



Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης  
Πολυτεχνική Σχολή  
Τμήμα Ηλεκτρολόγων Μηχανικών &  
Μηχανικών Υπολογιστών  
Τομέας Ηλεκτρονικής και Υπολογιστών  
Εργαστήριο Επεξεργασίας Πληροφορίας και Υπολογισμών

## Διπλωματική Εργασία

---

Μεθοδολογία ανάπτυξης γραφικών εφαρμογών  
για απομακρυσμένα ρομπότ, στο πλαίσιο  
κυβερνοφυσικών συστημάτων

---

*Εκπόνηση:*  
Γιόκοτος Κωνσταντίνος  
ΑΕΜ: 8791

*Επίβλεψη:*  
Ανδρέας Λ. Συμεωνίδης, Αναπληρωτής Καθηγητής  
Εμμανουήλ Τσαρδούλιας, Μεταδιδακτορικός Ερευνητής  
Κωνσταντίνος Παναγιώτου, Υποψήφιος Διδάκτορας

Θεσσαλονίκη, Ιούνιος 2021



---

## ΕΥΧΑΡΙΣΤΙΕΣ

---

Θα ήθελα να ευχαριστήσω θερμά τον Αναπληρωτή Καθηγητή κ. Ανδρέα Συμεωνίδη για την εμπιστοσύνη που μου έδειξε με την ανάθεση αυτής της διπλωματικής εργασίας, αλλά και για τη βοήθεια που μου παρείχε τις στιγμές που τη χρειάστηκα.

Θα ήθελα επίσης να ευχαριστήσω τον συνεπιβλέποντα της διπλωματικής εργασίας, Μεταδιδακτορικό Ερευνητή του τμήματος, Δρ. Εμμανουήλ Τσαρδούλια, καθώς και τον Γ.π. Δρ. Κωνσταντίνο Παναγιώτου για την καθοδήγησή, τις ιδέες, τις διορθώσεις και την υπομονή τους καθ' όλη τη διάρκεια εκπόνησης της διπλωματικής εργασίας.

Το μεγαλύτερο ευχαριστώ το οφείλω στην οικογένεια μου για την στήριξη, τις θυσίες και την αγάπη τους όλα αυτά τα χρόνια, δίνοντας μου κουράγιο να κάνω τα όνειρά μου πραγματικότητα. Τέλος, θα ήθελα να ευχαριστήσω τους φίλους μου για τη συμπαράστασή τους και τις όμορφες στιγμές που ζήσαμε καθ' όλη τη διάρκεια των σπουδών μας.



---

## Περίληψη

Όπως το Διαδίκτυο άλλαξε τον τρόπο που οι άνθρωποι αλληλεπιδρούν με την πληροφορία, έτσι και τα κυβερνοφυσικά συστήματα αλλάζουν τον τρόπο που οι άνθρωποι αλληλεπιδρούν με τα υπολογιστικά συστήματα. Τα κυβερνοφυσικά συστήματα ενσωματώνουν αισθητήρες, υπολογιστικές δυνατότητες, έλεγχο και δικτύωση σε φυσικά αντικείμενα, συνδέοντάς τα στο Διαδίκτυο, αλλά και μεταξύ τους.

Ενδεικτικό παράδειγμα τέτοιων συστημάτων είναι τα ρομποτικά συστήματα, καθώς συνδυάζουν αλληλεπίδραση με το περιβάλλον και υπολογιστικές ικανότητες. Παρόλο που η ρομποτική είναι παραδοσιακά συνυφασμένη με τη βιομηχανία, τα τελευταία χρόνια έχει επεκταθεί και σε άλλους κλάδους, όπως στην ιατρική, στην αυτόνομη εξερεύνηση αλλά και σε τομείς της καθημερινής ζωής, όπως για οικιακή χρήση και ψυχαγωγία.

Παράλληλα, ραγδαία αύξηση παρουσιάζει και το Διαδίκτυο των Πραγμάτων (Internet of Things – IoT), όπου πλέον αντικείμενα της καθημερινότητας είναι εξοπλισμένα με αισθητήρες για τη συλλογή δεδομένων από το περιβάλλον και συνδέονται στο Διαδίκτυο για να μοιραστούν αυτά τα δεδομένα. Λόγω της κινητικότητας που προσφέρουν τα ρομποτικά συστήματα, η ενσωμάτωση τους στον IoT κόσμο θα επιτρέπει την καλύτερη επίδραση στο περιβάλλον, ενώ παράλληλα τα ρομπότ θα λαμβάνουν αποφάσεις βάσει δεδομένων άλλων συσκευών.

Για να καταστεί αυτό δυνατό, πρέπει να ξεπεραστούν ορισμένοι περιορισμοί. Αφ' ενός, είναι ιδιαίτερα σημαντικό να υπάρχει η δυνατότητα του απομακρυσμένου ελέγχου και παρακολούθησης ενός ρομπότ. Δυστυχώς, το Robot Operating System (ROS), το πιο διαδεδομένο μεσολειτουργικό σύστημα για ανάπτυξη ρομποτικών εφαρμογών, περιορίζει τη διαχείριση του ρομπότ σε τοπικό δίκτυο. Παράλληλα, είναι επιθυμητό, χρήστες χωρίς ιδιαίτερες γνώσεις ρομποτικής και προγραμματισμού να έχουν τη δυνατότητα να δημιουργήσουν τις εφαρμογές τους.

Η παρούσα διπλωματική εργασία εστιάζει στην ανάπτυξη ενός συστήματος που θα αντιμετωπίζει τους παραπάνω περιορισμούς. Για την επικοινωνία μεταξύ του ρομπότ και του απομακρυσμένου υπολογιστή, χρησιμοποιείται ο μεσολαβητής μηνυμάτων RabbitMQ. Ταυτόχρονα, η ανάπτυξη των εφαρμογών και η ενσωμάτωση του ρομπότ στον IoT κόσμο πραγματοποιείται μέσω του Node-RED, ενός εργαλείου που επιτρέπει τη δημιουργία εφαρμογών για IoT συστήματα μέσω γραφικής διεπαφής, γεγονός που απλοποιεί σημαντικά τη δυσκολία της διαδικασίας του προγραμματισμού. Επιπλέον, υλοποιήθηκαν διάφορα σενάρια χρήσης που αναδεικνύουν τις δυνατότητες του συστήματος για την ανάπτυξη ρομποτικών εφαρμογών στα πλαίσια του IoT.

**Λέξεις κλειδιά:** Κυβερνοφυσικά συστήματα, ρομποτική, Internet of Things, απομακρυσμένος έλεγχος ρομπότ, ROS, RabbitMQ, Node-RED.

Γιόκοτος Κωνσταντίνος  
Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών,  
Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης, Ελλάδα  
Ιούνιος 2021



---

# A graphical application development methodology for remote robots in the context of cyber-physical systems

## Abstract

Just as the Internet has transformed the way people interact with information, cyber-physical systems are transforming the way people interact with computational systems. Cyber-physical systems integrate sensing, computation, control and networking into physical objects, connecting them to the Internet and to each other.

A typical example of such systems are robotic systems, as they combine interaction with the environment and computational abilities. Even though robotics is closely tied to the manufacturing industry, in recent years it has branched out to other fields, such as medicine and autonomous exploration, and even in aspects of our daily life, such as for domestic use.

A growth of similar scale can be seen in the Internet of Things (IoT) domain, where everyday objects are equipped with sensors to collect data from the environment and are able to connect to the Internet to share this data. We envision that, due to the mobility offered by robotic systems, their integration with IoT would enable better interaction with the environment, and simultaneously allow robots to make decisions based on data from other devices.

To make this possible, there are certain limitations that must be overcome. On one hand, it is especially important to have the ability to control and monitor the robot remotely. Unfortunately, the Robot Operating System (ROS), the most widespread middleware for robotics development, restricts the management of the robot to the local network. On the other hand, it is desirable for users to have the ability to create their applications without having extensive robotics and programming knowledge.

This thesis focuses on developing a system to address the aforementioned limitations. To establish the communication between the robot and the remote computer, the RabbitMQ message broker is used. At the same time, application development and the integration of the robot with the IoT world are accomplished through Node-RED, a tool for building applications for IoT systems through a graphical interface, thus simplifying the programming procedure. Furthermore, various use cases are presented, which showcase the capabilities of the system for developing robotic applications as part of the IoT.

**Keywords:** Cyber-physical systems, robotics, Internet of Things, remote robot control, ROS, RabbitMQ, Node-RED.

Giokotos Konstantinos  
kostasgioko8@gmail.com  
Electrical & Computer Engineering Department,  
Aristotle University of Thessaloniki, Greece  
June 2021

# Περιεχόμενα

Ευχαριστίες .....	i
Περίληψη .....	iii
Abstract .....	v
Λίστα Εικόνων .....	viii
<b>1. ΕΙΣΑΓΩΓΗ.....</b>	<b>1</b>
1.1 Κίνητρο.....	1
1.2 Περιγραφή του προβλήματος.....	2
1.3 Στόχοι της διπλωματικής.....	3
1.4 Μεθοδολογία .....	3
1.5 Διάρθρωση.....	4
<b>2. ΘΕΩΡΗΤΙΚΟ ΥΠΟΒΑΘΡΟ .....</b>	<b>5</b>
2.1 Διαδίκτυο των πραγμάτων .....	5
2.1.1 Ορισμός.....	6
2.1.2 Ιστορική Εξέλιξη .....	6
2.1.3 Χαρακτηριστικά .....	7
2.1.4 Αρχιτεκτονική.....	7
2.1.5 Πρωτόκολλα Επικοινωνίας Δεδομένων.....	8
2.2 Ρομποτική.....	10
2.2.1 Δίκτυα και ρομποτική .....	10
2.2.2 Internet of Robotic Things .....	11
2.3 Γραφικός Προγραμματισμός.....	13
2.3.1 Model Driven Engineering.....	13
<b>3. ΣΧΕΤΙΚΗ ΒΙΒΛΙΟΓΡΑΦΙΑ .....</b>	<b>16</b>
3.1 IoT και Ρομποτική.....	16
3.1.1 Ιατρική – Βιοϊατρική Τεχνολογία .....	16
3.1.2 Ευφυή Συστήματα Μεταφορών.....	18
3.2 Απομακρυσμένος έλεγχος ρομπότ μέσω ROS.....	19
3.3 Γραφικά περιβάλλοντα ανάπτυξης ρομποτικών εφαρμογών .....	20
<b>4. ΤΕΧΝΟΛΟΓΙΚΟ ΥΠΟΒΑΘΡΟ .....</b>	<b>22</b>
4.1 TurtleBot .....	22
4.2 Robot Operating System .....	23
4.3 Βιβλιοθήκες ρομποτικής .....	25

## ΠΕΡΙΕΧΟΜΕΝΑ

---

4.3.1	GMapping.....	25
4.3.2	AMCL.....	25
4.3.3	ROS Navigation Stack .....	26
4.4	Gazebo .....	27
4.5	RabbitMQ.....	27
4.6	Node-RED .....	29
5.	ΥΛΟΠΟΙΗΣΗ.....	30
5.1	Γενική περιγραφή του συστήματος.....	30
5.1.1	Προδιαγραφές.....	30
5.1.2	Αρχιτεκτονική.....	31
5.2	Τοπικό τμήμα του συστήματος.....	32
5.2.1	Πακέτα του ROS.....	32
5.2.2	Διαχείριση των λειτουργιών .....	37
5.2.3	Επικοινωνία με το RabbitMQ .....	38
5.3	Μεσολαβητής μηνυμάτων .....	40
5.4	Απομακρυσμένο τμήμα του συστήματος .....	42
6.	ΣΕΝΑΡΙΑ ΧΡΗΣΗΣ.....	48
6.1	Coffee Shop.....	48
6.2	House Security .....	51
6.3	Art Gallery .....	54
7.	ΣΥΜΠΕΡΑΣΜΑΤΑ & ΜΕΛΛΟΝΤΙΚΗ ΕΡΓΑΣΙΑ .....	58
7.1	Συμπεράσματα .....	58
7.2	Προβλήματα.....	59
7.3	Μελλοντικές Επεκτάσεις .....	60
	Βιβλιογραφία .....	61

# Λίστα Εικόνων

Εικόνα 1.1: Κέρδη ρομπότ υπηρεσιών από επαγγελματική χρήση.....	1
Εικόνα 1.2: Κέρδη ρομπότ υπηρεσιών από οικιακή/προσωπική χρήση.....	2
Εικόνα 2.1: Μέγεθος αγοράς IoT .....	5
Εικόνα 2.2: Εξέλιξη του IoT [4].....	6
Εικόνα 2.3: Επίπεδα αρχιτεκτονικής IoT [6] .....	8
Εικόνα 2.4: Παράδειγμα ανταλλαγής μηνυμάτων με MQTT [11].....	9
Εικόνα 2.5: Παράδειγμα ανταλλαγής μηνυμάτων με AMQP .....	10
Εικόνα 2.6: Επίπεδα αρχιτεκτονικής IoRT [3].....	11
Εικόνα 2.7: Model Driven Architecture [14] .....	14
Εικόνα 2.8: Μετασχηματισμός μοντέλου [15] .....	15
Εικόνα 3.1: Σύστημα ανάλυσης σημάτων εγκεφάλου [24] .....	17
Εικόνα 3.2: Αναπαράσταση ενός ευφυούς συστήματος μεταφορών [28] .....	18
Εικόνα 3.3: Δομή του ROS Edge Node [35] .....	20
Εικόνα 3.4: Το γραφικό περιβάλλον Robokol [37] .....	21
Εικόνα 4.1: Το TurtleBot2.....	22
Εικόνα 4.2: Επικοινωνία δύο κόμβων στο ROS.....	24
Εικόνα 4.3: Γενική δομή του navigation stack.....	26
Εικόνα 4.4: Περιβάλλον διαχείρισης του RabbitMQ .....	28
Εικόνα 4.5: Γραφικό περιβάλλον ανάπτυξης του Node-RED .....	29
Εικόνα 5.1: Η αρχιτεκτονική του συστήματος.....	31
Εικόνα 5.2: Χάρτης από την εκτέλεση του SLAM .....	33
Εικόνα 5.3: Διάγραμμα μηνυμάτων για το SLAM .....	33
Εικόνα 5.4: Το φίλτρο σωματιδίων κατά τον εντοπισμό θέσης.....	34
Εικόνα 5.5: Διάγραμμα μηνυμάτων για τον εντοπισμό θέσης.....	35
Εικόνα 5.6: Οι χάρτες κόστους και τα μονοπάτια κατά την πλοιήγηση .....	35
Εικόνα 5.7: Διάγραμμα μηνυμάτων για την πλοιήγηση.....	36
Εικόνα 5.8: Διάγραμμα μηνυμάτων για την πλοιήγηση πολλαπλών ρομπότ.....	37
Εικόνα 5.9: Διαχείριση εντολών χρήστη από το handler .....	38
Εικόνα 5.10: Αντιστοίχιση ROS topics σε ουρές AMQP για λειτουργία ενός ρομπότ .....	39
Εικόνα 5.11: Αντιστοίχιση ROS topics σε ουρές AMQP για λειτουργία πολλαπλών ρομπότ .....	40
Εικόνα 5.12: Μεταφορά μηνυμάτων από το RabbitMQ για λειτουργία ενός ρομπότ .....	42
Εικόνα 5.13: Μεταφορά μηνυμάτων από το RabbitMQ για λειτουργία πολλαπλών ρομπότ .....	42
Εικόνα 5.14: Η χαρτέλα Home του Dashboard.....	43
Εικόνα 5.15: Η χαρτέλα Launchers του Dashboard .....	44
Εικόνα 5.16: Η χαρτέλα Robot Control του Dashboard.....	44
Εικόνα 5.17: Το Node-RED flow για τον τηλεχειρισμό .....	45

## ΛΙΣΤΑ ΕΙΚΟΝΩΝ

---

Εικόνα 5.18: Το Node-RED flow για την πλοήγηση.....	45
Εικόνα 5.19: Λειτουργία SLAM μέσω Dashboard.....	46
Εικόνα 5.20: Λειτουργία πλοήγησης για δύο ρομπότ μέσω Dashboard.....	47
Εικόνα 6.1: Η καφετέρια στο περιβάλλον του Gazebo .....	48
Εικόνα 6.2: Σενάριο χρήσης Coffee Shop .....	49
Εικόνα 6.3: Διάγραμμα ροής της εφαρμογής Coffee Shop .....	50
Εικόνα 6.4: Το Node-RED flow της εφαρμογής Coffee Shop.....	50
Εικόνα 6.5: Το σπίτι στο περιβάλλον του Gazebo.....	51
Εικόνα 6.6: Σενάριο χρήσης House Security.....	51
Εικόνα 6.7: Διάγραμμα ροής της εφαρμογής House Security .....	52
Εικόνα 6.8: Το Node-RED flow της εφαρμογής House Security (1/2) .....	53
Εικόνα 6.9: Το Node-RED flow της εφαρμογής House Security (2/2) .....	53
Εικόνα 6.10: Η γκαλερί στο περιβάλλον του Gazebo .....	54
Εικόνα 6.11: Σενάριο χρήσης Art Gallery .....	54
Εικόνα 6.12: Διάγραμμα ροής της εφαρμογής Art Gallery .....	55
Εικόνα 6.13: Το Node-RED flow της εφαρμογής Art Gallery (1/2) .....	56
Εικόνα 6.14: Το Node-RED flow της εφαρμογής Art Gallery (2/2) .....	56



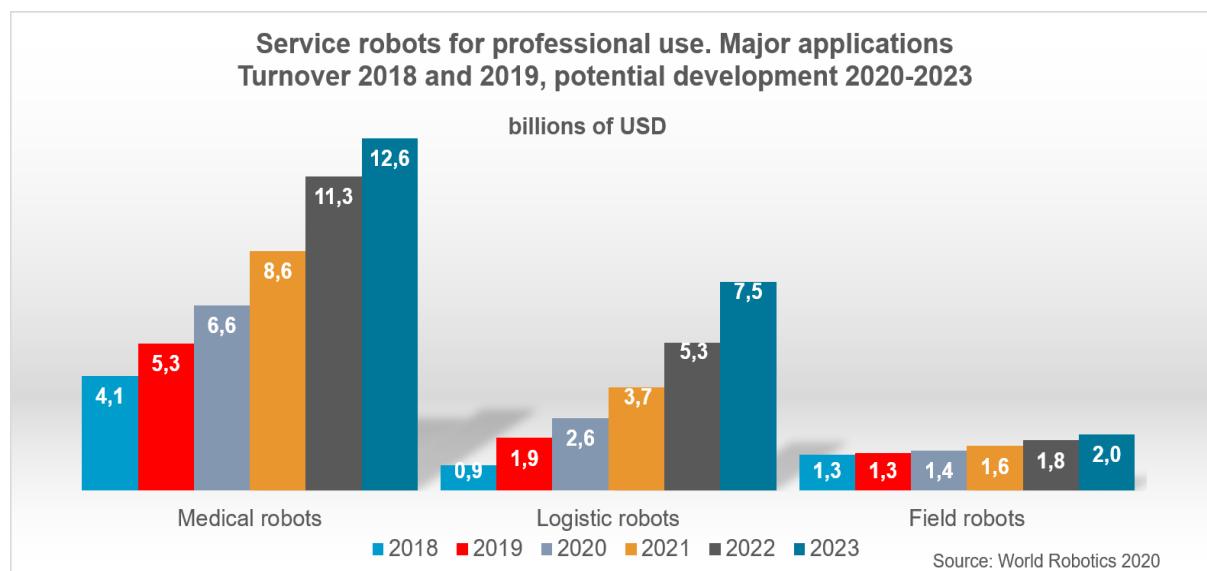
# Κεφάλαιο 1

## ΕΙΣΑΓΩΓΗ

Το παρόν κεφάλαιο αποτελεί τον πρόλογο της διπλωματικής εργασίας, όπου γίνεται μια προσπάθεια να γίνει ξεκάθαρο το πρόβλημα που επιχειρούμε να αντιμετωπίσουμε, αλλά και ο τρόπος με τον οποίο συνεισφέρει η διπλωματική εργασία στην επίλυσή του. Στην Ενότητα 1.1 αναλύονται τα κίνητρα που οδήγησαν στην εκπόνηση της εργασίας αυτής. Η περιγραφή του προβλήματος γίνεται στην Ενότητα 1.2. Στις Ενότητες 1.3 και 1.4 παρουσιάζονται οι στόχοι της διπλωματικής εργασίας και η μεθοδολογία που ακολουθήθηκε αντίστοιχα. Τέλος, στην Ενότητα 1.5 παρέχεται μια σύνοψη της διάρθρωσης της εργασίας.

