβε από την εκτέλεση της προηγούμενης ενέργειας και τα στοιχεία της προς τον KeyRock, με στόχο να επιστραφεί το Access Token.
GET	https://account.lab.fiware.org/user?access_token=$access_token	-	Το Access Token είναι ένα αναγνωριστικό που μας επιτρέπει να έχουμε πρόσβαση στα στοιχεία του χρήστη που βρίσκονται στον KeyRock, στέλνοντας αυτό το αίτημα.

4.2.1.2 Υπηρεσία Διαχείρισης Συμβάντων και Συνδρομών Orion[39]

Τα μοντέλα πληροφορίας NGSI-9/NGSI-10[40]

Οι προδιαγραφές διαχείρισης περιβάλλοντος του FI-WARE NGSI βασίζονται στις προδιαγραφές διαχείρισης περιβάλλοντος NGSI που ορίζονται από την OMA[41] (Open Mobile Alliance). Λαμβάνουν τη μορφή κλήσεων RESTful για τις δύο διασυνδέσεις που ορίζονται στις προδιαγραφές διαχείρισης OMA NGSI, συγκεκριμένα NGSI-9 και NGSI-10.

Η διεπαφή OMA NGSI-10 χρησιμοποιείται για την ανταλλαγή πληροφοριών σχετικά με τις οντότητες και τα χαρακτηριστικό τους, δηλαδή αξίες χαρακτηριστικών και μεταδεδομένα.

[39] https://catalogue.fiware.org/enablers/publishsubscribe-context-broker-orion-context-broker
[40] https://forge.fiware.org/plugins/mediawiki/wiki/fiware/index.php/NGSI-9/NGSI-10_information_model
[41] http://www.openmobilealliance.org/wp/

Η διεπαφή OMA NGSI-9 χρησιμοποιείται για πληροφορίες διαθεσιμότητας σχετικά με τις οντότητες και τα χαρακτηριστικά τους. Εδώ, αντί να ανταλλάσσονται τιμές χαρακτηριστικών, ανταλλάσσονται πληροφορίες σχετικά με το ποιος πάροχος μπορεί να παρέχει ορισμένες τιμές χαρακτηριστικών.

Orion Context Broker – Δημιουργία Οντότητας

Κάθε φορά που δημιουργούμε ένα νέο χρήστη στην εφαρμογή μας, δημιουργούμε παράλληλα και την αντίστοιχη οντότητα (entity) στον context broker, για τον χρήστη αυτό. Η οντότητα περιλαμβάνει πληροφορίες για το κοντινότερο beacon και τον επιλεγμένο προορισμό, όπως οι πληροφορίες αυτές θα αποστέλλονται στο εξής από το τηλέφωνο του χρήστη μέσω της εφαρμογής πλοήγησης.

Η δημιουργία μιας νέας οντότητας στον Context Broker γίνεται μέσω ενός REST αιτήματος (κάνοντας χρήση της μεθόδου POST του HTTP πρωτοκόλλου για τη μεταφορά δεδομένων), το σώμα του οποίου περιλαμβάνει ενδεικτικά τα εξής:

```
{
  "contextElement" : {
    "type" : "user",
    "isPattern" : "false",
    "id" : "140",
    "attributes" : [
      {
        "name" : "uploadinfo",
        "type" : "string",
        "value" : "D7:F4:A2:FD:1B:F0rad0"
      }
    ]
  },
  "statusCode" : {
    "code" : "200",
    "reasonPhrase" : "OK"
  }
}
```

Orion Context Broker – Δημιουργία Συνδρομής σε Οντότητα

Μόλις δημιουργηθεί μια νέα οντότητα με την παραπάνω διαδικασία, χρειάζεται επίσης να δημιουργήσουμε μια νέα συνδρομή (subscribe) προς την οντότητα αυτή. Μέσω της συνδρομής μπορούμε να επιλέξουμε κάποιο ή κάποια από τα χαρακτηριστικά (attributes) της οντότητάς μας και να παρακολουθούμε για αλλαγές στην τιμή αυτών (συνθήκη "ONCHANGE"). Επίσης μπορούμε να ορίσουμε ένα URI αναφοράς, το οποίο θα καλεί ο Context Broker κάθε φορά που θα έχουμε αλλαγή σε κάποια συνδρομή, σύμφωνα με τις συνθήκες που ορίσαμε.

Οι συνθήκες που εμείς παρακολουθούμε και ενημερώνουμε το Back-End της εφαρμογής μας, είναι είτε η αλλαγή στην τιμή κάποιου αισθητήρα είτε η αλλαγή στον προορισμό που έχει επιλεγεί.

Η δημιουργία μιας νέας συνδρομής για μια οντότητα στον Context Broker γίνεται μέσω ενός REST αιτήματος (κάνοντας χρήση της μεθόδου POST του HTTP πρωτοκόλλου για τη μεταφορά δεδομένων), το σώμα του οποίου περιλαμβάνει ενδεικτικά τα εξής:

```
{
    "entities": [{
        "type": "user",
        "isPattern": "false",
        "id": "' . $last_id . '"
    }],
    "attributes": [],
    "reference": "http://147.27.50.121/anto/",
    "duration": "P12M",
    "notifyConditions": [{
        "type": "ONCHANGE",
        "condValues": ["uploadinfo"]
    }],
    "throttling": "PT1S"
}
```

Orion Context Broker – Διαγραφή Οντοτήτων και Συνδρομών

Όταν θέλουμε να διαγράψουμε ένα χρήστη από το σύστημα μας, χρειάζεται να διαγράψουμε από τον Context Broker τόσο την οντότητά που είχαμε δημιουργήσει για το χρήστη αυτό, όσο και όλες τις συνδρομές που αφορούσαν την οντότητα αυτή. Να σημειώσουμε εδώ ότι πρέπει να αφαιρεθούν και τα δύο παραπάνω ξεχωριστά, καθώς η αφαίρεση μόνο της οντότητας δεν συνεπάγεται σε αυτόματη διαγραφή και των συνδρομών της. Προκειμένου να θυμόμαστε το αναγνωριστικό της κάθε συνδρομής (subscription id) για να το χρησιμοποιήσουμε στο μέλλον σε πιθανή διαγραφή τη;, το αποθηκεύουμε στη βάση μας κατά την δημιουργία του.

Η διαγραφή συνδρομής πραγματοποιείται μέσω ενός REST αιτήματος, κάνοντας χρήση της DELETE μεθόδου του HTTP πρωτοκόλλου (*βλ. Πίνακα 2*).

Orion Context Broker – Ενημέρωση Συνδρομών

Η ενημέρωση μιας συνδρομής λαμβάνει χώρα κάθε φορά που αλλάζει η τιμή κάποιου χαρακτηριστικού μια οντότητας που μας ενδιαφέρει και παρακολουθούμε. Στην περίπτωσή μας, όπως έχουμε ήδη αναφέρει, η αλλαγή στο αναγνωριστικό του κοντινότερου beacon ή του επιλεγμένου προορισμού, ενημερώνει το σύστημά μας με σκοπό την αποστολή νέας οδηγίας πλοήγησης.

Η ενημέρωση μιας συνδρομής πραγματοποιείται μέσω ενός REST αιτήματος, κάνοντας χρήση της PUT μεθόδου του HTTP πρωτοκόλλου για μεταφορά δεδομένων (*βλ. Πίνακα 2*).

Πίνακας 2: Μέθοδοι HTTP αιτημάτων Orion Context Broker

Μέθοδος	URL Μεθόδου	Σώμα αιτήματος	Περιγραφή μεθόδου

POST	http://147.27.60.116:1026//v1/contextEntities/	{
		"id": "' . $last_id . '",
		"type": "user",
		"attributes": [{
		"name": "uploadinfo",
		"type": "string",
		"value": ""
		}]
		}
			Δημιουργία οντότητας

POST	http://147.27.60.116:1026/v1/subscribeContextv1/subscribeContext	{
		"entities": [{
		"type": "user",
		"isPattern": "false",
		"id": "' . $last_id . '"
		}],
		"attributes": [],
		"reference": "http://147.27.50.121/anto/",
		"duration": "P12M",
		"notifyConditions": [{
		"type": "ONCHANGE",
		"condValues": ["uploadinfo"]
		}],
		"throttling": "PT1S"
		}
			Δημιουργία συνδρομής σε οντότητα

PUT	http://147.27.60.116:1026/v1/contextEntities/type/user/id/" + user.id + "/attributes/uploadinfo	JSON-encoded: uploadinfo: string	Ενημέρωση πληροφοριών οντότητας
DELETE	http://147.27.60.116:1026//v1/contextEntities/{entity id}		Διαγραφή οντότητας
DELETE	http://147.27.60.116:1026/v2/subscriptions/{sub id}		Διαγραφή συνδρομής

4.2.1.3 Υπηρεσία Διαχείρισης Χρηστών

Η υπηρεσία περιλαμβάνει μεθόδους για τη διαχείριση των χρηστών του συστήματός μας. Η λογική όλων των μεθόδων βασίζεται στην ανάγνωση των δεδομένων κάποιου αιτήματος (request data) προερχόμενο από την αντίστοιχη εφαρμογή πελάτη. Ακολουθεί η δημιουργία ενός ερωτήματος (query) προς την MySQL βάση, με σκοπό να πραγματοποιηθεί η επιθυμητή ενέργεια. Εφόσον χρειαστεί να ανακτηθούν δεδομένα από τη βάση που ικανοποιούν το ερώτημα που θέσαμε, αυτά επιστρέφονται πίσω στην εφαρμογή ως απάντηση (response), μαζί με έναν κωδικό κατάστασης (response code). Χρειάζεται επίσης να τονίσουμε ότι και στην ίδια τη βάση δεδομένων υπάρχουν μέθοδοι[42] που αναλαμβάνουν την επεξεργασία και κατηγοριοποίηση ορισμένων καταχωρήσεων την στιγμή που αυτές εισάγονται. Έτσι διευκολύνεται κατά πολύ η ανάκτησή τους.