### 1.1 Κίνητρο

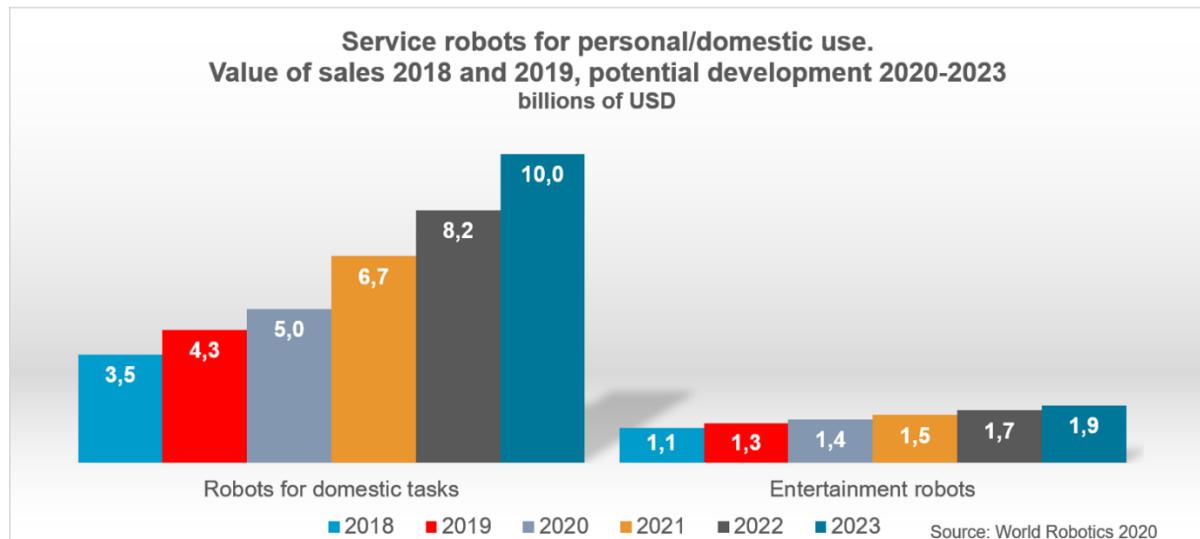
Στη σύγχρονη εποχή, τα κυβερνοφυσικά συστήματα, δηλαδή συστήματα που συνδυάζουν υπολογιστικές δυνατότητες και αλληλεπίδραση με το περιβάλλον και τον άνθρωπο, εμφανίζονται όλο και περισσότερο, και με συνεχώς διαφορετικές μορφές. Χαρακτηριστικό παράδειγμα τέτοιων συστημάτων είναι τα ρομποτικά συστήματα. Οι εφαρμογές τους είναι πολυάριθμες και κυμαίνονται από βραχίονες που χρησιμοποιούνται στη βιομηχανία, ρομπότ που εκτελούν χειρουργικές επεμβάσεις έως και ρομπότ που βοηθούν στις δουλειές του σπιτιού. Στα σχήματα 1.1 και 1.2<sup>1</sup> φαίνεται η ραγδαία αύξηση στη χρήση των ρομπότ σε σχεδόν όλους τους τομείς, η οποία εκτιμάται ότι θα συνεχίσει να αυξάνεται και τα επόμενα χρόνια.



Εικόνα 1.1: Κέρδη ρομπότ υπηρεσιών από επαγγελματική χρήση.

<sup>1</sup> <https://ifr.org/ifr-press-releases/news/service-robots-record-sales-worldwide-up-32>

## ΚΕΦΑΛΑΙΟ 1. ΕΙΣΑΓΩΓΗ



Εικόνα 1.2: Κέρδη ρομπότ υπηρεσιών από οικιακή/προσωπική χρήση.

Μια ακόμη τεχνολογία που έχει ραγδαία αύξηση τα τελευταία έτη είναι το Διαδίκτυο των Πραγμάτων (Internet of Things – IoT), όπου καθημερινά αντικείμενα όπως λάμπες, οικιακές συσκευές, ρολόγια κ.ά. διαθέτουν αισθητήρες, είναι συνδεδεμένα μεταξύ τους μέσω Διαδικτύου και μπορούν να ανταλλάξουν δεδομένα. Δεδομένου ότι ένα από τα πιο σημαντικά χαρακτηριστικά των ρομποτικών συστημάτων είναι η κίνηση στο χώρο, η ενσωμάτωσή τους στο IoT θα επιφέρει σημαντικά πλεονεκτήματα. Πέρα από τη λήψη δεδομένων του περιβάλλοντος από τις υπόλοιπες συσκευές, θα προστεθεί και η ικανότητα αλληλεπίδρασης με αυτό μέσω των ρομποτικών συσκευών, ενώ το ρομπότ θα μπορεί να λαμβάνει αποφάσεις και βάσει δεδομένων πέρα από αυτά που συλλέγει το ίδιο.

Για το λόγο αυτό, υπάρχει έντονο επιστημονικό ενδιαφέρον προς αυτήν την κατεύθυνση, με πολλές εφαρμογές να εκμεταλλεύονται το συνδυασμό των δύο τεχνολογιών και να παρουσιάζουν εντυπωσιακά αποτελέσματα.

### 1.2 Περιγραφή του προβλήματος

Για να πραγματοποιηθεί η ενσωμάτωση ενός ρομπότ σε ένα σύστημα IoT υπάρχουν κάποιοι περιορισμοί που πρέπει να αντιμετωπιστούν και είναι οι άξονες στους οποίους θα επικεντρωθεί η διπλωματική εργασία.

Το πιο διαδεδομένο μεσολειτουργικό σύστημα για ανάπτυξη ρομποτικών εφαρμογών είναι το Robot Operating System<sup>1</sup> (ROS). Παρόλο που το ROS δεν είναι λειτουργικό σύστημα, παρέχει παρόμοιες υπηρεσίες που διευκολύνουν πολύ τη διαδικασία προγραμματισμού ενός ρομπότ. Ένα σημαντικό μειονέκτημα του ROS, όμως, είναι ότι δεν υπάρχει κάποιος εύκολος τρόπος διαχείρισης του ρομπότ μέσω Διαδικτύου με χρήση πρωτοκόλλων υπηρεσιών δικτύου, όπως REST ή πρωτοκόλλων που χρησιμοποιούνται στο IoT. Επιπλέον, μια συσκευή που χρησιμοποιεί το ROS μπορεί να επικοινωνήσει μόνο με άλλες συσκευές που χρησιμοποιούν και αυτές το ROS. Για να ξεπεραστούν αυτοί οι περιορισμοί έχουν αναπτυχθεί διάφορα εργαλεία που επιτρέπουν απομακρυσμένο έλεγχο και επικοινωνία με ένα ρομπότ, τα οποία θα αναλυθούν στο Κεφάλαιο 3.

<sup>1</sup> <https://www.ros.org/>

2 Μεθοδολογία ανάπτυξης γραφικών εφαρμογών για απομακρυσμένα ρομπότ, στο πλαίσιο κυβερνοφυσικών συστημάτων

Γιόκοτος Κ.

Παρόλο που η ρομποτική εξελίσσεται όλο και περισσότερο, λίγοι μπορούν να κατάσκευάσουν και να προγραμματίσουν ένα ρομπότ, διότι πρόκειται για μια δύσκολη διαδικασία, που απαιτεί εξειδικευμένες γνώσεις. Για την ενσωμάτωση μιας ρομποτικής συσκευής στο IoT, είναι σημαντικό ο χρήστης να έχει τη δυνατότητα να αναπτύξει την εφαρμογή του, χωρίς να διαθέτει τεχνικές γνώσεις ρομποτικής ή προγραμματισμού. Προς αυτήν την κατεύθυνση, γίνονται προσπάθειες για την απλοποίηση της διαδικασίας προγραμματισμού ενός ρομπότ, όπως μέσω γραφικών διεπαφών, που αναλύονται στο Κεφάλαιο 3.

### 1.3 Στόχοι της διπλωματικής

Τα προβλήματα που περιγράφονται στην προηγούμενη ενότητα καλείται να αντιμετωπίσει η παρούσα διπλωματική εργασία.

Ο τελικός στόχος είναι η υλοποίηση ενός συστήματος, που θα δίνει στο χρήστη τη δυνατότητα να αναπτύξει την εφαρμογή του για ένα ρομπότ από έναν απομακρυσμένο υπολογιστή, χωρίς να έχει εξειδικευμένες γνώσεις. Αυτό μπορεί να αναλυθεί σε δύο υποστόχους. Αφ' ενός θα πρέπει να υπάρχει η δυνατότητα ελέγχου του ρομπότ μέσω Διαδικτύου, αφ' επέρου θα πρέπει να παρέχεται σημαντική διευκόλυνση στο χρήστη για την ανάπτυξη της ρομποτικής του εφαρμογής.

### 1.4 Μεθοδολογία

Το ρομπότ που θα χρησιμοποιηθεί στα πλαίσια της διπλωματικής εργασίας είναι το TurtleBot2<sup>1</sup>, το οποίο επικοινωνεί μέσω ROS. Για να επιτευχθεί η επικοινωνία μεταξύ του ROS και του απομακρυσμένου υπολογιστή, γίνεται χρήση του RabbitMQ<sup>2</sup>, ενός μεσολαβητή για την αποστολή και λήψη μηνυμάτων. Η ανάπτυξη των εφαρμογών γίνεται με τη βοήθεια του Node-RED<sup>3</sup>, ενός εργαλείου που επιτρέπει τη δημιουργία εφαρμογών για IoT συστήματα μέσω γραφικής διεπαφής.

Το RabbitMQ υποστηρίζει διάφορα πρωτόκολλα μηνυμάτων, όπως AMQP, MQTT, STOMP κ.ά. Επειδή τα δεδομένα του ROS είναι δομημένα με διαφορετικό τρόπο, το πρώτο βήμα είναι η μετατροπή των δεδομένων αυτών σε μηνύματα που μπορούν να μεταδοθούν μέσω του RabbitMQ. Εφόσον φτάσουν τα δεδομένα στο μεσολαβητή, μπορούν εύκολα να σταλθούν στο Node-RED, αφού και αυτό δέχεται μηνύματα MQTT.

Παράλληλα, με το Node-RED απλοποιείται σημαντικά και ο προγραμματισμός του ρομπότ, αφού ο χρήστης δε γράφει κώδικα για να σχεδιάσει την εφαρμογή, αλλά ενώνει κόμβους που επιτελούν βασικές λειτουργίες για να επιτύχει αυτό που θέλει. Το Node-RED δεν μπορεί να επικοινωνήσει απευθείας με το ROS, γιατί τα μηνύματα που χρησιμοποιεί το καθένα ακολουθούν άλλα πρωτόκολλα. Χρησιμοποιώντας όμως, ως μεσολαβητή το RabbitMQ, ξεπερνάμε αυτόν τον περιορισμό, κάνοντας δυνατή την επικοινωνία μεταξύ των δύο. Τέλος, επιτρέπει σε πολλές συσκευές IoT να επικοινωνούν εύκολα μεταξύ τους, και άρα με αυτόν τον τρόπο ενσωματώνεται και το ρομπότ στο IoT.

<sup>1</sup> <https://www.turtlebot.com/>

<sup>2</sup> <https://www.rabbitmq.com>

<sup>3</sup> <https://nodered.org>

## 1.5 Διάρθρωση

Η διάρθρωση της παρούσας διπλωματικής εργασίας είναι η εξής:

- **Κεφάλαιο 1:** Γίνεται μια περιγραφή του προβλήματος και της προτεινόμενης αντιμετώπισής του.
- **Κεφάλαιο 2:** Παρουσιάζεται το θεωρητικό υπόβαθρο των εννοιών που πραγματεύεται η διπλωματική εργασία, για την καλύτερη κατανόηση του κειμένου από τον αναγνώστη.
- **Κεφάλαιο 3:** Γίνεται ανασκόπηση της ερευνητικής περιοχής σχετικά με υπάρχοντα συστήματα που συνδυάζουν το IoT και τη ρομποτική, με τον απομακρυσμένο έλεγχο ρομπότ μέσω ROS, καθώς επίσης και με γραφικά περιβάλλοντα ανάπτυξης ρομποτικών εφαρμογών.
- **Κεφάλαιο 4:** Παρουσιάζεται το τεχνολογικό υπόβαθρο, δηλαδή τα εργαλεία λογισμικού που χρησιμοποιήθηκαν για την υλοποίηση του συστήματος.
- **Κεφάλαιο 5:** Αναλυτική περιγραφή της υλοποίησης του συστήματος της διπλωματικής εργασίας, και συγκεκριμένα των επιμέρους τμημάτων που το απαρτίζουν: α) τοπικό τμήμα που επικοινωνεί άμεσα με το ρομπότ, β) μεσολαβητής μηνυμάτων για την επικοινωνία μεταξύ του τοπικού και του απομακρυσμένου τμήματος και, γ) απομακρυσμένο τμήμα, όπου γίνεται η ανάπτυξη των εφαρμογών από το χρήστη.
- **Κεφάλαιο 6:** Γίνεται μία παρουσίαση των σεναρίων χρήσης του συστήματος, όπου υλοποιήθηκαν κάποιες εφαρμογές που αναδεικνύουν τα πλεονεκτήματα και τις δυνατότητες του συστήματος.
- **Κεφάλαιο 7:** Παρουσιάζονται τα συμπεράσματα που προέκυψαν κατά την ανάπτυξη του συστήματος, τα προβλήματα που εμφανίστηκαν κατά τη διάρκεια υλοποίησής του, καθώς επίσης και μερικές μελλοντικές επεκτάσεις για τη βελτίωσή του.

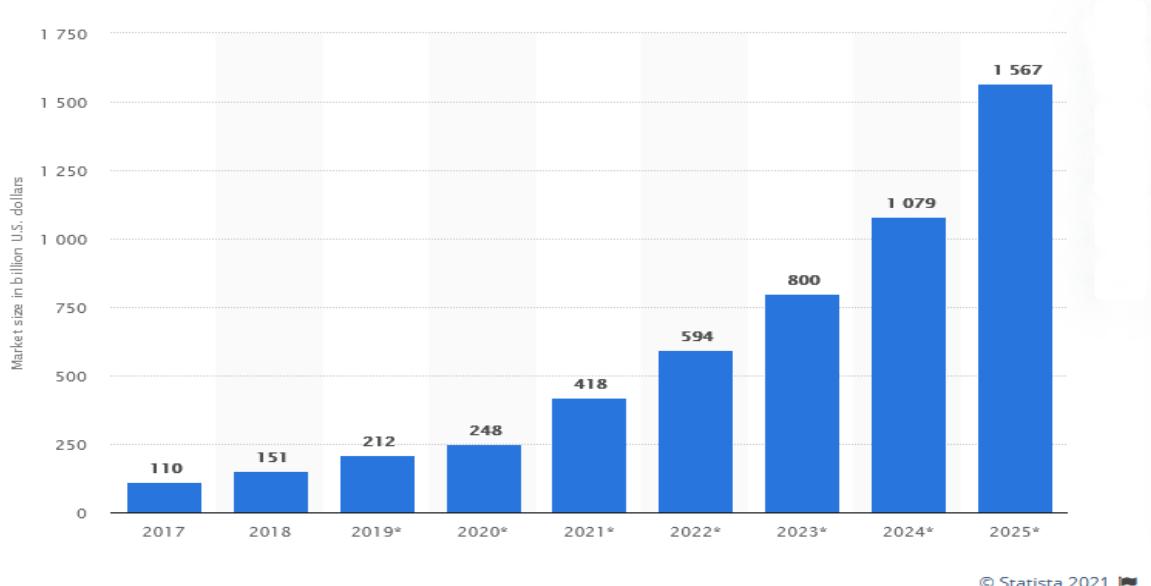
## Κεφάλαιο 2

# ΘΕΩΡΗΤΙΚΟ ΥΠΟΒΑΘΡΟ

Στο κεφάλαιο αυτό θα παρουσιαστεί το βασικό θεωρητικό υπόβαθρο που θα βοηθήσει τον αναγνώστη στην κατανόηση του κειμένου της παρούσας διπλωματικής εργασίας. Όπως περιγράφηκε και στο προηγούμενο κεφάλαιο, ο στόχος της διπλωματικής εργασίας είναι η υλοποίηση ενός συστήματος που θα επιτρέπει σε ένα ρομποτικό σύστημα να ενσωματωθεί στο IoT, ενώ παράλληλα θα διευκολύνει το χρήστη στην ανάπτυξη των εφαρμογών του. Στην Ενότητα 2.1 γίνεται μία περιγραφή του Διαδικτύου των Πραγμάτων και των πρωτοκόλλων επικοινωνίας δεδομένων που χρησιμοποιεί. Το πώς μπορεί η ρομποτική να εξελιχθεί και να ενσωματωθεί στο IoT αναλύεται στην Ενότητα 2.2. Τέλος, στην Ενότητα 2.3 παρουσιάζεται η μέθοδος του γραφικού προγραμματισμού για την απλοποίηση της διαδικασίας ανάπτυξης εφαρμογών.

### 2.1 Διαδίκτυο των Πραγμάτων

Το IoT έχει εισχωρήσει για τα καλά στην καθημερινή μας ζωή, είτε μέσω οικιακών συσκευών που συνδέονται στο Διαδίκτυο, είτε μέσω ‘έξυπνων’ ρολογιών και περικάρπιων που καταγράφουν πληροφορίες για τις δραστηριότητες μας, ή ακόμα και στις μεταφορές μέσω των αυτοκινήτων [1]. Τα τελευταία χρόνια, το IoT εξελίσσεται ραγδαία και η αύξηση αυτή προβλέπεται ότι θα συνεχιστεί και τα επόμενα χρόνια, όπως φαίνεται στην εικόνα 2.1<sup>1</sup>.



Εικόνα 2.1: Μέγεθος αγοράς IoT

<sup>1</sup> <https://www.statista.com/statistics/976313/global-iot-market-size/>

### 2.1.1 Ορισμός

Ορισμοί για το IoT έχουν δοθεί πολλοί και διαφορετικοί, διότι σε αντίθεση με άλλους τεχνικούς όρους, το IoT δεν είναι μια καινούρια ιδέα, αλλά μια νέα αναπαράσταση ενός μοντέλου που ενώνει ήδη υπάρχουσες τεχνολογίες. Το Ινστιτούτο Ηλεκτρολόγων και Ηλεκτρονικών Μηχανικών (Institute of Electrical and Electronics Engineers - IEEE) περιγράφει το IoT ως [2]:

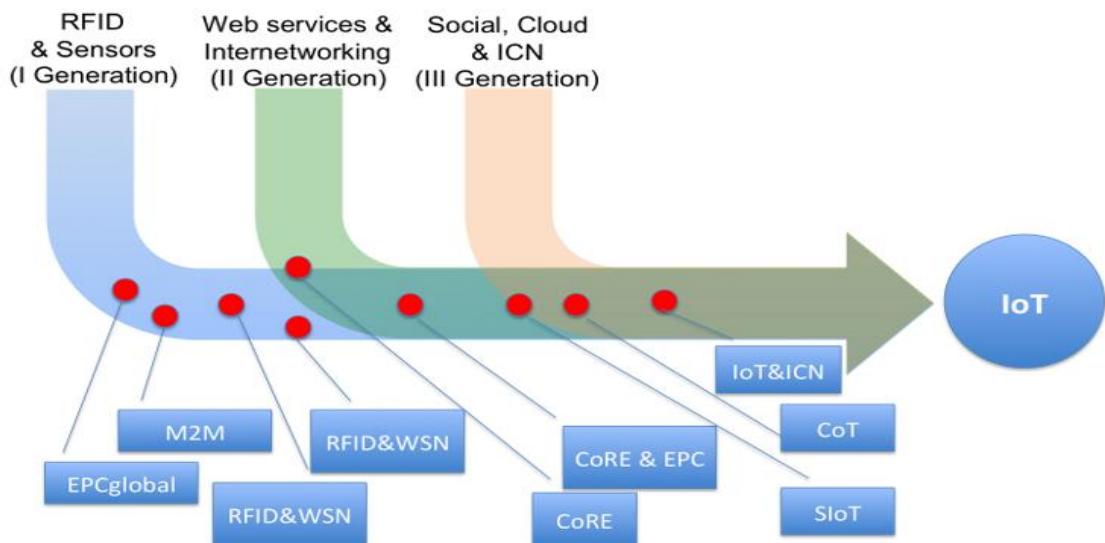
«Ενα δίκτυο αντικειμένων, το καθένα ενσωματωμένο με αισθητήρες, που είναι συνδεδεμένα στο Διαδίκτυο.»