Από τις μεθόδους που περιλαμβάνει η υπηρεσία, χρήζουν ενδιαφέροντος αυτές τις Εισαγωγής και Διαγραφής. Συγκεκριμένα κατά τη δημιουργία ενός νέου χρήστη καλούνται οι δύο POST μέθοδοι του Orion Context Broker για δημιουργία νέας οντότητας και νέας συνδρομής προς αυτήν. Αντίστοιχα κατά την διαγραφή ενός χρήστη, αφού πρώτα ανακτηθούν από τη βάση οι κωδικοί οντότητας (entity id) και συνδρομής (subscription id) διαγράφονται και τα δύο καλώντας τις DELETE μεθόδους του Πίνακα 2.

Τέλος η μέθοδος που δημιουργεί κωδικούς πρόσβασης για τα τηλέφωνα των χρηστών, αφού τους αποθηκεύσει στη βάση για μελλοντική ταυτοποίηση, τους προωθεί σε μια υπηρεσία που κατασκευάσαμε με σκοπό την αυτόματη αποστολή τους στη διεύθυνση ηλεκτρονικού ταχυδρομείου που έχει δηλώσει ο αντίστοιχος χρήστης. Αυτό επιτυγχάνεται μέσω μιας βιβλιοθήκης της PHP, η οποία αναλαμβάνει την αποστολή email.[43]

Πίνακας 3: Μέθοδοι HTTP αιτημάτων Υπηρεσίας Διαχείρισης Χρηστών

Μέθοδος	**URL Μεθόδου**	**Σώμα αιτήματος**	**Περιγραφή μεθόδου**
GET	/users/		Ανάκτηση συνόλου χρηστών
GET	/users/{id}/log/		Ανάκτηση ιστορικού πλοήγησης χρήστη
POST	/users/	JSON-encoded: email: string name: string permissions: string	Προσθήκη νέου χρήστη
POST	/users/authorize/	JSON-encoded: code: string	Ταυτοποίηση χρήστη
DELETE	/users/{id}/		Διαγραφή χρήστη

[42] https://dev.mysql.com/doc/refman/5.7/en/triggers.html
[43] https://github.com/PHPMailer/PHPMailer

PUT	/users/{id}/	JSON-encoded: email: string name: string permissions: string	Επεξεργασία και ενημέρωση στοιχείων χρήστη
PUT	/users/{id}/code/	JSON-encoded: code: string	Δημιουργία νέου κωδικού χρήστη

4.2.1.4 Υπηρεσία Διαχείρισης Χώρων

Η συγκεκριμένη υπηρεσία έχει ως στόχο την εύκολη δημιουργία και διαχείριση χώρων. Στην χρήσιμη πληροφορία που κρατάμε για ένα χώρο στη βάση περιλαμβάνεται το μοναδικό αναγνωριστικό του, το όνομα αυτού, το αναγνωριστικό[44] (MAC address) του αισθητήρα που είναι συσχετισμένος με τον χώρο, μια προκαθορισμένη διαδρομή που οδηγεί σε αυτόν και τέλος, το σύνολο των γειτονικών τοποθεσιών με τις οποίες συνορεύει.

Η προκαθορισμένη διαδρομή είναι μια συμβολοσειρά (string), ο κάθε χαρακτήρας της οποίας αντιστοιχεί στον κωδικό μιας τοποθεσίας που ανήκει σε αυτή ενώ το τελευταίο σύμβολο είναι πάντοτε ο κωδικός της τοποθεσίας προορισμού. Για παράδειγμα ο συμβολισμός BCEGHFA δηλώνει την ακολουθία τοποθεσιών [B->C->E->G->H->F->A] με αφετηρία την θέση B και προορισμό τη θέση A.

Οι συνδεδεμένες τοποθεσίες είναι μια δυαδική string αναπαράσταση, η τιμή του κάθε χαρακτήρα της οποίας (0 ή 1), δηλώνει αν η τοποθεσία με αριθμό τη θέση του χαρακτήρα αυτού μέσα στη συμβολοσειρά συνορεύει με το χώρο. Για παράδειγμα ο συμβολισμός 1001101 δηλώνει ότι ο χώρος που εξετάζουμε συνορεύει με τις τοποθεσίες με κωδικούς 0, 3, 4 και 6.

Οι παραπάνω δύο αναπαραστάσεις, χρησιμοποιούνται από την υπηρεσία πλοήγησης, αφού αποκωδικοποιούνται από τον αλγόριθμο που αναπτύξαμε, προκειμένου να υπολογίσουμε την επόμενη τοποθεσία, δεδομένης της τρέχουσας σε μια διαδρομή.

Πίνακας 4: Μέθοδοι HTTP αιτημάτων Υπηρεσίας Διαχείρισης Χώρων

Μέθοδος	URL Μεθόδου	Σώμα αιτήματος	Περιγραφή μεθόδου
GET	/areas/		Ανάκτηση όλων των χώρων
GET	/areas/{id}/log/		Ανάκτηση πρόσφατης

[44] https://en.wikipedia.org/wiki/MAC_address

			δραστηριότητας χώρου
POST	/areas/	JSON-encoded: area_id: integer area_dd: string area_name: string area_path: string area_beacon: string	Προσθήκη νέου χώρου
PUT	/areas/{id}/	JSON-encoded: area_id: integer area_dd: string area_name: string area_path: string area_links: string area_beacon: string	Επεξεργασία και ενημέρωση πληροφοριών χώρου
DELETE	/areas/{id}/		Διαγραφή χώρου

4.2.1.5 Υπηρεσία Οδηγιών Πλοήγησης

Αποτελεί την σπουδαιότερη υπηρεσία του "Back-End" μας, αφού μέσω αυτής υπολογίζονται οι οδηγίες πλοήγησης που λαμβάνουν οι επισκέπτες στο τηλέφωνό τους. Η υπηρεσία καλείται αποκλειστικά από τον Orion Context Broker όταν ικανοποιηθεί η συνθήκη "ONCHANGE" στις συνδρομές που είχαμε νωρίτερα ορίσει για τους χρήστες (βλ. § 4.2.1.2). Περιλαμβάνει μία μόνο μέθοδο POST, το σώμα της οποίας περιέχει τις πληροφορίες που έστειλε ο Context Broker, δηλαδή το αναγνωριστικό του χρήστη της αντίστοιχης συνδρομής, το αναγνωριστικό του πλησιέστερου αισθητήρα που ανιχνεύτηκε και τον επιλεγμένο προορισμό.

Πίνακας 5: Μέθοδοι HTTP αιτημάτων Υπηρεσίας Οδηγιών Πλοήγησης

Μέθοδος	URL Μεθόδου	Σώμα αιτήματος	Περιγραφή μεθόδου
POST	/navigation/	JSON-encoded: user_id: integer beacon_id: string destination_id: string	Υπολογισμός οδηγίας για τον χρήστη με id = "user_id", ο οποίος βρίσκεται κοντά στο beacon με MAC Address = "beacon_id" και θέλει να φτάσει στη θέση με id = "destination_id"

Στο σημείο αυτό κρίνεται ορθό να παραθέσουμε τον αλγόριθμο υπολογισμού της επόμενης τοποθεσίας σε μια διαδρομή, δεδομένης της τρέχουσας:

1. Ανάκτησε από τη βάση δεομένων τις πληροφορίες της τρέχουσας τοποθεσίας location_id, location_name, location_links, όπου ο αισθητήρας αυτής έχει αναγνωριστικό "beacon_id".

a. Αν δεν βρέθηκαν πληροφορίες συσχετισμένης τοποθεσίας για το συγκεκριμένο "beacon id", σημαίνει πως η BLE συσκευή που εντοπίστηκε δεν αποτελεί κάποιο από τα beacons της υποδομής μας.

b. Τερμάτισε τον υπολογισμό και κατάγραψε στο log το περιστατικό ώστε το τηλέφωνο του χρήστη να επαναλάβει τη σάρωση.

2. Ανάκτησε από τη βάση δεομένων τη συμβολοσειρά (string) της προκαθορισμένης διαδρομής που οδηγεί στην τοποθεσία προορισμού με id = "destination_id".

3. Βρες μέσα στο string της διαδρομής τη θέση του χαρακτήρα, η τιμή του οποίου ταυτίζεται με το αναγνωριστικό της τρέχουσας τοποθεσίας "location_id".