Η παραπάνω δήλωση εστιάζει στη φυσική υπόσταση του IoT, καθώς πέρα από τους αισθητήρες, απαραίτητο είναι και το λογισμικό που καθιστά δυνατή τη σύνδεση στο Διαδίκτυο και την ανταλλαγή δεδομένων. Παρ' όλα αυτά είναι μια αρκετά απλή και περιεκτική περιγραφή του IoT.

### 2.1.2 Ιστορική Εξέλιξη

Η ιδέα ενός δικτύου από «έξυπνες» συσκευές πρωτοεμφανίστηκε το 1982 στο πανεπιστήμιο Carnegie Mellon, με τη σύνδεση ενός αυτόματου πωλητή στο Διαδίκτυο. Το 1991, το άρθρο του Mark Weiser 'The Computer of the 21<sup>st</sup> Century' αναφέρθηκε στη σύγχρονη μορφή του IoT. Ο όρος Internet of Things επινοήθηκε το 1999 από τον Kevin Ashton του Auto-ID Center, ο οποίος επισήμανε τη ζωτική σημασία του radio-frequency identification (RFID) για τη διαχείριση των συσκευών [3].

Στην εξέλιξη του IoT μπορούν να εντοπιστούν τρία κύρια στάδια ή γενιές, όπως φαίνεται στην εικόνα 2.2.



Εικόνα 2.2: Εξέλιξη του IoT [4]

Στην πρώτη γενιά, έμφαση δόθηκε στην ηλεκτρονική παρακολούθηση αντικειμένων μέσω RFID, Electronic Product Code (EPC) και Wireless Sensor Networks (WSN). Η δεύτερη γενιά είχε ως στόχο την απευθείας σύνδεση των αντικειμένων στο Διαδίκτυο με χρήση του πρωτοκόλλου IP. Στην τρίτη γενιά, εκτιμάται ότι το IoT θα ενσωματωθεί στο νέφος (cloud), λόγω του μεγάλου όγκου δεδομένων που πρέπει να αποθηκεύονται και να επεξεργάζονται. Παράλληλα, τα αντικείμενα θα μπορούν να δημιουργούν αυτόνομα συνδέσεις μεταξύ τους, όπως επίσης και δίκτυα ανεξάρτητα από τα δίκτυα των ανθρώπων [4].

### 2.1.3 Χαρακτηριστικά

Τα κύρια χαρακτηριστικά του IoT είναι τα ακόλουθα [5]:

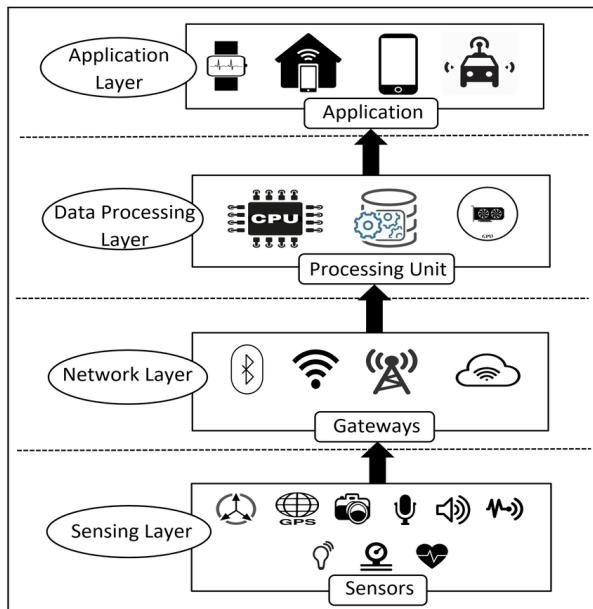
- **Διασυνδεσιμότητα:** Οποιαδήποτε συσκευή μπορεί να συνδεθεί στο παγκόσμιο δίκτυο.
- **Ετερογένεια:** Οι συσκευές στο IoT είναι αισθητά διαφορετικές μεταξύ τους, και ως προς το υλικό, και ως προς το λογισμικό.
- **Δυναμικές Αλλαγές:** Η κατάσταση των συσκευών αλλάζει συνεχώς π.χ. αδράνεια, σύνδεση, τοποθεσία, όπως επίσης και ο αριθμός των συνδεδεμένων συσκευών.
- **Τεράστια Κλίμακα:** Ο αριθμός των συσκευών που θα είναι συνδεδεμένες και θα πρέπει να διαχειρίζομαστε, αλλά και των δεδομένων που θα χρειάζονται επεξεργασία είναι γιγαντιαίος.
- **Ασφάλεια:** Είναι υψηλής σημασίας, οι συνδέσεις στο δίκτυο, καθώς και η διαχείριση των δεδομένων να γίνεται με έμφαση στην ασφάλεια.

### 2.1.4 Αρχιτεκτονική

Η αρχιτεκτονική του IoT αποτελείται από τέσσερα επίπεδα: αίσθησης, δικτύου, επεξεργασίας δεδομένων και εφαρμογών, όπως φαίνεται στην εικόνα 2.3 [6].

1. **Επίπεδο Αίσθησης:** Το επίπεδο αυτό αποτελείται από πολλούς αισθητήρες και έχει ως στόχο την ανίχνευση φαινομένων στο περιβάλλον των συσκευών, καθώς επίσης και τη συλλογή δεδομένων.
2. **Επίπεδο Δικτύου:** Το επίπεδο δικτύου δρα ως κανάλι επικοινωνίας για τη μεταφορά των δεδομένων από τους αισθητήρες σε άλλες συνδεδεμένες συσκευές. Γλοποιείται με τεχνολογίες επικοινωνιών, όπως Wi-Fi, Bluetooth, Zigbee κ.ά.
3. **Επίπεδο Επεξεργασίας Δεδομένων:** Αποτελείται από την κεντρική μονάδα επεξεργασίας των συσκευών IoT. Δέχεται τα δεδομένα που συλλέχθηκαν από το επίπεδο αίσθησης, τα αναλύει και παίρνει αποφάσεις ανάλογα με το αποτέλεσμα. Σε μερικές περιπτώσεις, ενδέχεται να αποθηκεύει και τα δεδομένα για τη βελτίωσης της εμπειρίας του χρήστη, όπως για παράδειγμα σε ένα έξυπνο ρολόι.

**4. Επίπεδο Εφαρμογών:** Στο επίπεδο αυτό συλλέγονται και παρουσιάζονται τα αποτελέσματα από το επίπεδο επεξεργασίας δεδομένων, για την υλοποίηση εφαρμογών. Οι εφαρμογές υλοποιούν διάφορες εργασίες για τους χρήστες χρησιμοποιώντας τις συσκευές.



Εικόνα 2.3: Επίπεδα αρχιτεκτονικής IoT [6]

#### 2.1.5 Πρωτόκολλα Επικοινωνίας Δεδομένων

Υπάρχουν πολλά διαφορετικά πρωτόκολλα δεδομένων που χρησιμοποιούνται στο IoT για τη μετάδοση μηνυμάτων, ανάλογα με τις απαιτήσεις και τις συσκευές του συστήματος [10]. Στο σύστημα που υλοποιήθηκε, χρησιμοποιήθηκαν τα πρωτόκολλα *MQTT* και *AMQP*, τα οποία αγαλύονται παρακάτω.

##### α) *Message Queue Telemetry Transport (MQTT)*

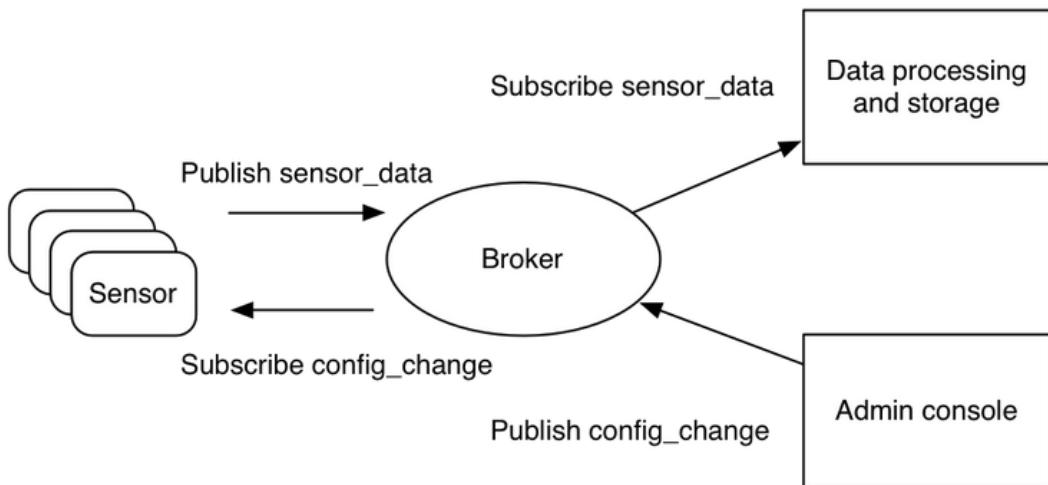
Το MQTT<sup>1</sup> είναι ένα πρότυπο του 2013 από τον Οργανισμό για την Προώθηση Δομημένων Προτύπων Πληροφοριών (Organization for the Advancement of Structured Information Standards – OASIS). Πρόκειται για ένα πρωτόκολλο ανταλλαγής μηνυμάτων που εφευρέθηκε από τον Andy Stanford-Clark και την Arlen Nipper το 1999.

Το πρότυπο βασίζεται στο μοντέλο publish-subscribe, όπου ο αποστολέας δε στέλνει απευθείας τα μηνύματα στον παραλήπτη, αλλά τα ομαδοποιεί σε κλάσεις, ενώ ο παραλήπτης εγγράφεται σε αυτές τις κλάσεις και λαμβάνει τα μηνύματά τους. Το πρότυπο ορίζει δύο οντότητες δικτύου: έναν μεσολαβητή μηνυμάτων (broker) και έναν αριθμό από πελάτες (clients). Ο μεσολαβητής είναι ένας διακομιστής, ο οποίος δέχεται όλα τα μηνύματα από τους clients και έπειτα τα δρομολογεί στους σχετικούς παραλήπτες. Client είναι οτιδήποτε μπορεί να επικοινωνήσει με το μεσολαβητή για να στείλει ή να λάβει μηνύματα.

<sup>1</sup> <https://mqtt.org/>

Η διαδικασία με την οποία πραγματοποιείται η επικοινωνία μεταξύ των πελατών, όπως φαίνεται και στην εικόνα 2.4, είναι η εξής [11]:

1. Ο client συνδέεται στο μεσολαβητή. Μπορεί να εγγραφεί σε οποιοδήποτε «θέμα» (topic) μηνυμάτων.
2. Ο client δημοσιεύει ένα μήνυμα σε ένα συγκεκριμένο topic, στέλνοντας το μήνυμα και το topic στο μεσολαβητή.
3. Ο μεσολαβητής προωθεί το μήνυμα σε όλους τους clients που έχουν εγγραφεί στο συγκεκριμένο topic.



Εικόνα 2.4: Παράδειγμα ανταλλαγής μηνυμάτων με MQTT [11]

Τα πλεονεκτήματα του MQTT είναι η απλότητά του, το μικρό μέγεθος των μηνυμάτων του, η χρήση ελάχιστου εύρους ζώνης δικτύου και πόρων. Όλοι αυτοί οι λόγοι καθιστούν το MQTT ιδανικό πρωτόκολλο για το IoT αλλά και εφαρμογές κινητών τηλεφώνων, όπου το εύρος ζώνης και η μπαταρία είναι υψηλής σημασίας.

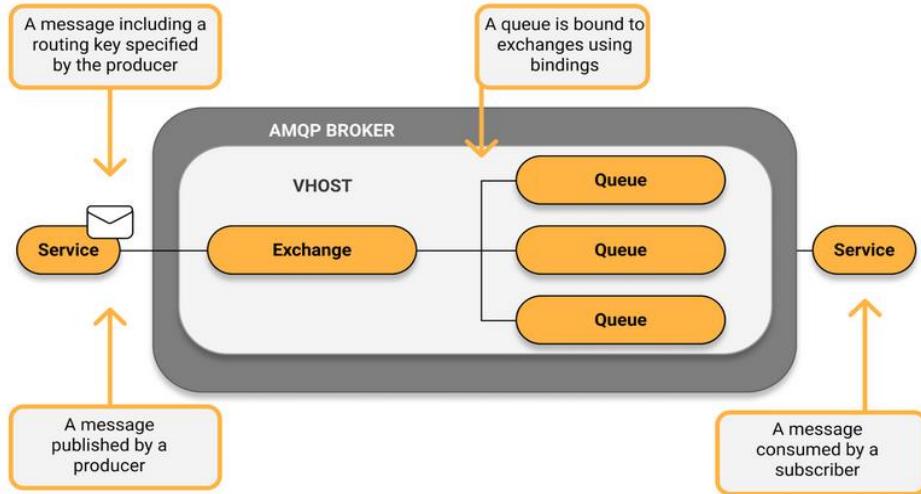
### β) Advanced Message Queuing Protocol (AMQP)

Το AMQP<sup>1</sup> είναι ένα ακόμα πρότυπο του OASIS που σχεδιάστηκε από τον John O’Hara κυρίως για τον κλάδο της χρηματοοικονομικής βιομηχανίας και χρησιμοποιεί το μοντέλο publish-subscribe, όπως και το MQTT.

Η κύρια διαφορά σε σχέση με το MQTT, είναι ότι ο μεσολαβητής αποτελείται από δύο τμήματα: τις ανταλλαγές (exchanges) και τις ουρές (queues). Τα exchanges είναι υπεύθυνα για τη παραλαβή των μηνυμάτων από τους client, και τη δρομολόγησή τους στις ουρές που έχουν συνδεθεί σε αυτές. Ανάλογα με τον τύπο του exchange αλλάζει η μέθοδος της δρομολόγησης. Για παράδειγμα, το ‘fanout’ exchange δρομολογεί κάθε μήνυμα που δέχεται σε όλες τις συνδεδεμένες σε αυτή ουρές, ενώ το ‘direct’ exchange δρομολογεί κάθε μήνυμα με βάση το κλειδί δρομολόγησής (routing key) του, το οποίο πρέπει να ταυτίζεται με το κλειδί δρομολόγησης με το οποίο συνδέθηκε η ουρά στο exchange [12]. Η διαδικασία φαίνεται αναλυτικά στην εικόνα 2.5<sup>2</sup>.

<sup>1</sup> <https://www.amqp.org/>

<sup>2</sup> <https://www.cloudamqp.com/blog/what-is-amqp-and-why-is-it-used-in-rabbitmq.html>



Εικόνα 2.5: Παράδειγμα ανταλλαγής μηνυμάτων με AMQP

Σε σύγκριση με το MQTT, το AMQP προσφέρει μεγαλύτερη ασφάλεια, καλύτερη χρήση πόρων και αξιοπιστία στη μεταφορά των μηνυμάτων, ενώ παράλληλα, λόγω της πιο περίπλοκης δομής του μεσολαβητή, επιτρέπει τη δημιουργία πιο σύνθετων εφαρμογών για την αποστολή και λήψη μηνυμάτων.

## 2.2 Ρομποτική

Η ρομποτική είναι ο επιστημονικός κλάδος που ασχολείται με το σχεδιασμό, την κατάσκευή και τη λειτουργία των ρομπότ, με σκοπό να βοηθούν τον άνθρωπο στην καθημερινή του ζωή. Οι εφαρμογές της ρομποτικής είναι πολυάριθμες και εμφανίζονται σε πολλούς τομείς, όπως στη βιομηχανία, στην ιατρική, για οικιακή χρήση κ.ά. Όσο περισσότερο εξελίσσονται και γίνονται πιο αποδοτικά τα ρομποτικά συστήματα, τόσο περισσότερο διαδεδομένα θα γίνονται, το οποίο επιβεβαιώνεται και από τις εικόνες 1.1, 1.2.

### 2.2.1 Δίκτυα και ρομποτική

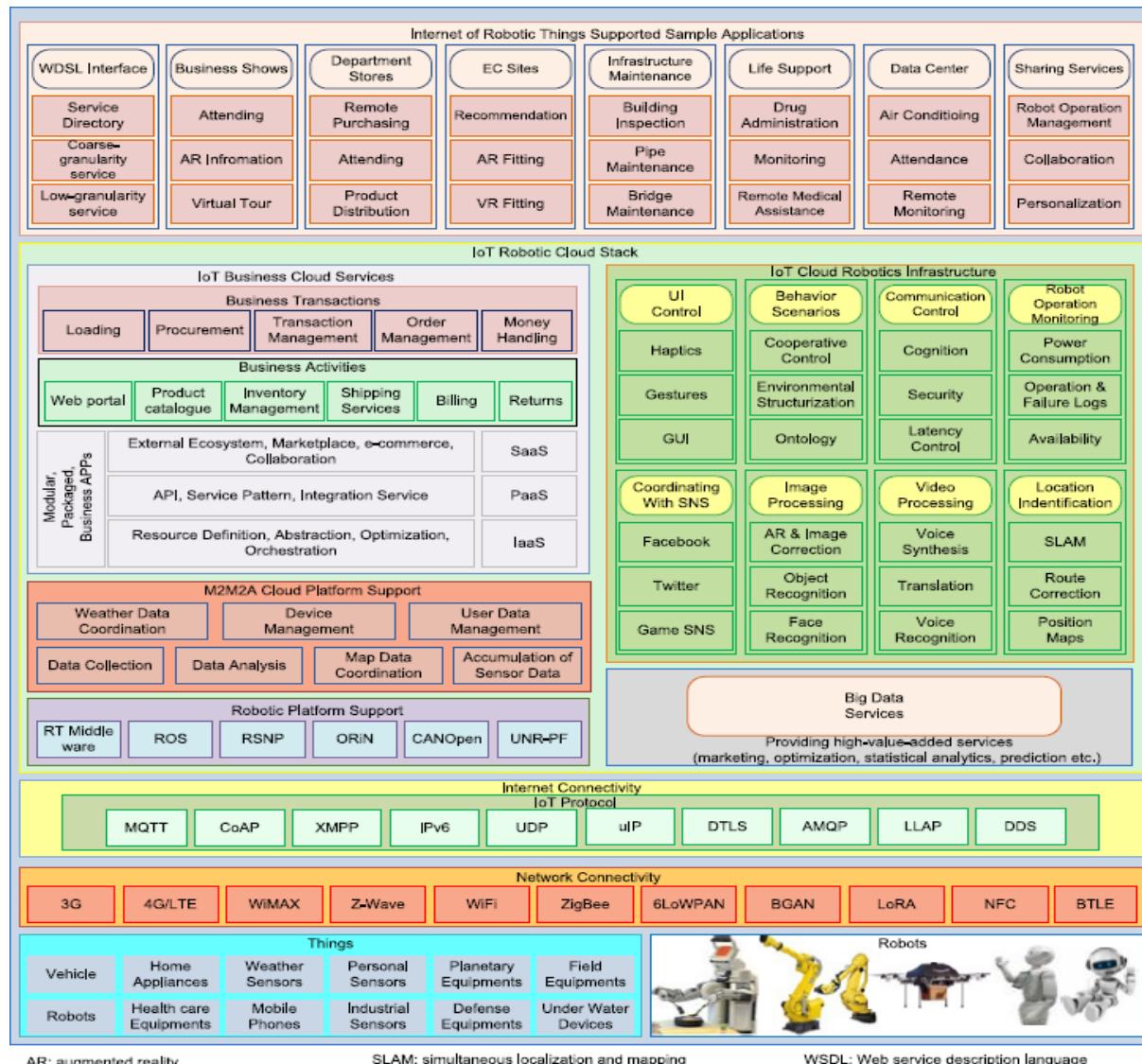
Το επόμενο βήμα για την εξέλιξη της ρομποτικής είναι η ενσωμάτωση των ρομπότ σε ένα δίκτυο. Σε πρώτη φάση, εμφανίστηκαν τα «Δικτυωμένα Ρομπότ» (Networked Robots), δηλαδή μια συλλογή από ρομποτικές συσκευές που είναι συνδεδεμένες σε ένα δίκτυο. Διακρίνονται σε τηλεχειριζόμενα ρομπότ, που ελέγχονται από απόσταση μέσω δικτύου από έναν άνθρωπο, και σε συστήματα πολλαπλών-ρομπότ, όπου πολλά ρομπότ είναι συνδεδεμένα στο δίκτυο και ανταλλάσσουν δεδομένα. Τα δικτυωμένα ρομπότ όμως, υποφέρουν από φυσικούς περιορισμούς, όπως μικρό μέγεθος μνήμης, χαμηλή ταχύτητα εκτέλεσης, καθυστέρηση δικτύου και έλλειψη ευφυΐας [3].