4. Αν βρέθηκε η τρέχουσα τοποθεσία μέσα στο string της διαδρομής, θεώρησε την τιμή του αμέσως επόμενου χαρακτήρα ως το id της επόμενης θέσης. Αν επιπλέον η τρέχουσα τοποθεσία αποτελεί και τον τελευταίο χαρακτήρα της συμβολοσειράς, τότε αυτή ταυτίζεται με την τοποθεσία προορισμού.

5. Ανάκτησε από τη βάση δεδομένων την οδηγία κατεύθυνσης από την τρέχουσα προς την επόμενη θέση, όπως αυτές υπολογίστηκαν στο βήμα 4.

6. Αν η τρέχουσα θέση δεν ανήκει στην διαδρομή, δηλαδή το location_id δεν αποτελεί τιμή κανενός χαρακτήρα στο string διαδρομής, τότε:

   a. Για κάθε χαρακτήρα, δηλαδή για κάθε τοποθεσία, στο string διαδρομής, ξεκινώντας από το τέλος προς την αρχή ανάκτησε από τη βάση το id της αντίστοιχης τοποθεσίας.

   b. Αναζήτησε μέσα στη συμβολοσειρά των γειτόνων της τρέχουσας τοποθεσίας "location_links", εάν ο χαρακτήρας με θέση ίση με το id της τοποθεσίας που εξετάζουμε έχει τιμή 1, επομένως πρόκειται για τοποθεσία που συνορεύει με την τρέχουσα, ικανοποιώντας δηλαδή τη σχέση location_links[id] = 1.

   c. Εάν βρέθηκε τοποθεσία η οποία συνορεύει με τη τρέχουσα και συγχρόνως ανήκει και στην διαδρομή, τότε καταχώρησε στο log την οδηγία προς αυτή, διαφορετικά καταχώρησε πως ο χρήστης χάθηκε.

4.2.1.6 Υπηρεσία Διαχείρισης Μηνυμάτων

Η υπηρεσία περιλαμβάνει όλες εκείνες τις μεθόδους που είναι απαραίτητες για την περάτωση ενός ολοκληρωμένου συστήματος επικοινωνίας ανάμεσα στους χρήστες του συστήματος, μέσω της ανταλλαγής μηνυμάτων.

Πίνακας 6: Μέθοδοι HTTP αιτημάτων Υπηρεσίας Διαχείρισης Μηνυμάτων

Μέθοδος	URL Μεθόδου	Σώμα αιτήματος	Περιγραφή μεθόδου
GET	/messages/		Ανάκτηση όλων των εισερχομένων μηνυμάτων για το διαχειριστή
GET	/messages/to/{id}/		Ανάκτηση όλων των μηνυμάτων προς έναν συγκεκριμένο παραλήπτη
POST	/messages/	JSON-encoded: text: string from: integer to: integer	Αποστολή μηνύματος
DELETE	/messages/{id}/		Διαγραφή μηνύματος

Στο σημείο αυτό αξίζει να σημειώσουμε ότι κατά την αποστολή μηνύματος, αυτό που συμβαίνει στην πραγματικότητα είναι η δημιουργία μιας νέας εγγραφής στον πίνακα «Μηνύματα» στη βάση δεδομένων, με τον τίτλο, το περιεχόμενο και το αναγνωριστικό του παραλήπτη αυτού. Η εφαρμογή πελάτη από την πλευρά της, είναι υπεύθυνη μέσω μιας διαδικασίας δειγματοληψίας (AJAX Long Polling[45]) να ελέγχει σε τακτά χρονικά διαστήματα στο server, αν υπάρχουν νέα μη αναγνωσμένα μηνύματα ώστε να τα «κατεβάσει» τοπικά στη συσκευή.

4.2.1.7 Υπηρεσία Διαχείρισης Ειδοποιήσεων Συστήματος

Αποτελώντας ουσιαστικά υποκατηγορία της Υπηρεσίας Διαχείρισης Μηνυμάτων (βλ. και Διάγραμμα Κλάσεων Συστήματος § 3.5), η συγκεκριμένη υπηρεσία κληρονομεί όλες τις μεθόδους της προηγούμενης.

[45] https://techoctave.com/c7/posts/60-simple-long-polling-example-with-javascript-and-jquery

Μια ειδοποίηση αποτελεί μια καταχώρηση στη βάση δεδομένων, η οποία δημιουργείται αυτόματα μέσω εναυσμάτων (triggers), όταν ικανοποιείται κάποια συνθήκη ενδιαφέροντος. Μια τέτοια συνθήκη μπορεί να είναι η απώλεια προσανατολισμού ενός επισκέπτη, η παρουσία του σε απαγορευμένη ζώνη κλπ. Ωστόσο υπάρχουν και ειδοποιήσεις με καθαρά ενημερωτικό περιεχόμενο, όπως για παράδειγμα ένα αυτοματοποιημένο μήνυμα καλωσορίσματος ("Message of the Day").

Σε κάθε περίπτωση, μια ειδοποίηση εκλαμβάνεται ως ένα απλό μήνυμα από το σύστημά μας, θεωρώντας ως αποστολέα πάντοτε τον διαχειριστή. Λόγω του ότι μια ειδοποίηση εκτός από απλό κείμενο ενδέχεται να περιλαμβάνει και περιεχόμενο πολυμέσων (εικόνα και ήχο), καλό είναι να χρησιμοποιούνται οι εξειδικευμένες μέθοδοι του Πίνακα 7 για την λήψη τους από την εφαρμογή πελάτη. Οι μέθοδοι αυτοί κωδικοποιούν στο σώμα απόκρισης (response) το περιεχόμενο των πολυμέσων σε μορφή συμβολοσειράς Base64,[46] με σκοπό την μετάδοση δυαδικών αρχείων μέσω του HTTP πρωτοκόλλου.

Πίνακας 7: Μέθοδοι HTTP αιτημάτων Υπηρεσίας Διαχείρισης Ειδοποιήσεων

Μέθοδος	URL Μεθόδου	Σώμα αιτήματος	Περιγραφή μεθόδου
GET	/notifications/		Ανάκτηση όλων των διαθέσιμων ειδοποιήσεων
GET	/notifications/{id}/		Ανάκτηση μιας συγκεκριμένης ειδοποίησης

4.2.1.8 Υπηρεσία Αποστολής Μηνυμάτων Ηλεκτρονικού Ταχυδρομείου

Η υπηρεσία κάνει χρήση του PHPMailer[47], μιας βιβλιοθήκης της PHP, που επιτρέπει την αποστολή email προγραμματιστικά. Για την αποστολή email απαιτείται η σύνδεση σε έναν SMTP[48] server, τον οποίο είτε μπορούμε να έχουμε στήσει εμείς οι ίδιοι στο σύστημά μας, είτε μπορούμε να κάνουμε χρήση κάποιου εξωτερικού παρόχου. Επιλέξαμε για λόγους απλότητας να χρησιμοποιήσουμε τον Gmail λογαριασμό μας, συνδεόμενοι στο smtp.gmail.com, που πρόκειται για τον SMTP host της Google.

Η αποστολή email γίνεται συνήθως με σκοπό τη γνωστοποίηση των κωδικών πρόσβασης στους χρήστες της εφαρμογής για το τηλέφωνο (βλ. § 4.1.2).

[46] https://en.wikipedia.org/wiki/Base64
[47] https://github.com/PHPMailer/PHPMailer/blob/master/examples/gmail.phps
[48] https://en.wikipedia.org/wiki/Simple_Mail_Transfer_Protocol

Για να χρησιμοποιήσουμε την υπηρεσία, αρκεί να καλέσουμε την POST μέθοδο του Πίνακα 8, βάζοντας τις παραμέτρους του email στο σώμα του αιτήματος κωδικοποιημένες κατά JSON[49].

Πίνακας 8: Μέθοδοι HTTP αιτημάτων Υπηρεσίας Αποστολής Μηνυμάτων Ηλεκτρονικού Ταχυδρομείου

Μέθοδος	URL Μεθόδου	Σώμα αιτήματος	Περιγραφή μεθόδου
POST	/mailer/	JSON-encoded: email_address: string email_title: string email_body: string	Αποστολή νέου μηνύματος ηλεκτρονικού ταχυδρομείου στη διεύθυνση "email_address" με θέμα "email_title" και σώμα μηνύματος "email_body"

4.3 Ανάλυση Απόδοσης Back-End

Προκειμένου να μετρήσουμε την απόδοση του συστήματός μας λάβαμε υπόψιν το πιο απαιτητικό σενάριο χρήσης αυτού, δηλαδή την περίπτωση της πλοήγησης. Το συγκεκριμένο σενάριο εξετάστηκε με τις εξής παραδοχές:

1. Το τηλέφωνο μας θεωρούμε πως έχει ήδη εντοπίσει κάποιον κοντινό αισθητήρα και έχει λάβει από αυτόν τις πληροφορίες ενδιαφέροντος (μέτρηση απόστασης από τον αισθητήρα και φυσική διεύθυνση αυτού). Αυτό γίνεται για να διαχωρίσουμε και να απομονώσουμε όσες καθυστερήσεις προέρχονται από το τοπικό δίκτυο των ασύρματων αισθητήρων, για τις οποίες και θα μιλήσουμε στην αμέσως επόμενη ενότητα.
2. Οι μετρήσεις αποστέλλονται σε μορφή JSON από το τηλέφωνο στην Υπηρεσία Διαχείρισης Συμβάντων και Συνδρομών. Αν μια μέτρηση είναι διαφορετική από την προηγούμενή της, τότε η Υπηρεσία Διαχείρισης Συμβάντων και Συνδρομών ενημερώνει την Υπηρεσία Πλοήγησης, διαφορετικά το αίτημα δεν χρειάζεται να δρομολογηθεί περεταίρω . Στην περίπτωσή μας φροντίσαμε όλες οι μετρήσεις να είναι διαφορετικές ώστε να συνεχίζεται ανεμπόδιστα η δρομολόγηση των αιτημάτων μεταξύ υπηρεσιών.
3. Γίνονται τα ερωτήματα στη βάση δεδομένων για την ανάκτηση των απαιτούμενων πληροφοριών (τοποθεσίες, προκαθορισμένες διαδρομές, δικαιώματα πρόσβασης) και στη συνέχεια εκτελείται ο αλγόριθμος πλοήγησης. Η οδηγία πλοήγησης αποθηκεύεται στην βάση δεδομένων και χρησιμοποιείται για την δημιουργία της απάντησης του αιτήματος αποστολής οδηγιών.

[49] https://www.w3schools.com/js/js_json_intro.asp

Η υποδομή του Back-End για την ικανοποίηση του παραπάνω σεναρίου χρήσης στεγάζεται στο περιβάλλουν Intellicloud του Πολυτεχνείου Κρήτης, αποτελούμενη από δύο εικονικές μηχανές. Στην πρώτη εικονική μηχανή εκτελείται η Υπηρεσία Διαχείρισης Συμβάντων και Συνδρομών (Context Broker), ενώ στην δεύτερη εκτελούνται όλες οι υπόλοιπες υπηρεσίες που εμείς χρειάστηκε να υλοποιήσουμε. Τα τεχνικά χαρακτηριστικά των εικονικών μηχανών είναι τα ακόλουθα:

CPU	x86_64 @ 2.8Ghz
Memory	2048MB
HDD	20GB
OS	Ubuntu 14.04

Στο σενάριο πλοήγησης που περιγράψαμε, καταλαβαίνουμε πως έχουμε επικοινωνία μεταξύ δύο διαφορετικών εικονικών μηχανών, στο ίδιο ωστόσο ΥΝ. Πιο συγκεκριμένα κάθε αίτημα πλοήγησης ακολουθεί την εξής πορεία μέχρι να απαντηθεί:

Κινητή Συσκευή → Υπηρεσία Διαχείρισης Συμβάντων και Συνδρομών (VM1) → Υπηρεσία Πλοήγησης (VM2)[50]

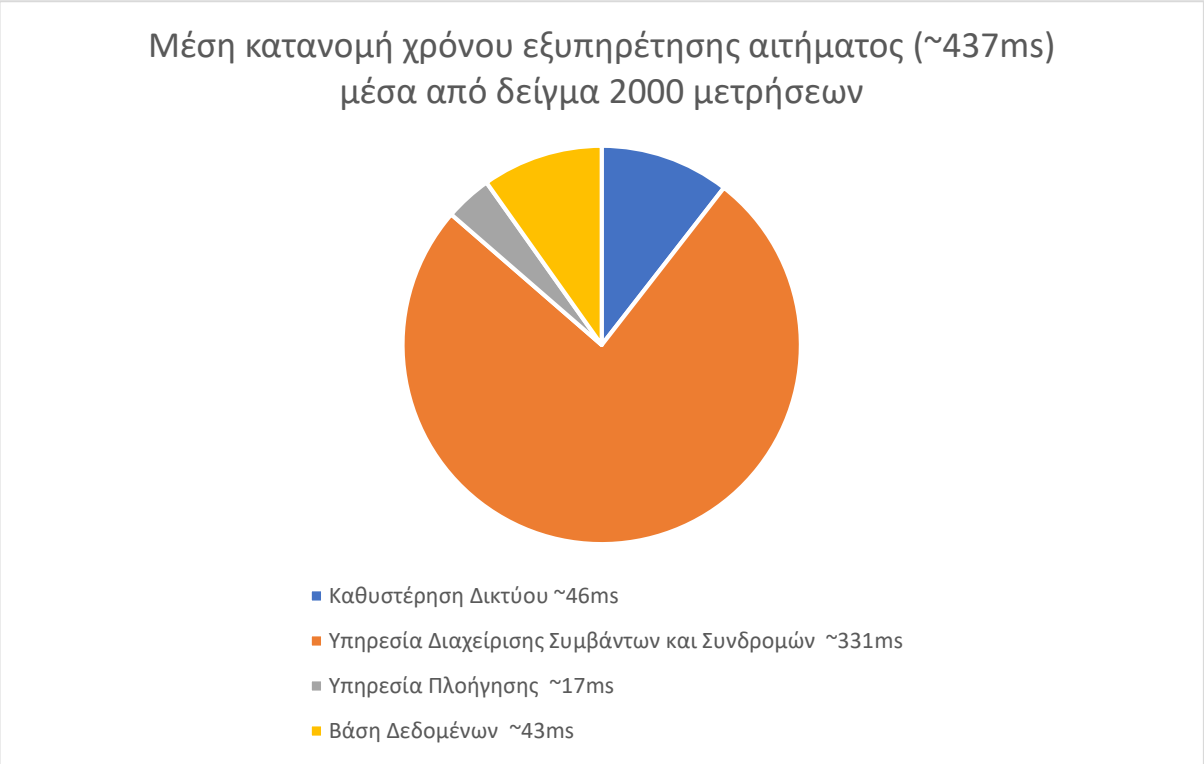
Είναι επόμενο λοιπόν κατά την απάντηση τέτοιων αιτημάτων να υπάρχουν καθυστερήσεις τόσο δικτύου, λόγω κόμβων (ανεξάρτητων υπηρεσιών), όσο και επεξεργασίας δεδομένων στον κάθε κόμβο. Σε αυτό το σημείο κρίνεται σκόπιμο να υπενθυμίσουμε πως η επικοινωνία μεταξύ των υπηρεσιών, ακόμα και αν βρίσκονται στην ίδια εικονική μηχανή, γίνεται μέσω REST αιτημάτων χρησιμοποιώντας το HTTP πρωτόκολλο. Κάτι τέτοιο προσθέτει επιπλέον καθυστερήσεις (δικτύου και πρωτοκόλλου), ωστόσο η επιλογή αυτή έγινε για να δείξουμε την ανεξαρτησία των επιμέρους υπηρεσιών της εφαρμογής μας. Θα μπορούσε για παράδειγμα οι υπηρεσίες να ήταν ακόμα και σε διαφορετικά ΥΝ, το μόνο που θα άλλαζε στον κώδικά μας να είναι το URI της κάθε υπηρεσίας.

Λαμβάνοντας υπόψιν τα παραπάνω δημιουργήσαμε ένα αυτοσχέδιο πείραμα, μετρώντας το χρόνο όπου ένα αίτημα χρειάζεται για να περάσει από κάθε μία υπηρεσία ξεχωριστά μέχρι να απαντηθεί. Για να το πετύχουμε αυτό χρησιμοποιήσαμε την εντολή round(microtime(true) * 1000) της PHP, η οποία μας δείχνει τον χρόνο σε milliseconds. Για να μετρήσουμε την καθυστέρηση internet μεταξύ της κινητής μας συσκευής και του ΥΝ, χρησιμοποιήσαμε την

[50] Στο εξής, στα πειράματα που ακολουθούν, όταν αναφερόμαστε σε ένα «ερώτημα» ή «αίτημα» αυτό αντιστοιχεί σε ένα ερώτημα πλοήγησης, το οποίο με την σειρά του για να εξυπηρετηθεί δημιουργεί μια αλληλουχία από διαδοχικά αιτήματα στις υπηρεσίες Διαχείρισης Συμβάντων και Συνδρομών, Συνδεσιμότητας, Πλοήγησης και Αποθήκευσης Δεδομένων (βλ. § 3.5.2 για περιγραφή των υπηρεσιών).

εντολή traceroute από υπολογιστή που βρισκόταν στο ίδιο τοπικό δίκτυο με την κινητή συσκευή, μέσω ασύρματης σύνδεσης Wi-Fi.