Λόγω αυτών των περιορισμών, έκανε την εμφάνισή της η «Ρομποτική Νέφους» (Cloud Robotics), όπου γίνεται χρήση του υπολογιστικού νέφους (cloud computing), δηλαδή η διάθεση υπολογιστικών πόρων μέσω Διαδικτύου, για την επεξεργασία των δεδομένων. Παρ' όλα αυτά, ένα μέρος της επεξεργασίας ενδέχεται να γίνει τοπικά λόγω χρονικών περιορισμών. Αν και προσφέρει αρκετά πλεονεκτήματα, όπως μεγάλη υπολογιστική ισχύ και συλλογική μάθηση ρομπότ, παρουσιάζει μειονεκτήματα, όπως μεταβαλλόμενη καθυστέρηση δικτύου, ετερογένεια, ασφάλεια, διαχείριση πολλαπλών ρομπότ κ.ά.

## 2.2.2 Internet of Robotic Things

Τα προβλήματα που αντιμετωπίζει η Ρομποτική Νέφους έρχεται να λύσει το «Διαδίκτυο των Ρομποτικών Προγράμματων» (Internet of Robotic Things – IoRT). Η διαφορά με το IoT είναι, ότι πλέον στο IoRT και τα ρομπότ θεωρούνται αντικείμενα, τα οποία προσφέρουν περισσότερες δυνατότητες, όπως αλληλεπίδραση με το περιβάλλον, μετακίνηση στο χώρο και αυτονομία. Χρησιμοποιώντας πτυχές της Ρομποτικής Νέφους και του IoT, στοχεύει στη παροχή μεγάλης ευελιξίας στο σχεδιασμό και την υλοποίηση νέων εφαρμογών, αλλά και κατανεμημένων υπολογιστικών πόρων [3].

Η αρχιτεκτονική του IoRT μπορεί να διαχωριστεί σε πέντε επίπεδα: υλικού, δικτύου, Internet, υποδομής και εφαρμογών, όπως φαίνεται στην εικόνα 2.6 [3].



Εικόνα 2.6: Επίπεδα αρχιτεκτονικής IoRT [3]

## ΚΕΦΑΛΑΙΟ 2. ΘΕΩΡΗΤΙΚΟ ΥΠΟΒΑΘΡΟ

1. **Επίπεδο Υλικού:** Το χαμηλότερο επίπεδο, που αποτελείται από διάφορα ρομπότ, συσκευές και αισθητήρες, με σκοπό την ανίχνευση φαινομένων στο περιβάλλον και την αποστολή τους στο επίπεδο δικτύου.
2. **Επίπεδο Δικτύου:** Σε αυτό το επίπεδο εμφανίζονται τεχνολογίες επικοινωνιών μικρών αποστάσεων, όπως Wi-Fi, BLE, BGAN και NFC για ανεμόδιστη επικοινωνία μεταξύ των ρομπότ, αλλά και μεσαίων-μεγάλων αποστάσεων, όπως WiMAX, Z-Wave, Zigbee και LoRA για μεταβίβαση πληροφοριών σε μεγαλύτερες αποστάσεις.
3. **Επίπεδο Internet:** Η σύνδεση στο Διαδίκτυο είναι πολύ σημαντικό κομμάτι του IoRT. Πέρα από πρωτόκολλα επικοινωνίας του Internet, έχουν προστεθεί και πρωτόκολλα αποστολής μηνυμάτων που χρησιμοποιούνται στο IoT, όπως MQTT, AMQP, DDS, XMPP και CoAP.
4. **Επίπεδο Υποδομής:** Αυτό το επίπεδο αποτελείται από πέντε διαφορετικά τμήματα, το Robotic Platform Support που προσφέρει υπηρεσίες σχετικές με ρομπότ, όπως RT (Robot Technology), ROS, RSNP κ.ά., το M2M2A Cloud Platform Support (Machine-to-Machine-to-Actuator) για υπηρεσίες συλλογής και ανάλυσης δεδομένων από αισθητήρες, διαχείρισης συσκευών κ.ά., τα IoT Business Cloud Services που προσφέρουν υπηρεσίες σε οργανισμούς και κατασκευαστές ρομποτικών συστημάτων για να μειώσουν τις δαπάνες τους, τα Big Data Services και τέλος, το IoT Cloud Robotics που ενισχύει τα ρομποτικά συστήματα με διάφορες υπηρεσίες, όπως φαίνεται στην εικόνα 2.6.
5. **Επίπεδο Εφαρμογών:** Το υψηλότερο επίπεδο, που περιέχει τις εφαρμογές που μπορούν να υλοποιηθούν μέσω του IoRT.

Η αρχιτεκτονική του IoRT μοιάζει αρκετά με την αρχιτεκτονική του IoT, αλλά είναι εμπλουτισμένη με πολλές υπηρεσίες και λειτουργίες, που σκοπεύουν στην αποτελεσματική ενσωμάτωση των ρομπότ, αλλά και την σωστή αξιοποίηση του cloud.

Τα βασικά χαρακτηριστικά γνωρίσματα του IoRT διαφέρουν από τις παραδοσιακές ρομποτικές υπηρεσίες των δικτυωμένων ρομπότ και της Ρομποτικής Νέφους και περιγράφονται παρακάτω [3]:

- **Εύκολη Σύνθεση:** Εφόσον γίνεται χρήση Web Service Description Language (WDSL) για την επικοινωνία μεταξύ των ρομπότ και των άλλων τμημάτων του IoRT, διευκολύνεται η δημιουργία σύνθετων εφαρμογών.
- **Αντίληψη:** Από τις πληροφορίες που συλλέγονται, οι αισθητήρες αποκτούν γνώση για το περιβάλλον τους και μέσω αυτής της γνώσης λαμβάνονται οι αποφάσεις από τα ρομποτικά συστήματα.
- **Ποικιλομορφία:** Στο IoRT υποστηρίζονται ετερογενή ρομπότ και ρομποτικά συστήματα που έχουν εντελώς διαφορετικό υλικό και λογισμικό.
- **Επεκτασιμότητα:** Για την επέκταση των υπαρχουσών ρομποτικών υπηρεσιών, μπορούν είτε να προστεθούν νέα είδη ρομπότ, είτε να δημιουργηθούν νέες υπηρεσίες στο σύστημα που θα είναι προσβάσιμες μέσω της διεπαφής δικτύου.

- **Διαλειτουργικότητα:** Λόγω της υποστήριξης πολυάριθμων πρωτοκόλλων επικοινωνίας, διαφορετικές συσκευές και ρομπότ επικοινωνούν μεταξύ τους χωρίς δυσκολία.
- **Προσαρμοστικότητα:** Τα συστήματα θα πρέπει να έχουν τη δυνατότητα να προσαρμοστούν στις αλλαγές του περιβάλλοντος και να λάβουν δράση με βάση αυτές.

## 2.3 Γραφικός Προγραμματισμός

Παρά τη μεγάλη εξάπλωση της ρομποτικής σε διάφορες πτυχές της ζωής μας, ο προγραμματισμός ενός ρομπότ παραμένει δύσκολη και απαιτητική διαδικασία. Αυτό οφείλεται στο γεγονός, ότι η ρομποτική είναι ένα διεπιστημονικό πεδίο που ενσωματώνει γνώσεις από τους κλάδους της επιστήμης του ηλεκτρολόγου μηχανικού, του μηχανολόγου μηχανικού και της επιστήμης υπολογιστών. Προς αυτήν την κατεύθυνση, γίνονται προσπάθειες για την απλοποίηση του προγραμματισμού ενός ρομπότ, ώστε να μπορεί και ένας χρήστης χωρίς γνώσεις να αναπτύξει εφαρμογές για ρομποτικά συστήματα.

Έχουν εξεταστεί πολλοί και διαφορετικοί τρόποι απλούστευσης του προγραμματισμού ενός ρομπότ. Ένας τρόπος είναι η επίδειξη της κίνησης που θα εκτελεί το ρομπότ, καθοδηγώντας το είτε με το χέρι [7], είτε μέσω επαυξημένης πραγματικότητας [8]. Μια άλλη προσέγγιση, είναι η εκμάθηση του ρομπότ μέσω «*tangible programming*»<sup>1</sup>, όπου οι εντολές σχεδιάζονται στο χαρτί και ανιχνεύονται από το σύστημα όρασης του ρομπότ [9].

Η πιο δημοφιλής μέθοδος όμως, είναι η χρήση οπτικών γλωσσών προγραμματισμού (visual programming languages – VPL), όπου το πρόγραμμα δεν δημιουργείται με τη συγγραφή κώδικα, αλλά με τη χρήση προγραμματιστικών στοιχείων μέσω ενός γραφικού περιβάλλοντος. Χρησιμοποιώντας τέτοιες γλώσσες, ο χρήστης καταλαβαίνει γρηγορότερα και ευκολότερα τι λειτουργία επιτελεί κάθε στοιχείο, ενώ ταυτόχρονα η αναπαράσταση της εφαρμογής οπτικά επιτρέπει την πιο γρήγορη επιθεώρηση και αποσφαλμάτωση του προγράμματος.

Οι VPL κατηγοριοποιούνται σε καθαρά οπτικές γλώσσες, όπου το πρόγραμμα μεταγλωττίζεται απευθείας από την οπτική του αναπαράσταση, σε υβριδικές, όπου το πρόγραμμα μεταφράζεται από την οπτική μορφή του σε μια γλώσσα κειμένου, αλλά και σε μικρότερες υποκατηγορίες, όπως συστήματα προγραμματισμού-μέσω-παραδείγματος. Η ανάλυση θα επικεντρωθεί στην δεύτερη κατηγορία των υβριδικών γλωσσών [13].

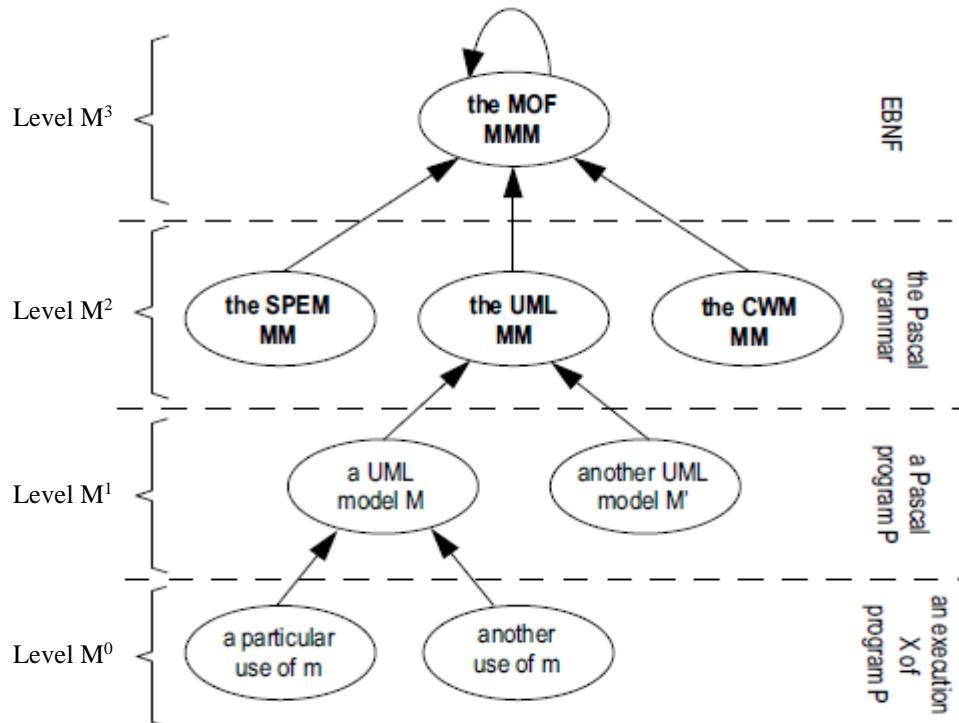
### 2.3.1 Model Driven Engineering

Για να υλοποιηθεί σωστά μια τέτοια γλώσσα, θα πρέπει να βασιστεί στις αρχές του Model Driven Engineering (MDE), που στηρίζεται σε μοντέλα και μετασχηματισμούς. Οι βασικές ιδέες της μεθοδολογίας αυτής είναι το μοντέλο και το μέτα-μοντέλο. Το μοντέλο είναι μια απλούστευμένη αναπαράσταση ενός συστήματος, ενώ το μέτα-μοντέλο είναι το σύνολο των συμβόλων και των περιορισμών που χρησιμοποιούνται κατά το σχεδιασμό του μοντέλου.

<sup>1</sup> <https://bitcubs.com/blogs/news/what-is-tangible-programming>

## ΚΕΦΑΛΑΙΟ 2. ΘΕΩΡΗΤΙΚΟ ΥΠΟΒΑΘΡΟ

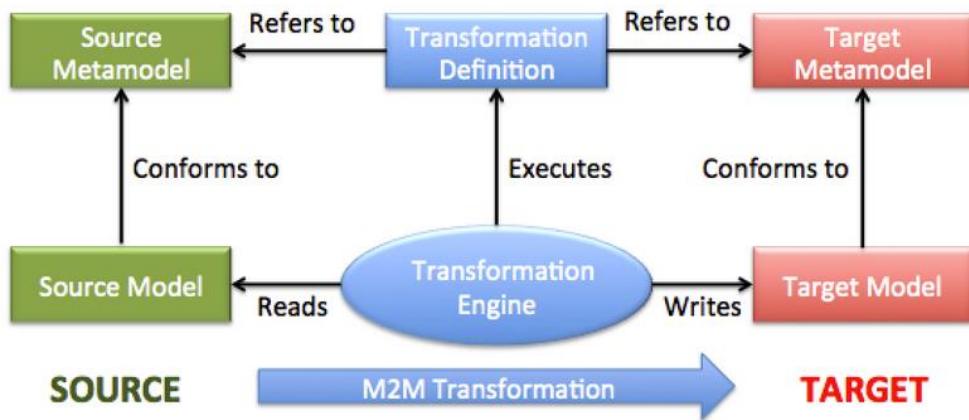
Το Object Management Group (OMG) δημιούργησε το Model Driven Architecture που παρέχει κάποιες κατευθυντήριες γραμμές για τη δόμηση των προδιαγραφών. Στην αρχιτεκτονική αυτή υπάρχουν τέσσερα επίπεδα, όπως φαίνεται και στην εικόνα 2.7. Στο ανώτερο επίπεδο  $M^3$  έχουμε το Meta Object Facility (MOF), ένα μέτα-μέτα-μοντέλο σύμφωνα με το οποίο ορίζονται όλα τα μέτα-μοντέλα. Το MOF υπάρχει για να διασφαλίζεται η συμβατότητα μεταξύ των μέτα-μοντέλων. Στο επίπεδο  $M^2$  εμφανίζονται τα μέτα-μοντέλα, το καθένα από τα οποία ορίζει μια γλώσσα για την δημιουργία των μοντέλων. Έπειτα, στο επίπεδο  $M^1$  έχουμε τα μοντέλα, που αναπαριστούν πραγματικά συστήματα, σύμφωνα με κάποιο μέτα-μοντέλο. Τέλος, στο επίπεδο  $M^0$  είναι το πραγματικό σύστημα που μοντελοποιούμε [14].



Εικόνα 2.7: Model Driven Architecture [14]

Ένα πολύ σημαντικό κοινό του MDE είναι ο μετασχηματισμός μοντέλων (Model-to-Model transformation). Ένας μετασχηματισμός είναι μια συστηματική και πλήρως αυτοματοποιημένη διαδικασία, κατά την οποία ένα μοντέλο μετασχηματίζεται σε ένα άλλο μοντέλο. Τα δύο αυτό μοντέλα μπορούν να βασίζονται σε διαφορετικά μέτα-μοντέλα. Στην ειδική περίπτωση που το τελικό μοντέλο είναι μια γλώσσα προγραμματισμού, ο μετασχηματισμός ονομάζεται Model-to-Text transformation [15]. Η διαδικασία περιγράφεται στην εικόνα 2.8.

Συνεπώς, το αρχικό μοντέλο μπορεί να δημιουργηθεί με τη χρήση μιας γραφικής διεπαφής και το μοντέλο που θα προκύψει μετά το μετασχηματισμό να είναι πηγαίος κώδικας που θα μπορεί να χρησιμοποιηθεί από κάποιο ρομπότ. Με αυτόν τον τρόπο αποστασιοποιείται σημαντικά η σχεδίαση της εφαρμογής, από τις λεπτομέρειες της υλοποίησης, επιτρέποντας σε χρήστες χωρίς ειδικές γνώσεις προγραμματισμού και ρομποτικής, να δημιουργήσουν τις δικές του εφαρμογές.



Εικόνα 2.8: Μετασχηματισμός μοντέλου [15]

## Κεφάλαιο 3

# ΣΧΕΤΙΚΗ ΒΙΒΛΙΟΓΡΑΦΙΑ

Στο κεφάλαιο αυτό θα αναλυθεί η ήδη υπάρχουσα βιβλιογραφία σχετικά με το πρόβλημα που προσπαθεί να επιλύσει η διπλωματική εργασία. Στην Ενότητα 3.1 περιγράφονται μερικές εφαρμογές, όπου έχει ενσωματωθεί η ρομποτική σε συστήματα IoT. Έπειτα, στην Ενότητα 3.2 αναλύονται ορισμένες μέθοδοι με τις οποίες έχει γίνει απομακρυσμένος έλεγχος και επικοινωνία μέσω του ROS. Τέλος, στην Ενότητα 3.3 παρουσιάζονται διάφορα περιβάλλοντα ανάπτυξης ρομποτικών εφαρμογών μέσω γραφικών διεπαφών.

### 3.1 IoT και Ρομποτική

Όπως έχουμε ήδη εξηγήσει, ο συνδυασμός της ρομποτικής και του IoT αναμένεται να επιφέρει πολύ σημαντικά πλεονεκτήματα σε σχέση με τα συνηθισμένα συστήματα IoT. Οι τομείς στους οποίους θα εμφανιστούν τέτοιες εφαρμογές είναι πολυάριθμοι, αλλά θα εστιάσουμε ενδεικτικά σε δύο, στον τομέα της ιατρικής και της βιοϊατρικής τεχνολογίας, και στον τομέα των ευφυών συστημάτων μετακίνησης.

#### 3.1.1 Ιατρική – Βιοϊατρική Τεχνολογία

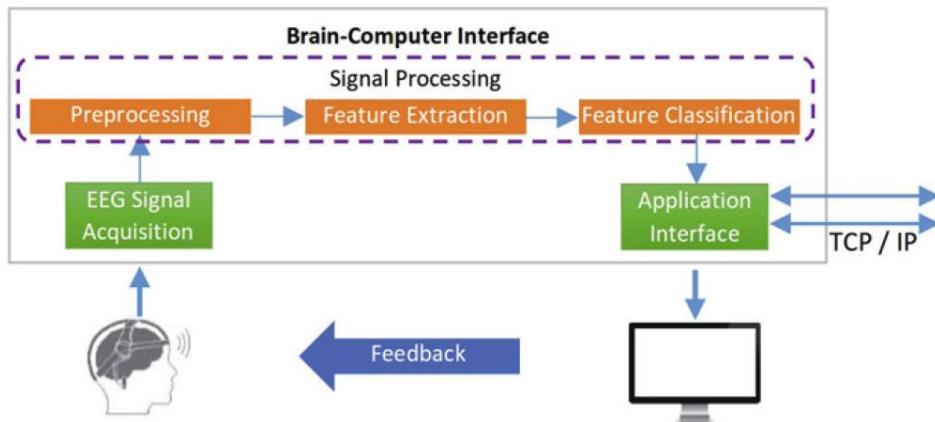
Οι εφαρμογές της ρομποτικής στον κλάδο της ιατρικής είναι πολυπληθείς, όπως για παράδειγμα «κοινωνικά ρομπότ» που κρατούν συντροφιά στους ασθενείς [18], ρομπότ που απολυμαίνουν ιατρικούς χώρους [19] και ρομπότ που χρησιμοποιούνται για χειρουργικές επεμβάσεις [20]. Τα τελευταία χρόνια, έχει ξεκινήσει η εμφάνιση διάφορων εφαρμογών βασισμένων στο IoT.