Για να εξασφαλίσουμε μεγαλύτερη αξιοπιστία στο δείγμα των μετρήσεων μας, φροντίσαμε μέσω επαναληπτικής διαδικασίας, να εκτελέσουμε το παραπάνω πείραμα αρκετές εκατοντάδες φορές, με τους χρόνους που προκύπταν κάθε φορά να αποθηκεύονται σε βάση δεδομένων αποκλειστικά για τους σκοπούς του πειράματος. Συγκεκριμένα είχαμε απενεργοποιημένους όλους τους ελέγχους εγκυρότητας κατά την εισαγωγή των μετρήσεων στη βάση μας, ώστε να ελαχιστοποιήσουμε τυχόν καθυστερήσεις που οφείλονται στην ίδια την πειραματική διαδικασία. Μετά την συλλογή των αποτελεσμάτων, υπολογίσαμε τους μέσους όρους των καθυστερήσεων της κάθε υπηρεσίας, καθώς και το ποσοστό χρόνου που η καθεμιά επιβαρύνει τη συνολική διάρκεια για την εξυπηρέτηση του αιτήματος. Τα αποτελέσματα που προέκυψαν από το παραπάνω πείραμα μέσα από δείγμα 2000 αιτημάτων πλοήγησης είναι τα ακόλουθα:



Παρατηρούμε λοιπόν πως από τα 437 milliseconds που χρειάζονται σχεδόν για να απαντηθεί ένα ερώτημα, τα 331 milliseconds οφείλονται στην Υπηρεσία Διαχείρισης Συμβάντων και Συνδρομών, ποσοστό που, όπως βλέπουμε και στο γράφημα, αντιστοιχεί στο 75% περίπου του συνολικού χρόνου εκτέλεσης του αιτήματος. Η εν λόγω υπηρεσία εκτελείται ως instance σε ξεχωριστό VM στο Intellicloud και χρησιμοποιήθηκε έτοιμη, χωρίς κάποια παραμετροποίηση ή επέμβαση στον κώδικα αυτής από τη μεριά μας. Μπορούμε λοιπόν αυθαίρετα να υποθέσουμε ότι η καθυστέρηση της υπηρεσίας οφείλεται στον τρόπο λειτουργίας της, χωρίς ωστόσο να μπορούμε να καταλήξουμε σε ασφαλή συμπεράσματα. Συγκεκριμένα κατά την αποστολή σε αυτήν ενός PUT αιτήματος για την ενημέρωση κάποιας μέτρησης, θα

πρέπει αρχικά να ανακτηθούν από τη βάση δεδομένων της υπηρεσίας οι προηγούμενες μετρήσεις που υπήρχαν για την συγκεκριμένη οντότητα. Κατόπιν πρέπει να γίνει σύγκριση με τη νέα τιμή, να αποθηκευτεί ξανά πίσω στη βάση δεδομένων το αποτέλεσμα αν είναι διαφορετικό, ενώ παράλληλα αν ικανοποιείται η συνθήκη "ONCHANGE", θα πρέπει να δημιουργηθεί από την υπηρεσία ένα POST αίτημα στην διεύθυνση που έχουμε ορίσει ως επιστροφή (callback), δηλαδή να δρομολογηθεί το αίτημα προς την Υπηρεσία Πλοήγησης.

Η Υπηρεσία Διαχείρισης Συμβάντων και Συνδρομών Orion χρησιμοποιεί MongoDB για την αποθήκευση των οντοτήτων. Πρόκειται για μία μη σχεσιακή βάση δεδομένων (NoSQL Database)[51], κύριο πλεονέκτημα της οποίας είναι η πολύ πιο γρήγορη εισαγωγή νέων καταχωρήσεων, αφού δεν γίνονται έλεγχοι εξαρτήσεων μεταξύ υπαρχόντων αποθηκευμένων κλειδιών στους πίνακες. Δεν ισχύει το ίδιο ωστόσο και για την ανάκτηση ενός κλειδιού από τη βάση (SELECT), αφού η έλλειψη συσχετίσεων επιβραδύνει κατά πολύ την αναζήτηση, συντελώντας σε σημαντικά μεγαλύτερους χρόνους, όπως έχει διαπιστωθεί και από άλλα πειράματα[52]. Προκειμένου λοιπόν να εκτελεστεί η εντολή UPDATE στη βάση (η λειτουργία της οποίας μοιάζει με της SELECT, αφού και εδώ απαιτείται αναζήτηση), κατά τον έλεγχο της συνθήκης "ONCHANGE" και την ανανέωση των τιμών, είναι δυνατόν να προκύψει πολύ μεγάλη καθυστέρηση, ειδικά στην περίπτωση που υπάρχουν ήδη αρκετές καταχωρήσεις. Ένας προτεινόμενος τρόπος προκειμένου να βελτιωθεί η απόδοση της MongoDB κατά την αναζήτηση είναι η δημιουργία προσαρμοσμένων ευρετηρίων (custom indices), μέθοδος η οποία ωστόσο δεν δοκιμάστηκε από εμάς, αφού στα πλαίσια της υλοποίησής μας δεν προχωρήσαμε σε τροποποιήσεις στο instance της υπηρεσίας. Επίσης ο Context Broker που διέθετε το Intellicloud είναι μια αρκετά παλιά έκδοση (0.11.0-1x86_64), γεγονός που σημαίνει πως η ενσωματωμένη MongoDB δεν διέθετε τις τελευταίες βελτιώσεις απόδοσης (performance optimizations), οι οποίες μπορεί να μείωναν σημαντικά τους χρόνους.

Προκειμένου να διαπιστώσουμε την απόδοση του συστήματός μας σε μεγάλο αριθμό αιτημάτων χρησιμοποιήσαμε το εργαλείο ApacheBench[53], το οποίο μας επιτρέπει να στέλνουμε πολλά ταυτόχρονα αιτήματα. Έχουμε τη δυνατότητα να καθορίσουμε τις παραμέτρους που αφορούν το πλήθος αιτημάτων, καθώς και πόσα από αυτά θα υπόκεινται σε ταυτόχρονη εκτέλεση. Για την παρακολούθηση της συμπεριφοράς των φυσικών πόρων του συστήματος μας κατά την διάρκεια διεξαγωγής των πειραμάτων χρησιμοποιήσαμε το εργαλείο HTOP. Το HTOP είναι ένα ελαφρύ πρόγραμμα διαχείρισης διεργασιών που εκτελείται κατευθείαν μέσω γραμμής εντολών και μας επιτρέπει να παρακολουθήσομε σε πραγματικό χρόνο την κατανάλωση πόρων ανά διεργασία, αλλά και συνολικά του συστήματός.

Για αρχή δοκιμάσαμε 2000 αιτήματα στη σειρά (πολλαπλότητα 1), με τα αποτελέσματα να φαίνονται στον Πίνακα 9:

[51] https://www.mongodb.com/nosql-explained
[52] https://github.com/webcaetano/mongo-mysql
[53] https://httpd.apache.org/docs/2.4/programs/ab.html

Πίνακας 9: Χρόνος απάντησης σε 2000 αιτήματα ανά 1 ταυτόχρονα

Ποσοστό αιτημάτων που εξυπηρετήθηκαν	Χρόνος (ms)
50%	435
66%	438
75%	451
80%	453
90%	476
95%	519
98%	586
99%	645
100%	890

Ο μέσος χρόνος εκτέλεσης ανά ερώτημα ήταν 441ms ενώ ο ρυθμός μετάδοσης κυμαινόταν στα 0.39Kbytes/sec. Τα αποτελέσματα αυτά για σειριακή εκτέλεση αιτημάτων ήταν αναμενόμενα, καθώς η εκτέλεση κάθε αιτήματος δεν επηρεάζει τα υπόλοιπα, ενώ συμφωνούν στην πλειονότητά τους με τις μετρήσεις του προηγούμενου πειράματος για το ένα αίτημα. Επίσης δεν παρατηρήθηκε επιπλέον χρήση υπολογιστών πόρων σε όλη την διαδικασία.

Συνεχίζουμε με το επόμενο πείραμα (Πίνακας 10), όπου εκτελούμε 2000 αιτήματα ανά 50 ταυτόχρονα:

Πίνακας 10: Χρόνος απάντησης σε 2000 αιτήματα ανά 50 ταυτόχρονα

Ποσοστό αιτημάτων που εξυπηρετήθηκαν	Χρόνος (ms)
50%	642
66%	655
75%	671
80%	678
90%	704

95%	725
98%	753
99%	767
100%	798

Ο μέσος χρόνος εκτέλεσης ανά ομάδα αιτημάτων ήταν 652ms (13.04ms ανά αίτημα) ενώ ο ρυθμός μετάδοσης κυμαινόταν στα 9.07Kbytes/sec. Η χρήση της CPU στην περίπτωση αυτή ανέρχεται στα 30-35% ενώ η κατανάλωση μνήμης στα 330MB.

Συνεχίζουμε με το επόμενο πείραμα (Πίνακας 11), όπου εκτελούμε 2000 αιτήματα ανά 100 ταυτόχρονα:

Πίνακας 11: Χρόνος απάντησης σε 2000 αιτήματα ανά 100 ταυτόχρονα

Ποσοστό αιτημάτων που εξυπηρετήθηκαν	Χρόνος (ms)
50%	1226
66%	1240
75%	1251
80%	1264
90%	1278
95%	1304
98%	1320
99%	1352
100%	1436

Ο μέσος χρόνος εκτέλεσης ανά ομάδα αιτημάτων ήταν 1232ms (12.32ms ανά αίτημα) ενώ ο ρυθμός μετάδοσης κυμαινόταν στα 10.17Kbytes/sec. Η χρήση της CPU στην περίπτωση αυτή ανέρχεται στα 45-50% ενώ η κατανάλωση μνήμης στα 440MB.