Μία τέτοια εφαρμογή περιγράφεται στο [21], η οποία επιδιώκει να ενώσει τις τεχνολογίες της ρομποτικής και του IoT για τη δημιουργία ενός συστήματος που θα εκτελεί ελέγχους ρουτίνας σε ασθενείς ή για την καταπολέμηση επιδημιών. Τα ρομπότ θα μπορούν εύκολα να μετακινηθούν για να πλησιάσουν τους ασθενείς, ενώ θα είναι εξοπλισμένα με αισθητήρες για διάφορες μετρήσεις, όπως της αρτηριακής πίεσης, και του σφυγμού της καρδιάς. Παράλληλα, η αποθήκευση των ιατρικών δεδομένων αλλά και η επεξεργασία πολύπλοκων γεγονότων θα πραγματοποιείται στο cloud. Αξιοποιώντας την κινητικότητα των ρομπότ, τη δυνατότητα μετρήσεων των έξυπνων συσκευών και την επεξεργαστική ισχύ του cloud για τον εντοπισμό των ασθενειών, το σύστημα αυτό έχει τεράστιες προοπτικές στον τομέα της υγείας.

Στο [22] αναλύεται ένα σύστημα που έχει ως στόχο τη βελτίωση της ανθρώπινης αλληλεπίδρασης εξ' αποστάσεως. Το κύριο στοιχείο αυτού του συστήματος, είναι ένα ρομπότ που προσομοιώνει την επαφή κάθε ανθρώπου με το συνομιλητή του. Αρχικά, συλλέγονται δεδομένα σχετικά με τα συναισθήματα του χρήστη μέσω των έξυπνων ρούχων που φοράει, αλλά και μέσω των δραστηριοτήτων του από το κινητό του. Αυτά τα δεδομένα, αποστέλλονται στο cloud, όπου και γίνεται η επεξεργασία τους, για την αναγνώριση

των συναισθημάτων του χρήστη. Όταν ο χρήστης καλέσει έναν άλλο χρήστη, το ρομπότ του καθενός δρα ως το άλλο άτομο, δηλαδή εκφράζει τα συναισθήματα του άλλου χρήστη, αλλά και άλλα χαρακτηριστικά, όπως θερμοκρασία σώματος, σφυγμό, ακόμα και αγκαλιά. Με αυτόν τον τρόπο, οι δύο χρήστες παρόλο που βρίσκονται μακριά, μπορούν να μοιραστούν τα συναισθήματά τους πολύ καλύτερα απ' ότι με μία απλή κλήση.

Ενδιαφέρουσες εφαρμογές μπορούν να προκύψουν από συστήματα όπου ο έλεγχος του ρομπότ γίνεται μέσω εγκεφάλου. Σε τέτοια συστήματα, χρησιμοποιείται ένας ηλεκτροεγκεφαλογράφος, ο οποίος μπορεί να εντοπίσει εκφράσεις προσώπου, νοητικές εντολές αλλά και συναισθήματα, τα οποία στη συνέχεια αντιστοιχίζονται σε εντολές κίνησης του ρομπότ. Η διαδικασία ανάλυσης των ηλεκτρικών σημάτων του εγκεφάλου παρουσιάζεται συνοπτικά στην εικόνα 3.1.



Εικόνα 3.1: Σύστημα ανάλυσης σημάτων εγκεφάλου [24]

Ένα τέτοιο σύστημα παρουσιάζεται στο [23], όπου στόχος είναι η βοήθεια παιδιών με ειδικές μαθησιακές δυσκολίες σε εκπαιδευτικές δραστηριότητες. Χρησιμοποιώντας έναν ηλεκτροεγκεφαλογράφο, εντοπίζει εκφράσεις προσώπου και συναισθήματα του χρήστη. Έπειτα, μέσω του Node-RED γίνεται η αντιστοίχιση μεταξύ αυτών των εντοπισμών και των κινήσεων του ρομπότ. Οι εντολές αυτές στέλνονται στο Arduino<sup>1</sup>, που οδηγεί τους δύο κινητήρες του ρομπότ.

Παρόμοιο είναι και το σύστημα που αναλύεται στο [24]. Σε αντίθεση με το προηγούμενο σύστημα, εδώ εντοπίζεται το επίπεδο προσοχής του χρήστη για τον εντοπισμό της εγκεφαλικής δραστηριότητας. Αν το επίπεδο προσοχής ξεπεράσει κάποιο όριο, τότε δίνεται η εντολή στο ρομπότ να ξεκινήσει την κίνησή του. Όλες οι επικοινωνίες μεταξύ των συσκευών γίνονται ασύρματα μέσω Bluetooth και ασύρματων TCP συνδέσεων.

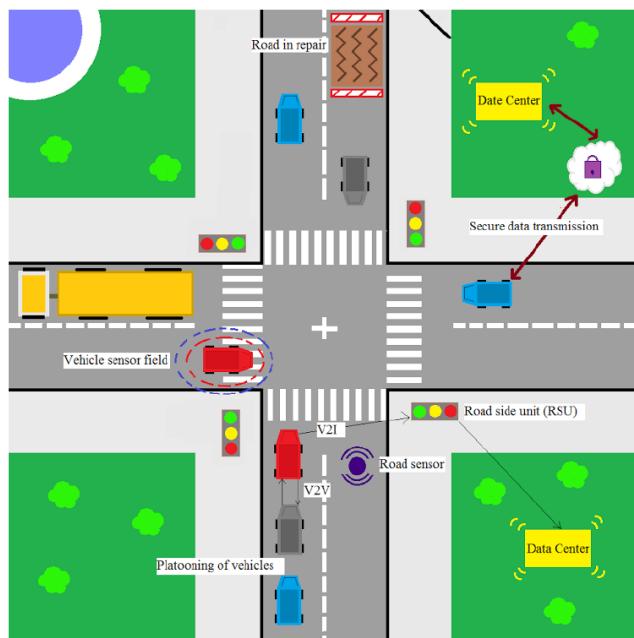
Τέλος, στο [25] περιγράφεται ένα σύστημα ελέγχου ρομπότ μέσω εγκεφάλου για τηλεπαρούσία, που θα επιτρέπει στους χρήστες με κινητικά προβλήματα, να ακολουθούν τους φίλους τους σε διάφορες δραστηριότητες. Για την ανίχνευση εγκεφαλικής δραστηριότητας, οι χρήστες φαντάζονται κάποια κίνηση είτε προς τα αριστερά είτε προς τα δεξιά, η οποία στην συνέχεια αντιστοιχίζεται σε ανάλογη κίνηση του ρομπότ. Ο χρόνος που χρειάστηκαν οι χρήστες για να ολοκληρώσουν μια διαδρομή, είναι συγκρίσιμος με το χρόνο που χρειάστηκαν για έλεγχο με τηλεχειρισμό.

<sup>1</sup> <https://www.arduino.cc/>

### 3.1.2 Ευφυή Συστήματα Μεταφορών

Τα ευφυή συστήματα μεταφορών (Intelligent Transportation Systems – ITS) είναι ένα από τα βασικά στοιχεία μιας έξυπνης πόλης [26]. Χρησιμοποιώντας τεχνολογίες του IoT και των τηλεπικοινωνιών, στοχεύουν στη βελτίωση των μετακινήσεων των ανθρώπων, μειώνοντας αισθητά την κυκλοφοριακή συμφόρηση και αυξάνοντας την ασφάλεια των επιβατών. Το επόμενο βήμα στα ITS είναι η ενσωμάτωση των αυτόνομων οχημάτων, που παρουσιάζουν σημαντική ανάπτυξη τα τελευταία χρόνια [27].

Όπως περιγράφονται στα [28] και [29], τα συστήματα αυτά αποτελούνται από αισθητήρες και συσκευές όπως οχήματα και υπηρεσίες cloud για την αποθήκευση και επεξεργασία των δεδομένων. Τα οχήματα σε αυτήν την περίπτωση παίζουν το ρόλο των ρομπότ σε ένα σύστημα IoRT, αφού μπορούν να μετακινηθούν μόνα τους και είναι εξοπλισμένα με αισθητήρες για αλληλεπίδραση με το περιβάλλον τους. Έχουν τη δυνατότητα να επικοινωνούν μεταξύ τους, αλλά και με τις συσκευές του δρόμου για να έχουν καλύτερη εικόνα της κατάστασης στο δρόμο, ενώ παράλληλα προσφέρουν διάφορες παροχές ψυχαγωγίας στους επιβάτες.



Εικόνα 3.2: Αναπαράσταση ενός ευφυούς συστήματος μεταφορών [28]

Ένα σημαντικό ζήτημα σε τέτοια είδους συστήματα, είναι ο τρόπος επικοινωνίας μεταξύ των οχημάτων. Στο [30] παρουσιάζεται το CarSpeak, ένα σύστημα για την επικοινωνία αυτόνομων οχημάτων, όπου κάθε οχήμα μπορεί να έχει πρόσβαση σε αισθητήρες άλλων οχημάτων, αλλά και σε στατικούς αισθητήρες της υποδομής. Δεδομένου ότι πολλά αυτόνομα οχήματα χρησιμοποιούν το ROS, το CarSpeak επικοινωνεί κανονικά μαζί του και δρα ως αισθητήρας που στέλνει δεδομένα στον planner για το σχεδιασμό της τροχιάς του οχήματος. Χρησιμοποιώντας ένα περιεχομενο-κεντρικό (content-centric) έλεγχο πρόσβασης μέσων (MAC) για τη μεταβίβαση της πληροφορίας, λειτουργεί αρκετά καλύτερα σε μεγάλο αριθμό οχημάτων σε σύγκριση με το πρότυπο 802.11, ενώ ταυτόχρονα συμπιέζει σε πολύ ικανοποιητικό επίπεδο τα δεδομένα, που είναι πολύ σημαντικό λόγω του μεγάλου όγκου δεδομένων σε τέτοια συστήματα.

Πέρα από τα αυτόνομα οχήματα που θα χρησιμοποιούν οι επιβάτες για τη μετακίνησή τους, πολύ χρήσιμα στα ITS θα είναι και τα μη επανδρωμένα εναέρια οχήματα (Unmanned Aerial Vehicles – UAV) [31], [32]. Μέσω αυτών θα αυτοματοποιηθεί η οδική βιοήθεια, η τροχονόμευση και οι ομάδες διάσωσης, αφού τα UAV μπορούν να φτάσουν γρηγορότερα στην τοποθεσία ενός ατυχήματος, έχουν περισσότερη ευελιξία κινήσεων και δεν εμποδίζονται από την κυκλοφοριακή συμφόρηση. Συγχρόνως, μπορούν να βοηθούν στην παρακολούθηση των οχημάτων για τη διασφάλιση της τήρησης των Κανόνων Οδικής Κυκλοφορίας, αλλά και στην μεταβίβαση πληροφοριών μεταξύ των οχημάτων.

## 3.2 Απομακρυσμένος έλεγχος ρομπότ μέσω ROS

Όπως έχει ήδη αναφερθεί, ο ένας από τους κύριους στόχους της διπλωματικής εργασίας είναι ο απομακρυσμένος έλεγχος ενός ρομπότ που χρησιμοποιεί το ROS. Δυστυχώς, αυτό δεν είναι δυνατό απευθείας μέσω του ROS, διότι η διαχείριση του ρομπότ περιορίζεται στο τοπικό δίκτυο. Στη συνέχεια, θα παρουσιαστούν μερικές μέθοδοι που προσπαθούν να αντιμετωπίσουν το πρόβλημα αυτό.

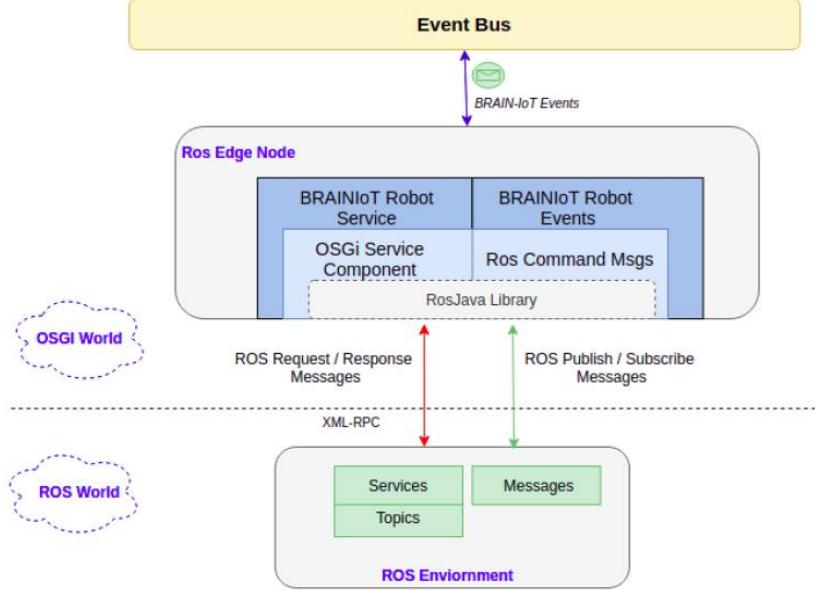
Μία λύση είναι το Rosbridge [33], το οποίο χειρίζεται το ROS ως ένα “back end”, παρέχοντας ένα JSON API για την επικοινωνία με αυτό. Το ρομπότ σε αυτήν την περίπτωση, δρα ως διακομιστής δικτύου και απαντάει στα αιτήματα που δέχεται μέσα από το Διαδίκτυο. Αν και μπορεί να επικοινωνήσει με οποιαδήποτε γλώσσα υποστηρίζει JSON, λόγω της μεγάλης διάδοσης της JavaScript σε εφαρμογές δικτύου, παρέχει μία βιβλιοθήκη JavaScript με πολλά χαρακτηριστικά, τη roslibjs. Μέσω αυτής παρέχονται πολλές δυνατότητες, όπως οπτικοποίηση και χρήση αλγορίθμων πλοήγησης. Συνεπώς, ο χρήστης μπορεί μέσω μιας διεπαφής σε φυλλομετρητή, να διαχειρίζεται και να ενημερώνεται για διάφορες λειτουργίες και πληροφορίες του ρομπότ, ενώ συγχρόνως μπορούν να επικοινωνήσουν με αυτό και συσκευές που δε βασίζονται στο ROS.

Σε αντίθεση με την παραπάνω προσέγγιση, όπου έχουμε αρχιτεκτονική πελάτη-διακομιστή δύο επιπέδων, στο [34] αναλύεται το ROSLink, το οποίο προτείνει μια αρχιτεκτονική τριών επιπέδων. Σε αυτήν την περίπτωση, η εφαρμογή και το ρομπότ αποτελούν τους πελάτες και ο διακομιστής, που είναι ανεξάρτητος, γεφυρώνει την επικοινωνία μεταξύ των δύο. Σε πρώτο στάδιο, τα δεδομένα του ROS μετατρέπονται σε μορφή JSON στο ROSLink Bridge. Ακολούθως, στέλνονται στο διακομιστή ROSLink Proxy and Cloud, ο οποίος τα μεταβιβάζει στην εφαρμογή του χρήστη. Η αντίστροφη διαδικασία ακολουθείται για την αποστολή μηνυμάτων από την εφαρμογή στο ROS.

Στο [35] περιγράφεται το ROS Edge Node, το οποίο στοχεύει στη διασύνδεση του ROS με πλατφόρμες IoT. Για την υποστήριξη πολλών πλατφορμών IoT, γίνεται χρήση του BRAIN-IoT<sup>1</sup>, μιας υποδομής που προσφέρει πολλές υπηρεσίες για τη διευκόλυνση της ανάπτυξης εφαρμογών IoT. Η δομή του ROS Edge Node φαίνεται στην εικόνα 3.3. Αρχικά, χρησιμοποιώντας τη ROSJava, μπορεί να λάβει και να στείλει δεδομένα στο ROS, καθώς επίσης και να τα μετατρέψει με ευκολία σε αντικείμενα Java. Η επικοινωνία μεταξύ του BRAIN-IoT και των Java δεδομένων του ROS, γεφυρώνεται μέσω της OSGi<sup>2</sup> προδιαγραφής, η οποία παρέχει καλύτερες δυνατότητες, όπως ενημέρωση των συμπεριφορών του ρομπότ κατά την εκτέλεση.

<sup>1</sup> <https://www.brain-iot.eu/>

<sup>2</sup> <https://www.osgi.org/>



Εικόνα 3.3: Δομή του ROS Edge Node [35]

Τέλος, στο [36] παρουσιάζεται το ROStful, ένας διακομιστής δικτύου, μέσω του οποίου, γίνονται διαθέσιμα topics και υπηρεσίες του ROS ως RESTful<sup>1</sup> υπηρεσίες δικτύου. Οι υπηρεσίες αυτές χρησιμοποιούν κυρίως την αντιστοίχιση JSON του Rosbridge για τα μηνύματα ROS. Ωστόσο, διαφέρει από αυτό, διότι παρέχει τα θέματα και τις υπηρεσίες του ROS ως απλές αιτήσεις post και get, και όχι με χρήση web socket. Επιπλέον, ο ROStful διακομιστής είναι συμβατός με το WSGI<sup>2</sup> και επομένως, μπορεί να χρησιμοποιηθεί με τους περισσότερους διακομιστές δικτύου.

### 3.3 Γραφικά περιβάλλοντα ανάπτυξης ρομποτικών εφαρμογών

Όπως έχουμε ήδη αναφέρει, ο προγραμματισμός και η ανάπτυξη ρομποτικών εφαρμογών είναι περίπλοκη και χρονοβόρα διαδικασία και γίνονται σημαντικές προσπάθειες για την απλοποίησή της. Ως εκ τούτου, έχουν υλοποιηθεί αρκετά περιβάλλοντα ανάπτυξης εφαρμογών μέσω γραφικών διεπαφών. Παρακάτω, θα παρουσιαστούν μερικά από αυτά, καθώς και εφαρμογές χρήσης τους.

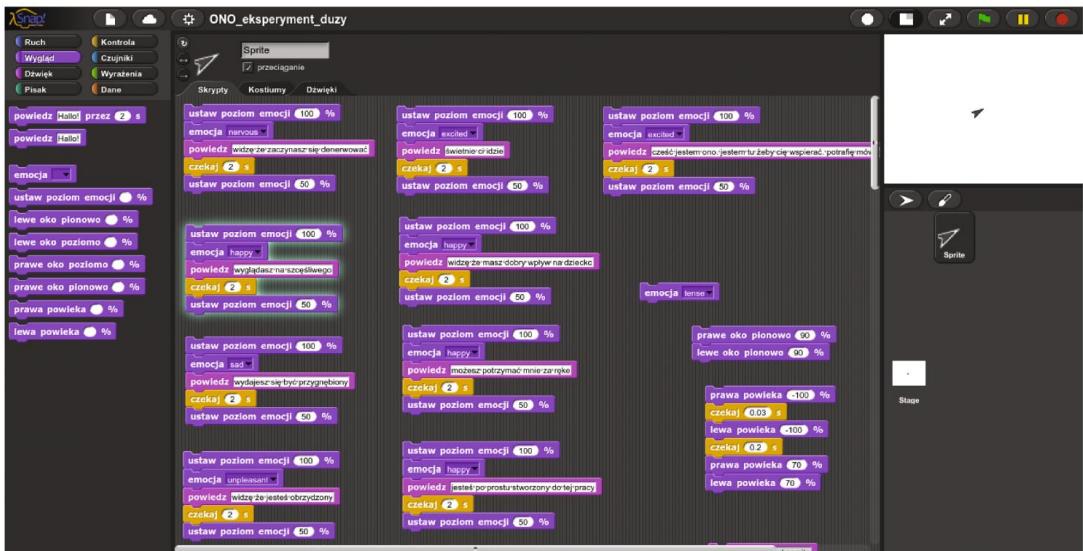
Στο [37] περιγράφεται το Robokol, που στοχεύει στον γρήγορο και εύκολο προγραμματισμό συσκευών IoT και ρομπότ. Συνδυάζει το Snap!<sup>3</sup>, μία δημοφιλής γλώσσα οπτικού προγραμματισμού, και το ROS. Προσφέρει ένα εύχρηστο και γρήγορο στην εκμάθηση περιβάλλον, ευκολία στον εντοπισμό νέων συσκευών, όπως επίσης και υποστήριξη πολλών συσκευών (υπολογιστές, tablet, smartphone) και λειτουργικών συστημάτων για την ανάπτυξη των εφαρμογών. Μία χρήση του συστήματος, ήταν ο προγραμματισμός ενός ρομπότ οπο και ενός μανικιού με αισθητήρες για θεραπεία αισθητηριακής ολοκλήρωσης σε παιδιά με αυτισμό. Το γραφικό περιβάλλον του Robokol φαίνεται στην εικόνα 3.4.