Συνεχίζουμε με το επόμενο πείραμα (Πίνακας 12), όπου εκτελούμε 2000 αιτήματα ανά 150 ταυτόχρονα:

Πίνακας 12: Χρόνος απάντησης σε 2000 αιτήματα ανά 150 ταυτόχρονα

Ποσοστό αιτημάτων που εξυπηρετήθηκαν	Χρόνος (ms)
50%	1583
66%	1613
75%	1634
80%	1657
90%	1723
95%	4925
98%	9202
99%	11087
100%	14194

Ο μέσος χρόνος εκτέλεσης ανά ομάδα αιτημάτων ήταν 2501ms (16.67ms ανά αίτημα) ενώ ο ρυθμός μετάδοσης κυμαινόταν στα 9.54Kbytes/sec. Η χρήση της CPU στην περίπτωση αυτή ανέρχεται στα 65-70% ενώ η κατανάλωση μνήμης στα 490MB. Παρατηρούμε πως αρχίζει να υπάρχει καθυστέρηση στην ταυτόχρονη εκτέλεση πολλαπλών αιτημάτων καθώς σιγά σιγά αρχίζει να αυξάνεται αισθητά η χρήση υπολογιστικών πόρων.

Συνεχίζουμε με το τελευταίο πείραμα (Πίνακας 13), όπου εκτελούμε 2000 αιτήματα ανά 300 ταυτόχρονα:

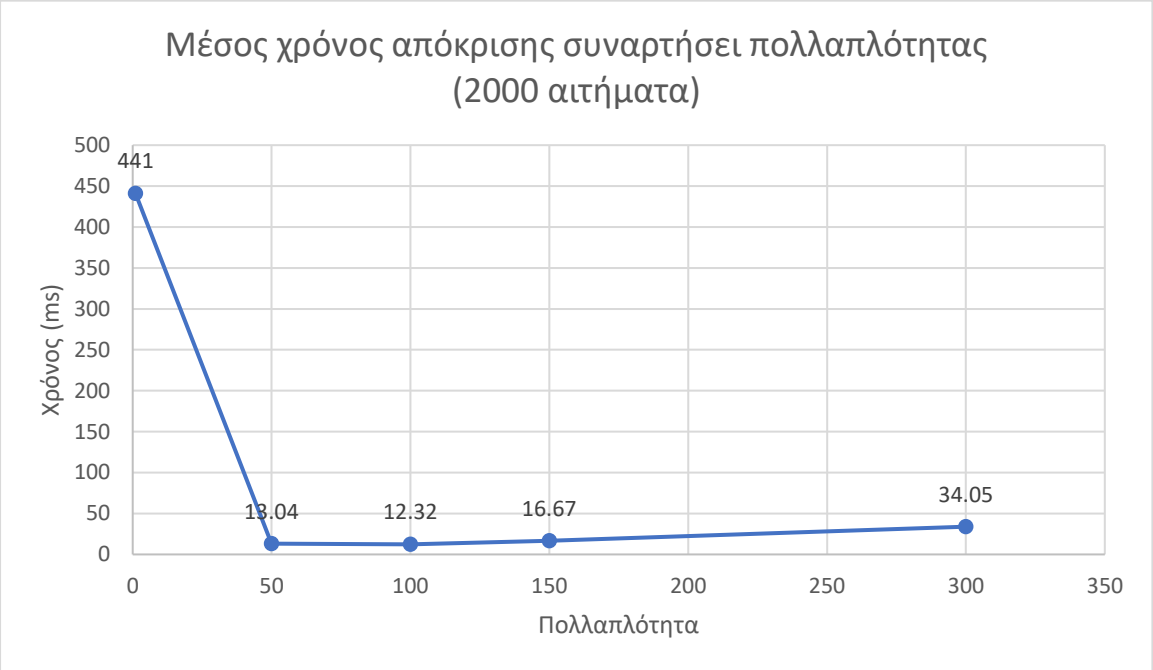
Πίνακας 13: Χρόνος απάντησης σε 2000 αιτήματα ανά 300 ταυτόχρονα

Ποσοστό αιτημάτων που εξυπηρετήθηκαν	Χρόνος (ms)
50%	1720
66%	1803
75%	4156
80%	5167
90%	9457

95%	12005
98%	25375
99%	34577
100%	68741

Ο μέσος χρόνος εκτέλεσης ανά ομάδα αιτημάτων ήταν 10217ms (34.05ms ανά αίτημα) ενώ ο ρυθμός μετάδοσης κυμαινόταν στα 8.11Kbytes/sec. Η χρήση της CPU στην περίπτωση αυτή ανέρχεται σε ~95% ενώ η κατανάλωση μνήμης στα 630MB. Παρατηρούμε πως υπάρχει πλέον σημαντική καθυστέρηση στην ταυτόχρονη εκτέλεση πολλαπλών αιτημάτων, καθώς η χρήση του επεξεργαστή είναι σχεδόν μέγιστη.

Στο παρακάτω γράφημα παρουσιάζουμε τα συγκεντρωτικά αποτελέσματα όλων των πειραμάτων μας.



Παρατηρούμε πως η ταυτόχρονη εκτέλεση αιτημάτων μειώνει δραστικά τους χρόνους απόκρισης σε σχέση με τη σειριακή τους εκτέλεση. Κάτι τέτοιο είναι αναμενόμενο καθώς γίνεται χρήση πολλαπλών νημάτων από τον εξυπηρετητή με αποτέλεσμα να μην χρειάζεται να περιμένουμε να απαντηθεί ένα ερώτημα προκειμένου να προχωρήσουμε στην διεκπεραίωση του επόμενου. Βλέπουμε ότι ακόμα και στην χειρότερη περίπτωση με πολλαπλότητα 300, παρόλο που ο φόρτος του εξυπηρετητή μεγιστοποιείται, ο μέσος χρόνος εκτέλεσης ανά αίτημα εξακολουθεί και είναι κατά πολύ (σχεδόν 13 φορές) μικρότερος σε σχέση με το χρόνο που απαιτείται για τη σειριακή εκτέλεση των αιτημάτων.

Τέλος βλέπουμε πως η βέλτιστη λειτουργία του συστήματός μας για την εξυπηρέτηση των 2000 αιτημάτων επιτυγχάνεται όταν έχουμε ταυτόχρονη εκτέλεση αυτών σε ομάδες είτε των 50 είτε των 100. Στην πρώτη περίπτωση εκτελούνται λιγότερα ταυτόχρονα αιτήματα, 50 τη φορά, δαπανάται επομένως και μικρότερη επεξεργαστική ισχύς. Στη δεύτερη περίπτωση τα αιτήματα απαντώνται περισσότερα μαζί, ανά 100 συγχρόνως, με συνέπεια όμως να επιβαρύνεται και περισσότερο το σύστημά μας. Παρατηρείται λοιπόν ένα trade-off ανάμεσα στην πολλαπλότητα αιτημάτων και τους απαιτούμενους υπολογιστικούς πόρους, το οποίο και αυτό ήταν αναμενόμενο. Φυσικά όταν η πολλαπλότητα αυξηθεί και άλλο, οι χρόνοι αρχίζουν και αυξάνονται καθώς το σύστημά μας ταλαιπωρείται όλο και περισσότερο.

4.4 Απόδοση της Εφαρμογής για Κινητές Συσκευές

Η απόδοση της εφαρμογής για κινητές συσκευές εξαρτάται από τα τεχνικά χαρακτηριστικά της συσκευής στην οποία αυτή τρέχει, συνεπώς, αν και πρόκειται για μια απλή εφαρμογή, ενδέχεται να υπάρχουν διαφορές στην απόδοσή της ανά συσκευή. Η συσκευή στην οποία δοκιμάσαμε την εφαρμογή μας είχε τις εξής προδιαγραφές:

CPU	32-bit ARM Quad-Core @ 2.2Ghz
Memory	2048MB
Flash Storage	32GB
OS	Android 5.1

Δεν παρατηρήθηκε κάποια αύξηση στους πόρους που χρησιμοποιούνταν από τη συσκευή μας ενώ η κατανάλωση μνήμης ανήλθε στα 93MB. Το μεγάλο ποσό της μνήμης που δαπανάται οφείλεται στο γεγονός ότι η εφαρμογή μας δεν είναι εγγενής Android εφαρμογή, άλλα εκτελείται μέσα από το περιβάλλον του Apache Cordova. Ωστόσο όλες σχεδόν οι σύγχρονες συσκευές έχουν μνήμη αρκετά μεγάλη, ώστε ο παραπάνω περιορισμός δεν αποτελεί πρακτικά εμπόδιο στη χρήση της.

Το μεγαλύτερο πρόβλημα ωστόσο στην εφαρμογή για το τηλέφωνο εντοπίζεται στους χρόνους σάρωσης που απαιτούνται για τους BLE αισθητήρες. Οι αισθητήρες που χρησιμοποιήσαμε για να δοκιμάσουμε τη λειτουργία πλοήγησης είναι τύπου ανίχνευσης προσέγγισης (proximity beacons) με κατασκευαστή την εταιρία Estimote. Τα τεχνικά χαρακτηριστικά τους, όπως αυτά αναγράφονται στην ιστοσελίδα του κατασκευαστή, είναι τα ακόλουθα:

MCU	Bluetooth® SoC ARM® Cortex®-M4 32-bit processor with FPU 64 MHz Core speed 512 kB Flash memory 64 kB RAM memory
Radio: 2.4 GHz transceiver	Bluetooth® 4.2 LE standard Range: up to 70 meters (230 feet) Output Power: -20 to +4 dBm in 4 dB steps, “Whisper mode” -40 dBm Sensitivity: -96 dBm Frequency range: 2400 MHz to 2483.5 MHz No. of channels: 40 Adjacent channel separation: 2 MHz Modulation: GFSK (FHSS) Antenna: PCB Meander, Monopole Antenna Gain: 0 dBi Over-the-air data rate: 1 Mbps (2 Mbps supported)
Sensors	Motion sensor (Ultra-low-power, high-performance, 3-axis "femto" accelerometer) Temperature sensor

Στη διάθεσή μας, για τις ανάγκες της εργασίας αυτής, είχαμε εννέα (9) τέτοιους αισθητήρες. Δοκιμάσαμε διάφορα σενάρια πλοήγησης με διαφορετικό αριθμό αισθητήρων κάθε φορά. Ο μέσος χρόνος πραγματοποίησης ενός κύκλου σάρωσης από το τηλέφωνό μας για BLE συσκευές στην εμβέλεια του ήταν περίπου 5000 milliseconds. Σε κάθε νέα σάρωση εντοπίζονταν ένας – ένας όλοι οι αισθητήρες με μια μικρή καθυστέρηση - της τάξης των 300 milliseconds - ο καθένας από τον προηγούμενό του. Με την σάρωση κάθε αισθητήρα λαμβάναμε ταυτόχρονα και την απόσταση του από το τηλέφωνό, ενώ το API της εφαρμογής φρόντιζε για την ταξινόμηση όλων των αποτελεσμάτων βάσει αύξουσας απόστασης. Υπήρχαν ωστόσο περιπτώσεις όπου ενώ κάποιοι αισθητήρες βρίσκονταν στην εμβέλεια της συσκευής αυτή αποτύγχανε να τους σαρώσει, με αποτέλεσμα να μεσολαβούν επιπλέον κύκλοι σάρωσης μέχρι να αναγνωριστούν όλοι.

Ένα άλλο θέμα που πρόκυπτε είναι πως η απόσταση που εντοπίζονταν οι αισθητήρες σε σχέση με το τηλέφωνό μας ήταν ανακριβής και πολλές φορές παραπλανητική. Αν υπήρχαν πολλοί αισθητήρες στην εμβέλεια, το τηλέφωνό μας κατάφερνε και τους εντόπιζε, ωστόσο σε πολλές περιπτώσεις κάποιοι που βρίσκονταν σε μεγαλύτερη απόσταση θεωρούνταν εσφαλμένα πως βρίσκονται πιο κοντά. Άμεση συνέπεια των παραπάνω ήταν να υπάρχουν προβλήματα στην πλοήγηση, με λήψη οδηγιών μετά από πολύ μεγάλη καθυστέρηση αρκετών δευτερολέπτων ή λήψη εσφαλμένων οδηγιών εφόσον ο εγγύτερος αισθητήρας δεν είχε

εντοπιστεί σωστά. Θα έπρεπε σταθούμε για αρκετά, τουλάχιστον 10 δευτερόλεπτα, δίπλα στον κάθε αισθητήρα κατά μήκος της διαδρομής προκειμένου οι μετρήσεις να σταθεροποιηθούν και να λάβουμε σωστές οδηγίες.

Δοκιμές έγιναν μειώνοντας σταδιακά τον αριθμό των αισθητήρων εντός εμβέλειας κατά μήκος της διαδρομής, απομακρύνοντάς τους αρκετά μεταξύ τους ώστε να μην εντοπίζονται πολλοί ταυτόχρονα, με σκοπό τη μείωση των παρεμβολών. Στην περίπτωση των λιγότερων αισθητήρων εντός εμβέλειας παρατηρήθηκε μείωση στα σφάλματα, αν και στην πραγματικότητα η μέτρηση της απόστασης εξακολουθούσε να είναι συχνά ανακριβής. Όσον αφορά την καθυστέρηση δεν παρατηρήθηκαν ιδιαίτερες διαφορές με την περίπτωση των περισσότερων αισθητήρων.

Σε αυτό το σημείο χρειάζεται να τονίσουμε πως ο τρόπος που υλοποιήσαμε την εφαρμογή μας δεν σχετίζεται με τα προβλήματα που αφορούν την ανίχνευση BLE συσκευών ούτε το χειρισμό της της συσκευής οδήγησης του Bluetooth πομποδέκτη. Για τις ενέργειες αυτές χρησιμοποιήθηκε έτοιμο API της Estimote (βλ. § 2.9). Στον κώδικα του API δεν είχαμε πρόσβαση καθώς ήταν ήδη σε μορφή εκτελέσιμου αρχείου Java Bytecode. Επίσης σε μία εφαρμογή επίδειξης (demo) του κατασκευαστή που συνόδευε το API υπήρχαν επίσης τα ίδια προβλήματα με πολύ αργούς χρόνους σάρωσης και ανακριβείς μετρήσεις.

Με τα μέσα που διαθέταμε δεν μπορέσαμε να ερμηνεύσουμε την αιτία της μεγάλης αυτής καθυστέρησης που παρατηρήθηκε. Πιθανόν να έφταιγε κάτι στον κώδικα του API που αφορούσε τη συχνότητα σάρωσης, πιθανόν όμως και να οφείλεται στη διαδικασία επίτευξης σύνδεσης μέσω του BLE πρωτοκόλλου. Δοκιμές επίσης, στα πρώτα στάδια ανάπτυξης της εφαρμογής, είχαν γίνει και με ένα Generic API[54] για BLE συσκευές, χωρίς ωστόσο να διορθώνονται τα παραπάνω προβλήματα, ενώ επιπλέον είχε και περιορισμένη συμβατότητα με τους αισθητήρες μας, πράγμα που συνέβαλε στην μη υιοθέτησή του για την τελική μας υλοποίηση. Τέλος δεν παραβλάψαμε τις δοκιμές της εφαρμογής πλοήγησης και από διαφορετικές κινητές συσκευές, με παρόμοια ωστόσο και πάλι αποτελέσματα.

[54] https://evothings.com/doc/tutorials/evothings-ble-api-guide.html

Κεφάλαιο 5 – Συμπεράσματα και Προτάσεις Βελτίωσης

5.1 Συμπεράσματα

Στόχος της υλοποίησής μας ήταν να εκμεταλλευτούμε τα οφέλη του Υπολογιστικού Νέφους και του Διαδικτύου των Πραγμάτων για τη δημιουργία διαδραστικών διαδικτυακών εφαρμογών. Μετά την περάτωση της εργασίας μας, τα συμπεράσματα, στα οποία καταλήξαμε είναι τα εξής:

- Η ανάπτυξη εφαρμογών στο Υπολογιστικό Νέφος προσφέρει πολλά πλεονεκτήματα, λόγω της ευρείας γκάμας υπηρεσιών που οι διάφοροι πάρχοι διαθέτουν έτοιμες για χρήση. Για παράδειγμα, η Υπηρεσία Διαχείρισης Συμβάντων και Συνδρομών (Orion Context Broker), αλλά και η Υπηρεσία Ταυτοποίησης και Εξουσιοδότησης (Keyrock Identity Manager) του FIWARE, είναι κλασικές υπηρεσίες απαραίτητες για την ανάπτυξη οποιασδήποτε διαδικτυακής εφαρμογής, ενώ φυσικά αντίστοιχες υπηρεσίες διατίθενται από οποιονδήποτε πάροχο. Η χρήση Υπηρεσιοκεντρικής Αρχιτεκτονικής και των υπηρεσιών ιστού RESTful διευκολύνουν ιδιαίτερα την επικοινωνία μεταξύ των υπηρεσιών, ακόμα και αν αυτές στεγάζονται σε διαφορετικά ΥΝ περιβάλλοντα. Παράλληλα με τη χρήση τεχνολογιών εικονικοποίησης από τους παρόχους, μπορούμε άμεσα να ξεκινήσουμε την ανάπτυξη της εφαρμογής μας, αποφεύγοντας τυχόν προβλήματα ασυμβατότητας σε επίπεδο αρχιτεκτονικής υλικού και φυσικών πόρων.

- Η χρήση των BLE αισθητήρων (beacons) της Estimote για τον προσδιορισμό θέσης, δεν αποδείχτηκε τελικά καλή επιλογή για την εφαρμογή μας. Μια εφαρμογή πλοήγησης σχεδιάζεται πάντοτε με στόχο την άμεση απόκριση για την παροχή οδηγιών σε πραγματικό χρόνο. Τα BLE Beacons ωστόσο που είχαμε στη διάθεσή μας υπονομεύονταν από υπερβολικά αργούς χρόνους σάρωσής (~5 δευτερόλεπτα ανά κύκλο σάρωσής), ενώ η καθυστέρηση κάποιες φορές γινόταν ακόμα μεγαλύτερη στην περίπτωση που υπήρχαν πολλά τέτοια beacons στην εμβέλεια. Τα beacons ήταν εύκολα ευάλωτα σε παρεμβολές από γειτονικές BLE συσκευές, επηρεάζοντας έτσι την ποιότητα σήματος τους. Ως αποτέλεσμα υπήρχαν φορές που παρόλο που ένας αισθητήρας ήταν τοποθετημένος κοντά, αυτός εμφανιζόταν ψευδώς σε άλλη πιο μακρινή θέση, ενώ αρκετός χρόνος απαιτούταν και για να σταθεροποιηθεί η ισχύς σήματος του, προκειμένου να επαναπροσδιοριστεί σωστά η θέση του.