<sup>1</sup> <https://www.codecademy.com/articles/what-is-rest>

<sup>2</sup> <https://wsgi.readthedocs.io/en/latest/what.html>

<sup>3</sup> <https://snap.berkeley.edu/>

### 3.3 Γραφικά περιβάλλοντα ανάπτυξης ρομποτικών εφαρμογών



Εικόνα 3.4: Το γραφικό περιβάλλον Robokol [37]

Ένα ακόμα εργαλείο γραφικής ανάπτυξης εφαρμογών, αυτή τη φορά υλοποιημένο στην πλατφόρμα MiniBloq<sup>1</sup>, αναλύεται στο [38] και χρησιμοποιεί το Electric Ray ρομπότ. Προτείνεται ως ένα εισαγωγικό περιβάλλον προγραμματισμού ρομπότ, για μαθητές χωρίς ειδικές γνώσεις προγραμματισμού. Ως παράδειγμα χρήσης, ζητήθηκε από μαθητές λυκείου να επιλύσουν διάφορα προβλήματα, όπως παρακολούθηση τροχιάς και αποφυγή εμποδίων, τα οποία έφεραν όλοι εις πέρας, χάρη στην ευκολία που παρείχε το εργαλείο.

Στο [39] παρουσιάζεται το RAFCON, ένα εργαλείο για υλοποίηση περίπλοκων ρομποτικών εργασιών. Χρησιμοποιεί μηχανές πεπερασμένων καταστάσεων (Finite State Machines – FSM) μέσω μιας εύχρονης γραφικής διεπαφής. Τα μέλη του Institute of Robotics and Mechatronics του DLR<sup>2</sup> έλαβαν μέρος στο SpaceBotCamp 2016, και με τη βοήθεια του RAFCON, ήταν η μόνη ομάδα που ολοκλήρωσε με επιτυχία την αποστολή, η οποία περιείχε εξερεύνηση άγνωστου χώρου, εύρεση δύο αντικειμένων και επιστροφή στη βάση.

Τέλος, στα [40] και [41] χρησιμοποιήθηκαν τα CHERP<sup>3</sup> και PROTEAS αντίστοιχα, που δίνουν τη δυνατότητα προγραμματισμού είτε μέσω tangible programming είτε μέσω γραφικών διεπαφών. Και στις δύο έρευνες, παιδιά μικρής ηλικίας κλήθηκαν να ολοκληρώσουν απλές εργασίες με τα ρομπότ με τις δύο παραπάνω μεθόδους προγραμματισμού, με σκοπό τη σύγκρισή τους.

<sup>1</sup> <http://blog.minibloq.org/>

<sup>2</sup> <https://www.dlr.de/rm/en/desktopdefault.aspx/tabcid-8017>

<sup>3</sup> <https://sites.tufts.edu/devtech/research/past-projects/tangible-kindergarten/>

## Κεφάλαιο 4

# ΤΕΧΝΟΛΟΓΙΚΟ ΥΠΟΒΑΘΡΟ

Στη συνέχεια, θα παρουσιαστεί το βασικό τεχνολογικό υπόβαθρο, δηλαδή τα εργαλεία λογισμικού που χρησιμοποιήθηκαν στα πλαίσια της υλοποίησης του συστήματος της διπλωματικής εργασίας. Η ανάλυσή τους είναι απαραίτητη, για την καλύτερη και ευκολότερη κατανόηση της λειτουργίας του συστήματος. Αρχικά, στην Ενότητα 4.1 παρουσιάζεται το TurtleBot, το ρομπότ που χρησιμοποιεί το σύστημα. Στις Ενότητες 4.2 και 4.3 περιγράφονται το Robot Operating System και ορισμένες βιβλιοθήκες ρομποτικής των οποίων έγινε χρήση αντίστοιχα. Το περιβάλλον προσομοίωσης ρομποτικής Gazebo παρουσιάζεται στην Ενότητα 4.4. Τέλος, ο μεσολαβητής μηνυμάτων RabbitMQ και το εργαλείο ανάπτυξης εφαρμογών Node-RED αναλύονται στις Ενότητες 4.5 και 4.6 αντίστοιχα.

### 4.1 TurtleBot

Το TurtleBot είναι ένα χαμηλού κόστους ρομπότ προσωπικής χρήσης, με λογισμικό ανοιχτού κώδικα. Δημιουργήθηκε το Νοέμβριο του 2010 στο Willow Garage από τη Melonee Wise και τον Tully Foote. Το TurtleBot είναι σχεδιασμένο ώστε να είναι προσιτό στην αγορά και την συναρμολόγηση, χρησιμοποιώντας απλά υλικά για τα ρομποτικά του μέρη. Οι κύριες λειτουργίες του είναι η *Ταυτόχρονη Χαρτογράφηση και Εντοπισμός Θέσης* (Simultaneous Localization and Mapping - SLAM) και η *Πλοήγηση* (Navigation), που το καθιστούν ιδανικό για οικιακή χρήση. Το TurtleBot μπορεί μέσω του αλγορίθμου SLAM να δημιουργήσει ένα χάρτη του σπιτιού, και έπειτα να πλοηγείται σε αυτό. Επιπλέον, είναι δυνατός και ο τηλεχειρισμός του ρομπότ μέσω φορητού υπολογιστή, χειριστηρίου ή κινητού.



Εικόνα 4.1: Το TurtleBot2

Στο σύστημα της διπλωματικής εργασίας χρησιμοποιήθηκε το TurtleBot2, το οποίο φαίνεται στην εικόνα 4.1<sup>1</sup>. Αποτελείται από μία Yujin Kobuki βάση, μία μπαταρία 2200 mAh, έναν 3D αισθητήρα ASUS Xtion Pro, έναν φορητό υπολογιστή, μια βάση φόρτισης και υλικό για τη στήριξη πρόσθετων εξαρτημάτων. Δυστυχώς λόγω της πανδημίας, τα πειράματα με το ρομπότ περιορίστηκαν σε προσομοιωτή.

## 4.2 Robot Operating System

Το ROS είναι ένα μεταλειτουργικό σύστημα (meta-operating system) ανοιχτού κώδικα για ανάπτυξη ρομποτικών εφαρμογών. Πρωτοπαρουσιάστηκε το 2007 από τη Willow Garage, ενώ υλοποιήθηκε σε συνεργασία με την Πανεπιστήμιο του Στάνφορντ. Είναι μία συλλογή από εργαλεία, βιβλιοθήκες και συμβάσεις που στοχεύουν στην απλοποίηση του έργου δημιουργίας περίπλοκων ρομποτικών συμπεριφορών σε μία πληθώρα πλατφορμών.

Οι σχεδιαστικοί στόχοι με τους οποίους δημιουργήθηκε το ROS συνοψίζονται παρακάτω [16]:

- 1. Peer-to-peer:** Σε ένα σύστημα ROS οι διεργασίες είναι συνδεδεμένες με μία peer-to-peer τοπολογία, χωρίς να υπάρχει κάποιος κεντρικός διακομιστής δεδομένων.
- 2. Πολυγλωσσία:** Το ROS υποστηρίζει διάφορες γλώσσες προγραμματισμού για περισσότερη ευελιξία, όπως C++, Python, LISP, Octave κ.ά., με την περισσότερη υποστήριξη να παρέχεται στις C++ και Python.
- 3. Βασισμένο στα εργαλεία:** Αντί να χρησιμοποιήσει ένα μεμονωμένο περιβάλλον ανάπτυξης, το ROS χρησιμοποιεί πολλά μικρά εργαλεία για να χτίσει και να τρέξει τα διάφορα τμήματα, για καλύτερη διαχείριση της πολυπλοκότητας.
- 4. Λεπτή εξάρτηση:** Για να μπορούν τα έργα λογισμικού να επαναχρησιμοποιηθούν και σε άλλα έργα, το ROS παροτρύνει η ανάπτυξη να γίνεται ανεξάρτητη από αυτό.
- 5. Ανοιχτού κώδικα:** Ο πλήρης πηγαίος κώδικας του ROS είναι διαθέσιμος στο κοινό. Ως αποτέλεσμα, η κοινότητα του είναι πολυάριθμη και είναι ο κύριος λόγος που είναι τόσο διαδεδομένο.

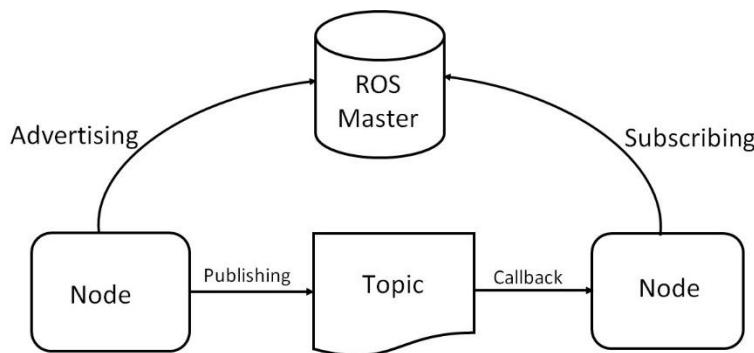
Η βασική λειτουργία του ROS εμπεριέχεται στο *Γράφο Υπολογισμού* (Computation Graph), ένα peer-to-peer δίκτυο διεργασιών του ROS, που επεξεργάζονται δεδομένα μαζί. Τα βασικά του στοιχεία είναι:

- **Nodes:** Οι κόμβοι είναι διεργασίες που υλοποιούν μια λειτουργία και έχουν γραφτεί με τη βοήθεια μιας βιβλιοθήκης ROS, όπως της roscpp για C++ ή της rosipy για Python. Για παράδειγμα, ένας κόμβος μπορεί να ελέγχει τους κινητήρες των τροχών και ένας άλλος να εκτελεί εντοπισμό θέσης.
- **Messages:** Οι κόμβοι επικοινωνούν μεταξύ τους μέσω μηνυμάτων. Το μήνυμα είναι μια δομή δεδομένων, όπως για παράδειγμα ακέραιος αριθμός, ακολουθία χαρακτήρων, πίνακας κ.ά.

<sup>1</sup> <https://www.generationrobots.com/en/401271-turtlebot-2-assembled-clearpathrobotics.html>

## ΚΕΦΑΛΑΙΟ 4. ΤΕΧΝΟΛΟΓΙΚΟ ΥΠΟΒΑΘΡΟ

- **Topics:** Τα μηνύματα διαβιβάζονται σύμφωνα με το publish-subscribe μοντέλο. Ένας κόμβος δημοσιεύει ένα μήνυμα σε ένα συγκεκριμένο topic. Όσοι κόμβοι θέλουν να λάβουν αυτό το μήνυμα, πρέπει να έχουν εγγραφεί στο topic αυτό.
- **Services:** Είναι ένας εναλλακτικός τρόπος επικοινωνίας μεταξύ των κόμβων που ακολουθεί τη λογική του *remote procedure call* (RPC). Σε αντίθεση με το publish-subscribe μοντέλο που υλοποιεί ασύγχρονη επικοινωνία, οι υπηρεσίες επιτρέπουν σύγχρονη επικοινωνία των κόμβων. Σε αυτήν την περίπτωση ένας κόμβος καλεί μια υπηρεσία για να λάβει ένα αποτέλεσμα που θα έχει υπολογιστεί σε έναν άλλον κόμβο.
- **Master:** Ο ROS Master κόμβος παρέχει τις υπηρεσίες ονοματοδοσίας και αναζήτησης στον υπόλοιπο Γράφο Υπολογισμού. Χωρίς αυτόν, οι άλλοι κόμβοι δε θα μπορούσαν να εντοπιστούν μεταξύ τους, να ανταλλάξουν μηνύματα ή να καλέσουν υπηρεσίες.
- **Parameter Server:** Οι κόμβοι χρησιμοποιούν το διακομιστή παραμέτρων για να απόθηκεύουν και να λαμβάνουν παραμέτρους κατά την εκτέλεσή τους.
- **Bags:** Είναι μία μορφή αρχείου για την αποθήκευση και αναπαραγωγή δεδομένων μηνυμάτων του ROS. Είναι ένας σημαντικός μηχανισμός για την αποθήκευση δεδομένων που είναι δύσκολο να μετρηθούν, αλλά είναι απαραίτητα για την ανάπτυξη αλγορίθμων.



Εικόνα 4.2: Επικοινωνία δύο κόμβων στο ROS<sup>1</sup>

Η κύρια λειτουργία του ROS επαυξάνεται από ένα σύνολο εργαλείων, που απλοποιούν και παρέχουν λύσεις σε πολλά κοινά προβλήματα κατά την ανάπτυξη ρομποτικών συστημάτων. Μερικά από αυτά είναι το RViz, που παρέχει τρισδιάστατη οπτικοποίηση διάφορων ρομπότ αλλά και δεδομένων αισθητήρων, το rqt που είναι βασισμένο στο Qt<sup>2</sup> και βοηθάει στη δημιουργία γραφικών διεπαφών για το ρομπότ και το roslaunch που επιτρέπει την ταυτόχρονη εκτέλεση πολλαπλών κόμβων και τη ρύθμιση παραμέτρων στον διακομιστή παραμέτρων.

<sup>1</sup> <https://robinrobotic.blogspot.com/2019/07/ros-topics-ros-tutorial.html>

<sup>2</sup> <https://www.qt.io/>

Ένα ακόμα πολύ χρήσιμο χαρακτηριστικό του ROS είναι ο μεγάλος αριθμός έτοιμων πακέτων λογισμικού και αλγορίθμων ρομποτικής που υπάρχουν, όπως πακέτα σχεδιασμού κίνησης, μηχανικής μάθησης και επεξεργασίας εικόνας. Αυτό καθίσταται δυνατό από την πολύ καλή ενσωμάτωση του ROS με άλλες βιβλιοθήκες, όπως την OpenCV<sup>1</sup> για μηχανική όραση, την MoveIt<sup>2</sup> για το σχεδιασμό κίνησης και την PCL<sup>3</sup> για τη διαχείριση και την επεξεργασία τρισδιάστατων δεδομένων και εικόνων βάθους.

## 4.3 Βιβλιοθήκες ρομποτικής

Κατά την ανάπτυξη του συστήματος της διπλωματικής εργασίας, έγινε ξεκάθαρο ότι το ρομπότ θα πρέπει να εκτελεί κάποιες πολύ βασικές λειτουργίες. Αυτές οι λειτουργίες είναι η **ταυτόχρονη χαρτογράφηση και εντοπισμός θέσης – SLAM**, ο **εντοπισμός θέσης – Localization** και η **πλοήγηση – Navigation**. Σε αυτήν την παράγραφο θα αναλυθούν οι αλγόριθμοι που χρησιμοποιήθηκαν για την υλοποίηση αυτών των λειτουργιών, οι οποίοι παρέχονται από το ROS.

### 4.3.1 GMapping

Το GMapping<sup>4</sup> είναι ένα υψηλής απόδοσης Rao-Blackwellized φίλτρο σωματιδίων (particle filter) που κατασκευάζει χάρτες κάλυψης πλέγματος (occupancy grid maps – OGMs) από δεδομένα αισθητήρα λέιζερ.

Τα Rao-Blackwellized φίλτρα σωματιδίων αποτελούν μια αποτελεσματική μέθοδος για την επίλυση του SLAM. Στη προσέγγιση του GMapping, κάθε σωματίδιο του φίλτρου σωματιδίων διαθέτει ένα ξεχωριστό χάρτη του περιβάλλοντος. Ένα σημαντικό πρόβλημα είναι η μείωση του αριθμού των σωματιδίων, που εν προκειμένω αντιμετωπίζεται με προσαρμοζόμενες τεχνικές. Για τη μείωση της αβεβαιότητας σχετικά με τη θέση και το προσανατολισμό του ρομπότ κατά το βήμα πρόβλεψης του φίλτρου, υπολογίζει μία ακριβής κατανομή που πέρα από τη κίνηση του ρομπότ, λαμβάνει υπ' όψιν και την πιο πρόσφατη παρατήρηση. Παράλληλα, εφαρμόζει μια τεχνική για επιλεκτική επαναδειγματοληψία που μειώνει δραστικά το πρόβλημα της εξάντλησης των σωματιδίων.

### 4.3.2 AMCL

Το Adaptive Monte Carlo Localization<sup>5</sup> (AMCL) είναι ένα σύστημα πιθανοτικού εντοπισμού θέσης ενός ρομπότ, που κινείται σε δισδιάστατο χώρο. Υλοποιεί τη προσαρμοζόμενη (adaptive ή KLD-sampling) μέθοδο του Monte Carlo localization [17], που χρησιμοποιεί ένα φίλτρο σωματιδίων για να ανιχνεύει τη θέση και τον προσανατολισμό του ρομπότ σε ένα γνωστό χάρτη.

Για την αποφυγή της εξάντλησης των δειγμάτων στον MCL, πρέπει να επιλεχθούν μεγάλα σύνολα δειγμάτων, που ισοδυναμεί με σπατάλη υπολογιστικών πόρων. Η δειγματοληψία KLD είναι μία παραλλαγή του MCL που προσαρμόζει τον αριθμό των σωματιδίων

<sup>1</sup> <https://opencv.org/>

<sup>2</sup> <https://moveit.ros.org/>

<sup>3</sup> <https://pointclouds.org/>

<sup>4</sup> <https://openslam-org.github.io/gmapping.html>

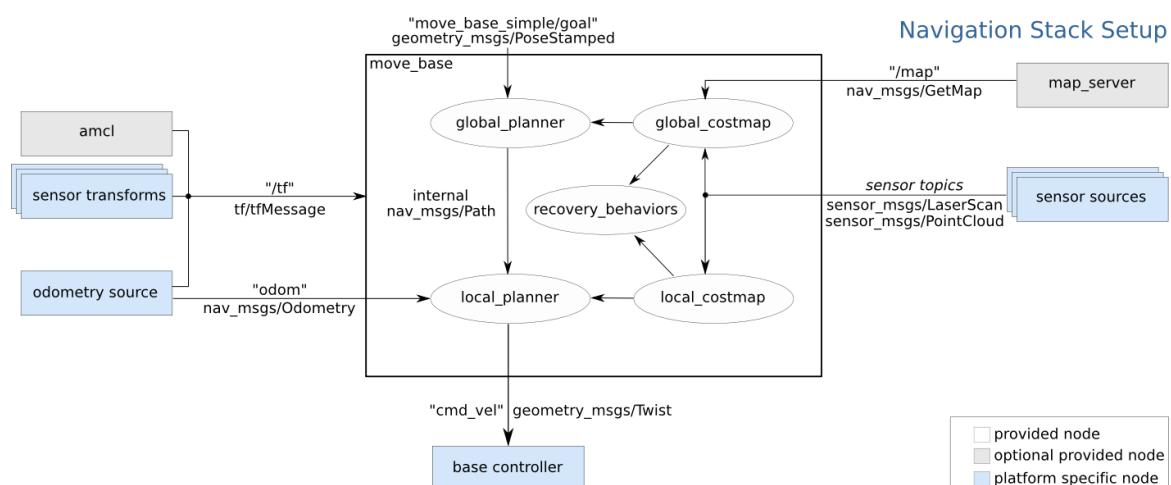
<sup>5</sup> <http://wiki.ros.org/amcl>

## ΚΕΦΑΛΑΙΟ 4. ΤΕΧΝΟΛΟΓΙΚΟ ΥΠΟΒΑΘΡΟ

σε σχέση με το χρόνο. Σε κάθε επανάληψη του φίλτρου σωματιδίων, προσδιορίζει τον αριθμό των σωματιδίων με βάση ένα στατιστικό φράγμα για την ποιότητα της βασισμένης σε δείγματα προσέγγισης. Έχει αποδειχθεί, ότι στα πλαίσια του εντοπισμού θέσης ενός ρομπότ, έχει πάντα καλύτερη απόδοση από τον αλγόριθμο MCL για σταθερά μεγέθη του συνόλου δειγμάτων. Το πλεονέκτημα της συγκεκριμένης τεχνικής είναι περισσότερο σημαντικό στο συνδυασμό προβλημάτων καθολικού εντοπισμού θέσης και παρακολούθησης.

### 4.3.3 ROS Navigation Stack

Το navigation stack χρησιμοποιείται για την πλοήγηση ενός ρομπότ στο δισδιάστατο χώρο. Η λογική της λειτουργίας του είναι αρκετά απλή. Λαμβάνει πληροφορίες από την οδομετρία (odometry) για την εκτίμηση της θέσης και του προσανατολισμού του ρομπότ, και από αισθητήρες για την αποφυγή εμποδίων στο χώρο. Σύμφωνα με αυτές τις πληροφορίες, υπολογίζει ασφαλείς ταχύτητες κίνησης για το ρομπότ. Το γενικό διάγραμμά του φαίνεται στην εικόνα 4.3<sup>1</sup>.



Εικόνα 4.3: Γενική δομή του navigation stack

Το navigation stack χρησιμοποιεί δύο χάρτες κόστους (costmaps) για να αποθηκεύσει την πληροφορία των εμποδίων. Ο ένας χάρτης κόστους χρησιμοποιείται για την καθολική (global) πλοήγηση, δηλαδή για τη δημιουργία ενός μονοπατιού σε έναν μακρινό στόχο, ενώ ο άλλος για την τοπική πλοήγηση, για την δημιουργία μονοπατιού κοντινών αποστάσεων και αποφυγή εμποδίων. Τις πληροφορίες αυτών των χαρτών χρησιμοποιούν οι δύο planners, για τον υπολογισμό των εντολών ταχύτητας, που στέλνουν στη συνέχεια στον ελεγκτή βάσης του ρομπότ. Οι λειτουργίες αυτές εμπεριέχονται στον κόμβο `move_base`.

Παράλληλα, εάν το ρομπότ κολλήσει σε κάποιο σημείο, θα γίνουν κάποιες προσπάθειες ανάκτησης. Αρχικά, σβήνει τα εμπόδια από το χάρτη, που βρίσκονται εκτός μίας προκαθορισμένης από το χρήστη περιοχής. Εάν αυτό δεν επιλύσει το πρόβλημα, τότε το ρομπότ θα κάνει μια περιστροφή για να αντιληφθεί καλύτερο το χώρο. Αν το πρόβλημα παραμένει, τότε θα σβήσει όλα τα εμπόδια εκτός της περιοχής που μπορεί να περιστραφεί, και θα εκτελέσει μία ακόμα περιστροφή. Αν αποτύχει και αυτό, το ρομπότ θεωρεί τον στόχο ακατόρθωτο και ειδοποιεί το χρήστη ότι τερμάτισε τη διαδικασία.

<sup>1</sup> <http://wiki.ros.org/navigation/Tutorials/RobotSetup>

## 4.4 Gazebo

Συχνά, ο έλεγχος ενός αλγορίθμου απευθείας στο ρομπότ είναι χρονοβόρος και μπορεί να προκαλέσει ζημιές. Για αυτό το λόγο, συνηθίζεται η χρήση προσομοιωτών για τον αρχικό έλεγχο και τη βελτιστοποίηση του αλγορίθμου, πριν προχωρήσουμε στο κανονικό ρομπότ.

Το Gazebo<sup>1</sup> είναι ένας ανοιχτού κώδικα προσομοιωτής ρομποτικής σε τρισδιάστατο περιβάλλον. Η ανάπτυξή του ξεκίνησε το 2002 στο Πανεπιστήμιο της Νότιας Καλιφόρνια, από το Δρ. Andrew Howard και το μαθητή του Nate Koenig. Ο αρχικός στόχος ήταν η δημιουργία ενός υψηλής ανάλυσης προσομοιωτή, που είχε τη δυνατότητα προσομοίωσης ρομπότ σε εξωτερικά περιβάλλοντα, υπό διάφορες συνθήκες.

Ένα πολύ θετικό χαρακτηριστικό του Gazebo ότι μπορεί να ενσωματωθεί με το ROS παρέχοντας πλήρη συμβατότητα, απλά με την εγκατάσταση ενός πακέτου ROS. Παράλληλα, προσομοιώνει και τη λειτουργία αισθητήρων, όπως 2D/3D κάμερες, αισθητήρες απόστασης λέιζερ κ.ά., ενώ υποστηρίζει διάφορες μηχανές φυσικής (physics engines), με κύρια επιλογή την ODE<sup>2</sup>. Τέλος, παρέχει μια πληθώρα μοντέλων για ρομπότ και άλλα αντικείμενα συμπεριλαμβανομένου και του TurtleBot.

## 4.5 RabbitMQ

Το RabbitMQ είναι ένας από τους πιο δημοφιλής μεσολαβητές μηνυμάτων (message brokers) ανοιχτού κώδικα. Είναι αρκετά ελαφρύ και εύκολο στην ανάπτυξη, ακόμα και σε κατανευμένες διατάξεις, ενώ είναι συμβατό με πολλά λειτουργικά συστήματα.

Το RabbitMQ υποστηρίζει αρκετά πρωτόκολλα μηνυμάτων, είτε άμεσα είτε μέσω plugin. Τα πρωτόκολλα αυτά αναφέρονται συνοπτικά παρακάτω:

- **AMQP 0-9-1:** Το κύριο πρωτόκολλο που χρησιμοποιεί ο μεσολαβητής, το οποίο αναλύθηκε στην υποενότητα 2.1.1.5.
- **MQTT:** Το MQTT υποστηρίζεται από το μεσολαβητή μέσω plugin. Η λειτουργία του αναλύθηκε στην υποενότητα 2.1.1.5.
- **STOMP<sup>3</sup>:** Πρόκειται για ένα πρωτόκολλο βασισμένο στο κείμενο, που δίνει έμφαση στην απλότητα και είναι πολύ εύκολο στην υλοποίηση. Υποστηρίζεται μέσω plugin.
- **AMQP 1.0:** Παρά την ονομασία, διαφέρει πολύ από το AMQP 0-9-1. Είναι αρκετά πιο περίπλοκο και υπάρχουν λιγότερες υλοποιήσεις. Υποστηρίζεται μέσω plugin.
- **HTTP και WebSockets:** Παρόλο που το HTTP δεν είναι πρωτόκολλο μηνυμάτων, το RabbitMQ μπορεί να μεταβιβάσει μηνύματα μέσω HTTP, μέσω plugin που επιτρέπουν την αποστολή μηνυμάτων STOMP ή MQTT με χρήση WebSockets.

<sup>1</sup> <http://gazebosim.org/>

<sup>2</sup> <http://www.ode.org/>

<sup>3</sup> <http://stomp.github.io/>

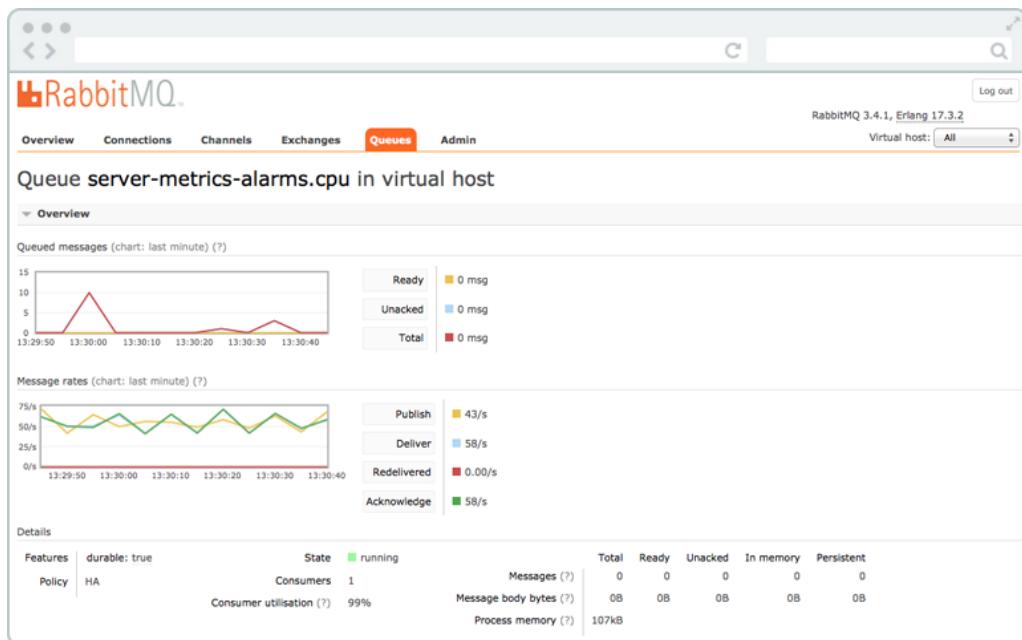
## ΚΕΦΑΛΑΙΟ 4. ΤΕΧΝΟΛΟΓΙΚΟ ΥΠΟΒΑΘΡΟ

Η λογική με την οποία γίνεται η επικοινωνία μεταξύ των πελατών ακολουθεί τη λογική του AMQP, μιας και αυτό είναι το πρωτόκολλο που χρησιμοποιεί χυρίως το RabbitMQ. Παρ' όλο που το κύριο μοντέλο επικοινωνίας είναι το publish-subscribe, υπάρχει και η δυνατότητα επικοινωνίας μέσω remote procedure call.

Οι συνδέσεις (connections) μεταξύ των εφαρμογών και του μεσολαβητή είναι συνήθως μακράς διάρκειας. Το πρωτόκολλο που χρησιμοποιείται είναι το TCP, για εξασφάλιση της αξιοπιστίας της παράδοσης, ενώ γίνεται και χρήση αυθεντικοποίησης. Μερικές εφαρμογές ενδέχεται να χρειάζονται περισσότερες από μία συνδέσεις με το μεσολαβητή. Ωστόσο, είναι ανεπιθύμητο να διατηρούμε πολλές συνδέσεις TCP ανοιχτές ταυτόχρονα, διότι αυτό καταναλώνει πολλούς πόρους συστήματος. Για αυτό το λόγο, οι συνδέσεις είναι πολυπλεγμένες με κανάλια (channels), που ουσιαστικά είναι εικονικές συνδέσεις μέσα στην TCP σύνδεση. Κάθε κανάλι είναι ανεξάρτητο από τα υπόλοιπα, και όλες οι λειτουργίες διεκπεραιώνονται μέσω των καναλιών.

Παράλληλα, παρέχεται η μέθοδος του virtual hosting για το διαχωρισμό των εφαρμογών στον ίδιο μεσολαβητή. Με αυτόν τον τρόπο, δημιουργούνται απομονωμένα περιβάλλοντα που το καθένα μπορεί να περιέχει συνδέσεις, exchanges, ουρές, δικαιώματα χρηστών κ.ά. Ως αποτέλεσμα, υπάρχει λογική ομαδοποίηση και διαχωρισμός των πόρων του συστήματος.

Τέλος, το RabbitMQ παρέχει μια διεπαφή δικτύου για τη διαχείριση και παρακολούθηση του διακομιστή. Επιτρέπει τον έλεγχο των συνδέσεων, καναλιών, ουρών, exchanges, χρηστών και δικαιωμάτων χρηστών, ενώ δίνει αρκετές πληροφορίες για το σύστημα, όπως το ρυθμό των μηνυμάτων, ποιες θύρες χρησιμοποιούνται ή τη χρήση πόρων.



Εικόνα 4.4: Περιβάλλον διαχείρισης του RabbitMQ<sup>1</sup>

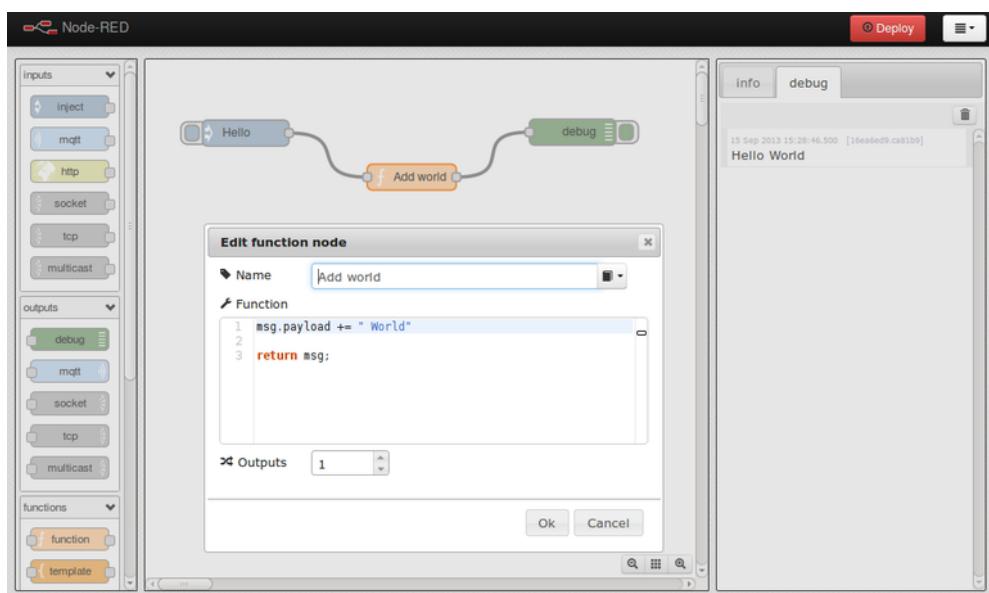
<sup>1</sup> [https://www.cloudamqp.com/blog/part3-rabbitmq-for-beginners\\_the-management-interface.html](https://www.cloudamqp.com/blog/part3-rabbitmq-for-beginners_the-management-interface.html)

## 4.6 Node-RED

Το Node-RED είναι ένα αναπτυξιακό εργαλείο γραφικού προγραμματισμού βασισμένο σε ροές (flow-based), για τη σύνδεση συσκευών υλικού, διεπαφών προγραμματισμού εφαρμογών (APIs) και διαδικτυακών υπηρεσιών ως μέρος του IoT. Πρωτοεμφανίστηκε το 2013 ως ένα εγχείρημα του Nick O'Leary και του Dave Conway-Jones της IBM, για τη διαχείριση και οπτικοποίηση αντιστοιχίσεων μεταξύ θεμάτων MQTT.

Η ανάπτυξη μιας εφαρμογής γίνεται με τη βοήθεια κόμβων (nodes), όπου ο καθένας επιτελεί μία συγκεκριμένη λειτουργία. Κάθε κόμβος δέχεται κάποια δεδομένα, τα επεξεργάζεται και στη συνέχεια τα μεταφέρει σε άλλον κόμβο. Το δίκτυο είναι υπεύθυνο για τη ροή των δεδομένων μεταξύ των κόμβων. Η επικοινωνία μεταξύ των κόμβων είναι ασύγχρονη και οδηγούμενη από τα γεγονότα (event-driven), όπου ένα γεγονός μπορεί να είναι η λήψη μιας τιμής από κάποιο αισθητήρα. Το Node-RED χρησιμοποιεί το Node.js ως περιβάλλον εκτέλεσης (runtime environment), εκμεταλλεύμενο το οδηγούμενο από γεγονότα, χωρίς μπλοκάρισμα (non-blocking) μοντέλο του. Ως εκ τούτου είναι ιδανικό για να τρέχει σε φτηνές συσκευές αλλά και στο cloud.

Το Node-RED παρέχει ένα γραφικό περιβάλλον που τρέχει σε φυλλομετρητή, όπου ο χρήστης τοποθετεί και ενώνει τους κόμβους για τη δημιουργία της εφαρμογής του, όπως φαίνεται στην εικόνα 4.5<sup>1</sup>.



Εικόνα 4.5: Γραφικό περιβάλλον ανάπτυξης του Node-RED

Πέρα από τους κόμβους που παρέχονται από το εργαλείο, υπάρχει μία τεράστια ποικιλία πακέτων που έχουν δημιουργηθεί από την κοινότητα και είναι δωρεάν για λήψη. Εάν, παρ' όλα αυτά δεν υπάρχει διαθέσιμος κάποιος κόμβος για τη λειτουργία που επιθυμούμε, μπορούμε να υλοποιηθεί εύκολα μέσω JavaScript. Επιπλέον, οι ροές που δημιουργούνται στο Node-RED αποθηκεύονται σε αρχεία JSON<sup>2</sup> (JavaScript Object Notation), τα οποία μπορούν εύκολα να διαμοιραστούν μεταξύ χρηστών.

<sup>1</sup> <https://www.techrepublic.com/article/node-red/>

<sup>2</sup> <https://www.json.org/json-en.html>

## Κεφάλαιο 5

# ΥΛΟΠΟΙΗΣΗ

Στο κεφάλαιο αυτό θα παρουσιαστεί αναλυτικά το σύστημα που υλοποιήθηκε στα πλαίσια της διπλωματικής εργασίας. Στην Ενότητα 4.1 δίνεται μια γενική περιγραφή του συστήματος και της αρχιτεκτονικής του. Το τμήμα που βρίσκεται στο τοπικό δίκτυο και επικοινωνεί άμεσα με το ρομπότ περιγράφεται στην Ενότητα 4.2, ενώ η λειτουργία του μεσολαβητή παρουσιάζεται στην Ενότητα 4.3. Τέλος, στην Ενότητα 4.4 αναλύεται το τμήμα του συστήματος που χρησιμοποιεί το Node-RED και επικοινωνεί απομακρυσμένα με το ρομπότ.

### 5.1 Γενική περιγραφή του συστήματος

Όπως έχουμε ήδη εξηγήσει, ο σκοπός του συστήματος είναι η ενσωμάτωση ενός ρομποτικού συστήματος στο IoT, δίνοντας τη δυνατότητα του απομακρυσμένου ελέγχου του ρομπότ στο χρήστη, ενώ ταυτόχρονα διευκολύνοντας τη διαδικασία ανάπτυξης των εφαρμογών του. Αρχικά, θα αναλυθούν οι προδιαγραφές που πρέπει να πληροί το σύστημα. Στη συνέχεια, θα παρουσιαστεί η αρχιτεκτονική του συστήματος, και θα δοθεί μια σύντομη περιγραφή των επιμέρους τμημάτων.

#### 5.1.1 Προδιαγραφές

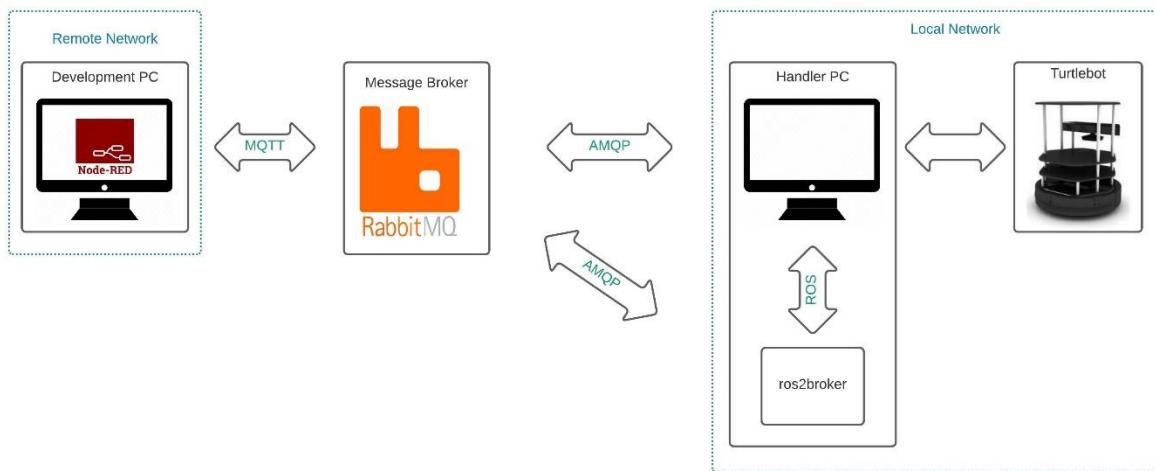
Οι προδιαγραφές του συστήματος προκύπτουν από τις λειτουργίες που μπορεί να επιτελεί το ρομπότ TurtleBot2 και θέλουμε να παρέχονται στο χρήστη, αλλά και από τη γενική φιλοσοφία του συστήματος.