5.2 Μελλοντικές Επεκτάσεις

Ο τρόπος υλοποίησης που ακολουθήσαμε, μπορούμε να θεωρήσουμε ότι εν τέλει ικανοποίησε τις αρχικές μας επιδιώξεις, επιτυγχάνοντας την δημιουργία ενός πλήρως

διαδραστικού συστήματος πλοήγησης, με το σύνολο των λειτουργιών του να φιλοξενούνται στο Υπολογιστικό Νέφος. Όπως ήταν ωστόσο αναπόφευκτο, η εφαρμογή μας έχει και κάποιες αδυναμίες τις οποίες και θα περιγράψουμε στη συνέχεια, προτείνοντας συγχρόνως πιθανές λύσεις και βελτιώσεις.

Η πρώτη και σπουδαιότερη αδυναμία της εφαρμογής, είναι αυτή που περιγράψαμε και στην προηγούμενη ενότητα και αφορά τους αισθητήρες που χρησιμοποιήθηκαν. Οι χρόνοι σάρωσης των BLE Estimote Proximity Beacons[55] ήταν υπερβολικά μεγάλοι, καθιστώντας τους συγκεκριμένους αισθητήρες μη ιδανικούς για εφαρμογές πλοήγησης. Δυστυχώς δεν μας δόθηκε η ευκαιρία να δοκιμάσουμε άλλα είδη αισθητήρων από τον ίδιο ή άλλους κατασκευαστές. Δεν μπορούμε λοιπόν να είμαστε σε θέση να γνωρίζουμε εάν οι αργοί χρόνοι σάρωσης οφείλονταν στα τεχνικά χαρακτηριστικά του συγκεκριμένου μοντέλου αισθητήρων, ή εάν έχει να κάνει με κάποιου είδους αδυναμία στο BLE πρωτόκολλο γενικότερα και στον τρόπο που αυτό συνδέεται με άλλες συσκευές. Ενδιαφέρον θα παρουσίαζε η μελέτη στην ταχύτητα σάρωσης διαφορετικών μοντέλων αισθητήρων αλλά και διαφορετικών πρωτοκόλλων συνδεσιμότητάς (π.χ. ZigBee, RFID).

Μία άλλη αδυναμία αφορά τον υπολογισμό των διαδρομών. Για την πλοήγηση αρκεστήκαμε να υπολογίζουμε οδηγίες βάσει κάποιων προκαθορισμένων μονοπατιών. Για το σενάριο χρήσης βέβαια που είχαμε αναφέρει ήδη από την εισαγωγή το παραπάνω είναι αρκετό. Ο διαχειριστής της υποδομής θέλει να έχει τον πλήρη έλεγχο, επομένως μόνο συγκεκριμένες διαδρομές θα επιτρέπονται που μπορεί να μην είναι πάντοτε και οι συντομότερες. Παρόλα αυτά, ως μελλοντική επέκταση συνίσταται η υιοθέτηση ενός αλγορίθμου εύρεσης συντομότερων διαδρομών από κοινή αφετηρία (single-source shortest path problem), με καταλληλότερο φυσικά τον αλγόριθμο του Dijkstra. Η αναπαράσταση ωστόσο των δεδομένων χρειάζεται και αυτή αλλαγές, καθώς θα πρέπει πλέον να λαμβάνονται υπόψιν και οι αποστάσεις μεταξύ των χώρων, μετατρέποντάς τες στα αντίστοιχα βάρη των ακμών του κατευθυνόμενου γράφου, πληροφορία η οποία δεν περιέχεται στην δική μας υλοποίηση.

Τέλος, μια ακόμα βελτίωση που προτείνεται είναι η δημιουργία ενός καλύτερου γραφικού περιβάλλοντος χρήσης στις εφαρμογές πλοήγησης αλλά και διαχείρισης. Η εφαρμογή μας περιορίζεται στην προβολή κυρίως λεκτικών οδηγιών. Θα ήταν σαφώς ανώτερη η εμπειρία χρήσης, αν υπήρχαν τρισδιάστατοι χάρτες με λειτουργίες αφής, όπου μάλιστα θα εμφανίζονταν σε πραγματικό χρόνο οι θέσεις όλων των επισκεπτών και άλλα ενδιαφέροντα σημεία στο χώρο, με τρόπο ανάλογο με αυτό των εμπορικών εφαρμογών.

[55] http://estimote.com/?gclid=EAIaIQobChMIu6ii29br1AIVIrXtCh246QYJEAAYASAAEgJ6hvD_BwE#get-beacons

Βιβλιογραφία

[1] Android Developers

https://developer.android.com/index.html

[2] OpenStack, “Open source software for creating private and public clouds”

https://www.openstack.org/

[3] FIWARE lab, “The open innovation Lab”

https://www.fiware.org/lab/

[4] “What is cloud computing? A beginner’s guide” | Microsoft Azure

https://azure.microsoft.com/en-us/overview/what-is-cloud-computing/

“Cloud computing” - Wikipedia

https://en.wikipedia.org/wiki/Cloud_computing

“A brief history of cloud computing” - Cloud computing news

https://www.ibm.com/blogs/cloud-computing/2014/03/a-brief-history-of-cloud-computing-3/

[5] Donald Bren School of Information and Computer Sciences

https://www.ics.uci.edu/

[6] FIWARE

https://www.fiware.org/

[7] Generic Enablers | FIWARE Catalogue

https://catalogue.fiware.org/enablers/

[8] OAuth 2.0

https://oauth.net/2/

[9] RESTful Web services: The basics

http://www.ibm.com/developerworks/library/ws-restful/

[10] IETF-related tools, standalone or hosted on tools.ietf.org

https://tools.ietf.org/

[11] “Internet of Things” - Springer

http://link.springer.com/chapter/10.1007/978-1-4419-8237-7_13#page-1

[12] “Unified Modeling Language” | UML

http://www.uml.org/

[13] W3C

http://www.w3.org/

[14] w3schools.com – “THE WORLD'S LARGEST WEB DEVELOPER SITE”

https://www.w3schools.com/

[15] “Introducing JSON”

http://www.json.org/

[16] National Institute of Standards and Technology

https://www.nist.gov/

[17] TechTarget – “Foundations for Growth”

http://www.techtarget.com/

[18] Apache Cordova

https://cordova.apache.org/

[19] MySQL

https://www.mysql.com/

[20] Slim Framework

https://www.slimframework.com/

[22] Base64 - Wikipedia

https://en.wikipedia.org/wiki/Base64

[23] NoSQL - Wikipedia

https://en.wikipedia.org/wiki/NoSQL

[24] Service-Oriented Architecture

http://www.service-architecture.com/

[25] National Science Foundation – “NSF FUTURE INTERNET ARCHITECTURE PROJECT”

http://www.nets-fia.net/

[26] P2P Foundation Wiki

http://wiki.p2pfoundation.net/

[27] Near Field Communication: “What is Near Field Communication?”

http://nearfieldcommunication.org/

[28] “develop with blue”

https://www.bluetooth.com/

[29] GitHub | “Example application: Beacon Finder”

https://github.com/evothings/phonegap-estimotebeacons/tree/master/examples/beacon-finder

[29] jQuery Mobile

http://demos.jquerymobile.com/1.4.0/

[30] “NGSI-9/NGSI-10 information model”

https://forge.fiware.org/plugins/mediawiki/wiki/fiware/index.php/NGSI-9/NGSI-10_information_model

[31] “How GAP and GATT Work”

https://punchthrough.com/bean/docs/guides/everything-else/how-gap-and-gatt-work/

[32] mapspeople – “Indoor Navigation Built with Google Maps”

https://www.mapspeople.com/

[33] infsoft – “Indoor Navigation, Indoor Positioning, Indoor Analytics and Indoor Tracking”

https://www.infsoft.com/

[34] IndoorAtlas – “Making indoor worlds discoverable”

http://www.indooratlas.com/

[35] mongoDB – “NoSQL Databases Explained”

http://www.indooratlas.com/

Βιβλιογραφία Εικόνων

[1] [Εικόνα 5] “Δομή μηνυμάτων στο SOAP”

http://flylib.com/books/2/439/1/html/2/images/f02mp01.jpg

[2] [Εικόνα 6] “Μοντέλα παροχής υπηρεσιών υπολογιστικού νέφους”

http://www.silverlighthack.com/image.axd?picture=2011%2F2%2FCloudServiceHierarchy.png

[3] [Εικόνα 7] “Απεικόνιση λειτουργικού συστήματος σε φυσικό και σε ιδεατό εξυπηρετητή”

https://ny5czrad2n-flywheel.netdna-ssl.com/wp-content/uploads/2016/08/VMware-Architecture-1.jpg

[4] [Εικόνα 8] “Υπερεπόπτης Τύπου 1”

http://4.bp.blogspot.com/-kUftu3Cj6xQ/U7EmHwRPPEI/AAAAAAAADpo/uKCf-K8MjiI/s1600/Type+1.png

[5] [Εικόνα 9] “Υπερεπόπτης Τύπου 2”

https://devopsunleashed.com/2017/04/11/hypervisor-type-1-type-2/

[6] [Εικόνα 37] “Διαδικασία σύνδεσης μέσω του πρωτοκόλλου OAuth2”

https://fhirblog.files.wordpress.com/2014/06/oauth2sequencediagram.png