- Απομακρυσμένος έλεγχος:** Στον υπολογιστή του χρήστη θα εκτελείται μόνο το Node-RED, το οποίο θα στέλνει τις εντολές μέσω του μεσολαβητή στο ρομπότ.
- Χαρτογράφηση:** Το ρομπότ θα πρέπει να έχει τη δυνατότητα της χαρτογράφησης ενός άγνωστου χώρου (SLAM) και της αποθήκευσης του χάρτη που θα προκύπτει.
- Εντοπισμός Θέσης:** Έχοντας το χάρτη ενός χώρου, το ρομπότ θα πρέπει να εντοπίζει τη θέση του σε αυτόν (Localization).
- Πλοήγηση:** Το ρομπότ θα πρέπει να δέχεται επιθυμητούς στόχους θέσης σε γνωστό χάρτη από το χρήστη, να σχεδιάζει τροχιά για να φτάσει στους στόχους, και να την ακολουθεί (Navigation).
- Τηλεχειρισμός:** Ο χρήστης θα πρέπει να έχει τη δυνατότητα του τηλεχειρισμού του ρομπότ.
- Πολλαπλά ρομπότ:** Θα πρέπει να παρέχεται υποστήριξη και για εφαρμογές πολλαπλών ρομπότ.

Η ανάπτυξη του συστήματος θα δομηθεί βάσει των παραπάνω προδιαγραφών, τόσο στη μεριά του χρήστη, όπου θα πρέπει να υπάρχει εύκολη πρόσβαση σε αυτές τις λειτουργίες, όσο και στη μεριά του ρομπότ, όπου θα πρέπει να υλοποιηθούν οι λειτουργίες αυτές.

### 5.1.2 Αρχιτεκτονική

Η αρχιτεκτονική του συστήματος στηρίζεται σε ένα μοντέλο client-server τριών επιπέδων, όπου η εφαρμογή και το ρομπότ αποτελούν τους clients, και ο μεσολαβητής μηνυμάτων παίρνει το ρόλο του server. Με αυτόν τον τρόπο, το σύστημα είναι κλιμακούμενο και μπορεί να υποστηρίξει εύκολα πολλούς χρήστες, σε σύγκριση με την περίπτωση που το ρομπότ λειτουργεί ως server. Η αρχιτεκτονική του συστήματος παρουσιάζεται στην ακόλουθη εικόνα.



Εικόνα 5.1: Η αρχιτεκτονική του συστήματος

Όπως είναι εμφανές το σύστημα αποτελείται από τρία επιμέρους τμήματα. Στη μεριά του ρομπότ έχουμε τον υπολογιστή που λειτουργεί ως χειριστής (handler), μέσω του οποίου γίνεται η επικοινωνία μεταξύ του ρομπότ και του RabbitMQ. Το RabbitMQ είναι υπεύθυνο για την επίτευξη της επικοινωνίας μεταξύ του Node-RED και του handler. Τέλος, στο Node-RED γίνεται η ανάπτυξη των εφαρμογών από το χρήστη. Πιο αναλυτικά έχουμε:

- Τοπικό τμήμα:** Στο τοπικό δίκτυο που είναι συνδεδεμένο το ρομπότ υπάρχει και ο υπολογιστής που λειτουργεί ως handler. Η λειτουργίας που επιτελεί είναι δύο. Αφ' ενός μέσω αυτού επιτυγχάνεται η μεταφορά των πληροφοριών μεταξύ του RabbitMQ και του ρομπότ. Για τη μεταφορά δεδομένων του ROS, όπως επιθυμητό στόχο ή εντολή ταχύτητας από τον τηλεχειρισμό, μεσολαβεί το ros2broker<sup>11</sup>. Με τη βοήθειά του, μετατρέπονται τα ROS δεδομένα σε μηνύματα AMQP και αντίστροφα, ώστε να μπορούν να μεταφερθούν από το μεσολαβητή. Αφ' ετέρου, είναι υπεύθυνος για την αρχικοποίηση και τον τερματισμό της λειτουργίας του ρομπότ, ανάλογα με τις εντολές που λαμβάνει από το χρήστη, αλλά και για την επεξεργασία ορισμένων δεδομένων είτε από το ρομπότ είτε από το Node-RED.

<sup>11</sup> <https://github.com/klpanagi/ros2broker>

- **Μεσολαβητής μηνυμάτων:** Όπως εξηγήσαμε ήδη, ο ρόλος του μεσολαβητή RabbitMQ είναι να δρα ως server, μεταφέροντας τα δεδομένα από το handler υπολογιστή στο Node-RED και αντίστροφα. Από τη μεριά του handler τα δεδομένα είναι σε μορφή AMQP, ενώ από τη μεριά του Node-RED σε μορφή MQTT.
- **Απομακρυσμένο τμήμα:** Στον απομακρυσμένο υπολογιστή του χρήστη, γίνεται η ανάπτυξη των εφαρμογών με τη βοήθεια του Node-RED. Οι λειτουργίες που μπορεί να χρησιμοποιήσει ο χρήστης στην εφαρμογή του είναι αυτές που προκύπτουν από τις προδιαγραφές που παρουσιάστηκαν παραπάνω, δηλαδή SLAM, εντοπισμό θέσης, πλοήγηση και τηλεχειρισμό, ενώ για όλες αυτές τις λειτουργίες εκτός του SLAM, υποστηρίζονται και πολλαπλά ρομπότ. Επιπλέον, για να είναι δυνατές οι λειτουργίες αυτές, παρέχεται στο χρήστη μία ζωντανή ροή του χάρτη του χώρου μαζί με την τρέχουσα τοποθεσία του ρομπότ.

## 5.2 Τοπικό τμήμα του συστήματος

Στην Ενότητα αυτή θα αναλυθούν οι λειτουργίες που παρέχει το τμήμα του συστήματος που βρίσκεται στο τοπικό δίκτυο και επικοινωνεί άμεσα με το ρομπότ. Τα κύρια μέρη του είναι τα ROS πακέτα που υλοποιούν τις επιθυμητές λειτουργίες, ο handler που είναι υπεύθυνος για την αρχικοποίηση και τον τερματισμό όλων των λειτουργιών και το ros2broker που καθιστά δυνατή τη μεταφορά των ROS δεδομένων μέσω του RabbitMQ.

### 5.2.1 Πακέτα του ROS

Ως μέρος του συστήματος υλοποιήθηκαν δύο πακέτα ROS, ένα για τη χρήση ενός ρομπότ και ένα για τη χρήση πολλαπλών ρομπότ. Στη συνέχεια, θα παρουσιαστούν οι λειτουργίες των δύο αυτών πακέτων, για να κατανοήσει καλύτερα ο αναγνώστης αυτό το τμήμα του συστήματος.

#### 5.2.1.1 Λειτουργία ενός ρομπότ

Το πρώτο πακέτο είναι υπεύθυνο για την παροχή όλων των λειτουργιών που πρέπει να προσφέρονται στο χρήστη και αναφέρθηκαν προηγουμένως, όταν χρησιμοποιείται ένα ρομπότ. Αυτές είναι το SLAM, ο εντοπισμός θέσης και η πλοήγηση. Οι λειτουργίες αυτές αναλύονται παρακάτω.

α) *SLAM*:

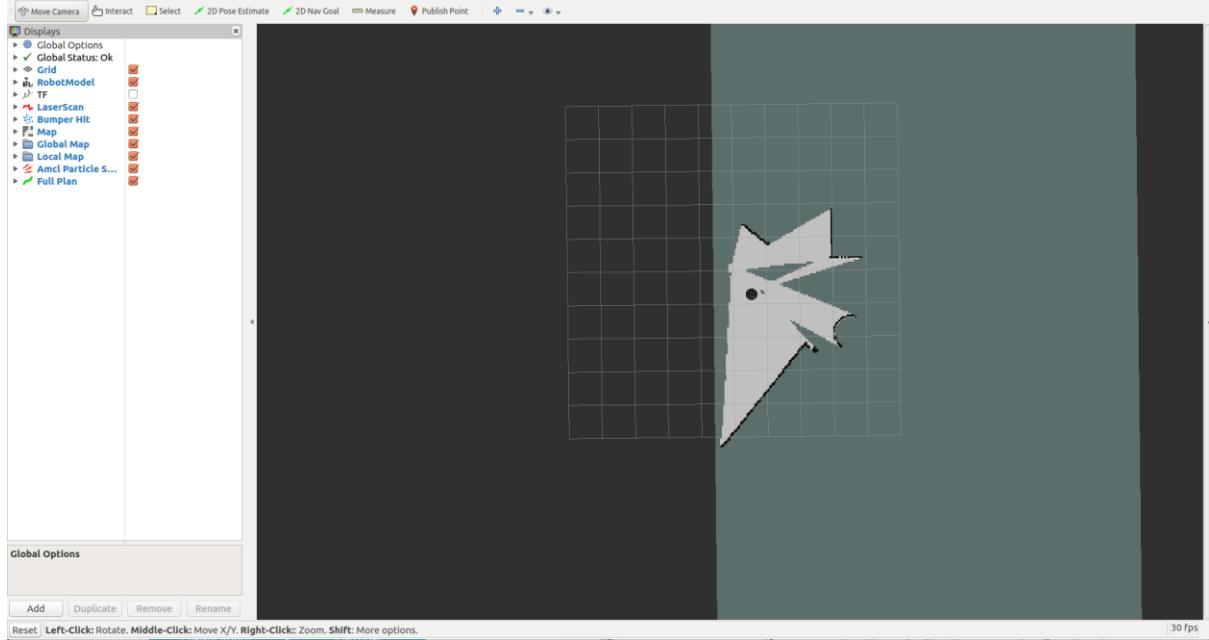
Όπως περιγράφηκε και στην υποενότητα 4.3.1, ο αλγόριθμος SLAM που χρησιμοποιείται είναι ο GMapping. Για το σχηματισμό της εικόνας του τρέχοντα χάρτη που έχει υπολογίσει ο αλγόριθμος, μαζί με τη θέση και τον προσανατολισμό του ρομπότ, αλλά και τα σημεία που έχει σαρώσει το λέιζερ<sup>1</sup>, είναι υπεύθυνος ο κόμβος SLAM\_map. Για τον υπολογισμό της θέσης και του προσανατολισμού του ρομπότ, χρησιμοποιείται ο κόμβος robot\_pose\_publisher<sup>2</sup>, που είναι μέρος ενός έτοιμου πακέτου του ROS. Μέσω αυτών των τριών κόμβων λοιπόν, υλοποιείται η λειτουργία του SLAM.

<sup>1</sup> Για τη δημιουργία δεδομένων σάρωσης λέιζερ (laser scan), εφόσον το TurtleBot2 δε διαθέτει κάποιο αισθητήρα λέιζερ όπως π.χ. Lidar, χρησιμοποιείται η κάμερα βάθους.

<sup>2</sup> [http://wiki.ros.org/robot\\_pose\\_publisher](http://wiki.ros.org/robot_pose_publisher)

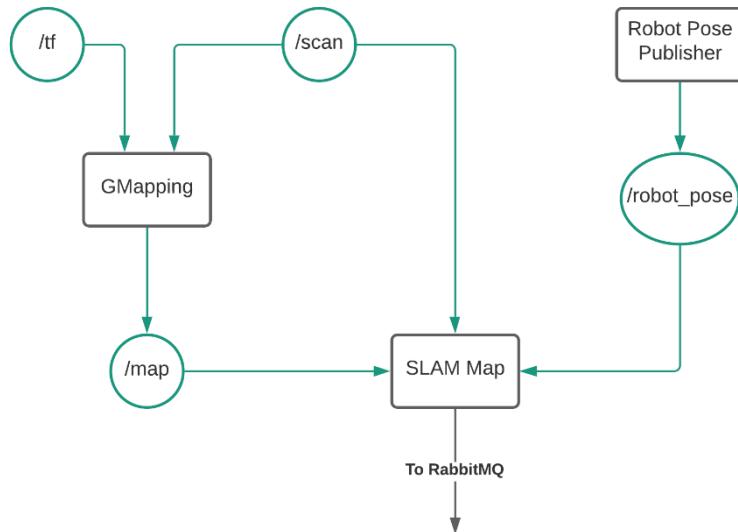
## 5.2 Τοπικό τμήμα του συστήματος

Στην εικόνα 5.2 φαίνεται ένα στιγμιότυπο από το RViz κατά την εκτέλεση του αλγορίθμου του SLAM.



Εικόνα 5.2: Χάρτης από την εκτέλεση του SLAM

Με μαύρο χρώμα απεικονίζονται τα εμπόδια που έχουν εντοπιστεί από τη σάρωση του λέιζερ, ενώ με άσπρο χρώμα απεικονίζεται ο ελεύθερος χώρος. Η τρέχουσα σάρωση του λέιζερ φαίνεται με έντονο άσπρο χρώμα, που εν προκειμένω ταυτίζεται με τα εμπόδια. Αυτές οι τρεις πληροφορίες χρησιμοποιούνται και από τον κόμβο SLAM\_map για το σχηματισμό της εικόνας που θα σταλθεί στο Node-RED. Στο διάγραμμα της εικόνας 5.3 φαίνονται τα topics που είναι εγγεγραμμένοι ή δημοσιεύονται οι κόμβοι.



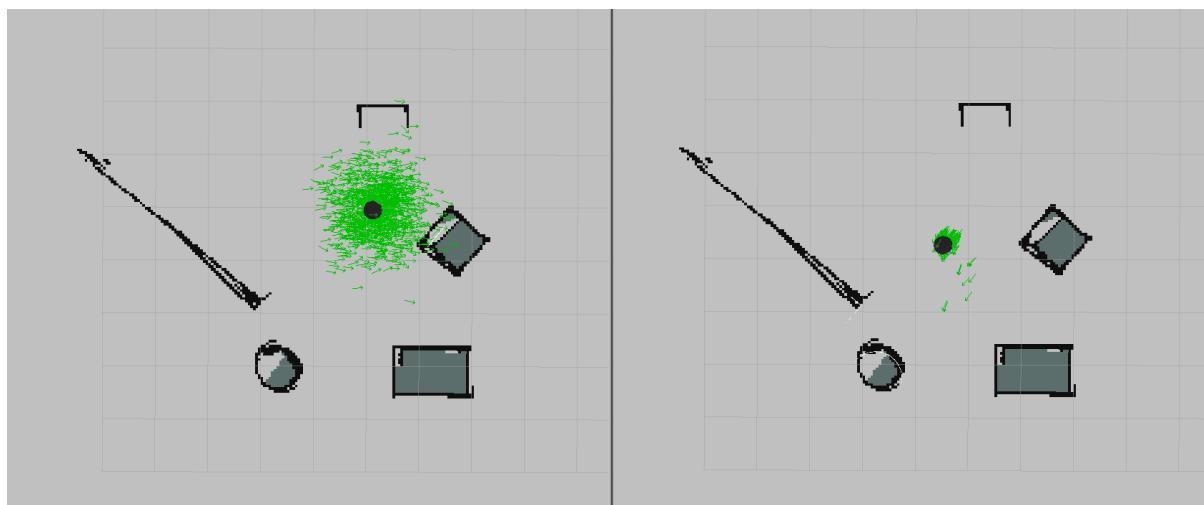
Εικόνα 5.3: Διάγραμμα μηνυμάτων για το SLAM

## ΚΕΦΑΛΑΙΟ 5. ΥΛΟΠΟΙΗΣΗ

Με πράσινο χρώμα απεικονίζονται τα ROS topics, καθώς και ποιοι κόμβοι είναι εγγεγραμμένοι ή δημοσιεύουν σε αυτά. Ο αλγόριθμος GMapping δέχεται μηνύματα από το /tf σχετικά με ορισμένους μετασχηματισμούς που χρειάζεται, και από το /scan για την τρέχουσα σάρωση του λέιζερ. Μέσω αυτών, σχηματίζει το χάρτη και το δημοσιεύει στο /map. Παράλληλα, ο κόμβος robot\_pose\_publisher υπολογίζει τη θέση και τον προσανατολισμό του ρομπότ και τα δημοσιεύει στο /robot\_pose. Τέλος, ο κόμβος SLAM\_map δέχεται τη τρέχουσα σάρωση του λέιζερ από το /scan, τη θέση του ρομπότ από το /robot\_pose και το χάρτη από το /map και με αυτές τις πληροφορίες σχηματίζει την εικόνα. Την εικόνα αυτή τη στέλνει στο RabbitMQ για να μεταβιβαστεί στο Node-RED.

### β) Εντοπισμός θέσης και πλοήγηση:

Ο εντοπισμός θέσης και η πλοήγηση έχουν υλοποιηθεί μαζί και χρησιμοποιούνται ο αλγόριθμος AMCL και το ROS Navigation Stack αντίστοιχα. Η λειτουργία του αλγορίθμου AMCL για τον εντοπισμό θέσης φαίνεται στην εικόνα 5.4.

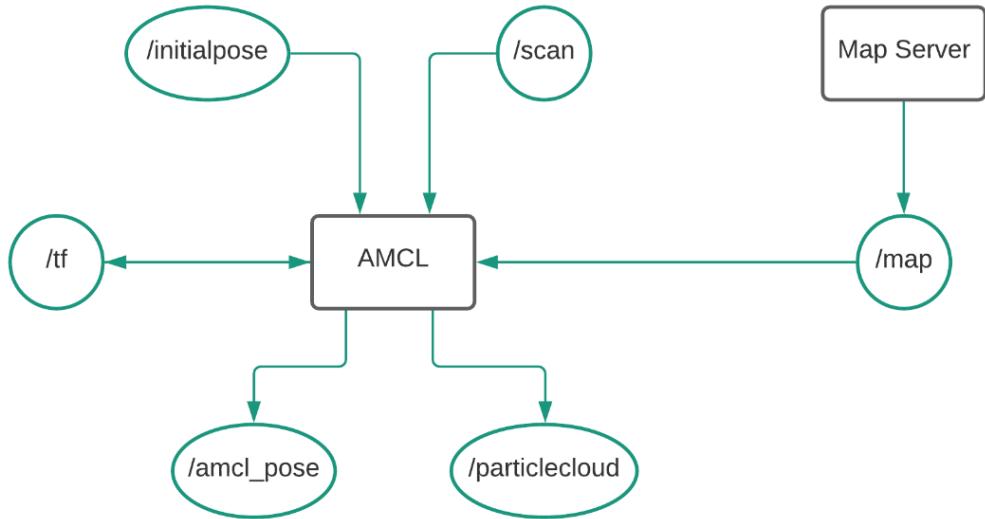


Εικόνα 5.4: Το φίλτρο σωματιδίων κατά τον εντοπισμό θέσης

Μόλις εκκινήσει ο αλγόριθμος υπάρχει μεγάλη αβεβαιότητα για τη θέση και τον προσανατολισμό του ρομπότ, και για αυτό το λόγο όπως φαίνεται και στο αριστερό τμήμα της εικόνας, τα σωματίδια του φίλτρου, που απεικονίζονται με πράσινα βέλη, είναι αρκετά διάσπαρτα. Όσο κινείται το ρομπότ, ο αλγόριθμος αναγνωρίζει μέσω των σαρώσεων λέιζερ σε ποιο σημείο το χάρτη βρίσκεται το ρομπότ, οπότε η αβεβαιότητα της θέσης και του προσανατολισμού του ρομπότ μειώνεται, με αποτέλεσμα τα περισσότερα σωματίδια του φίλτρου να είναι συγκεντρωμένα στη θέση του ρομπότ, το οποίο φαίνεται στο δεξί τμήμα της εικόνας.

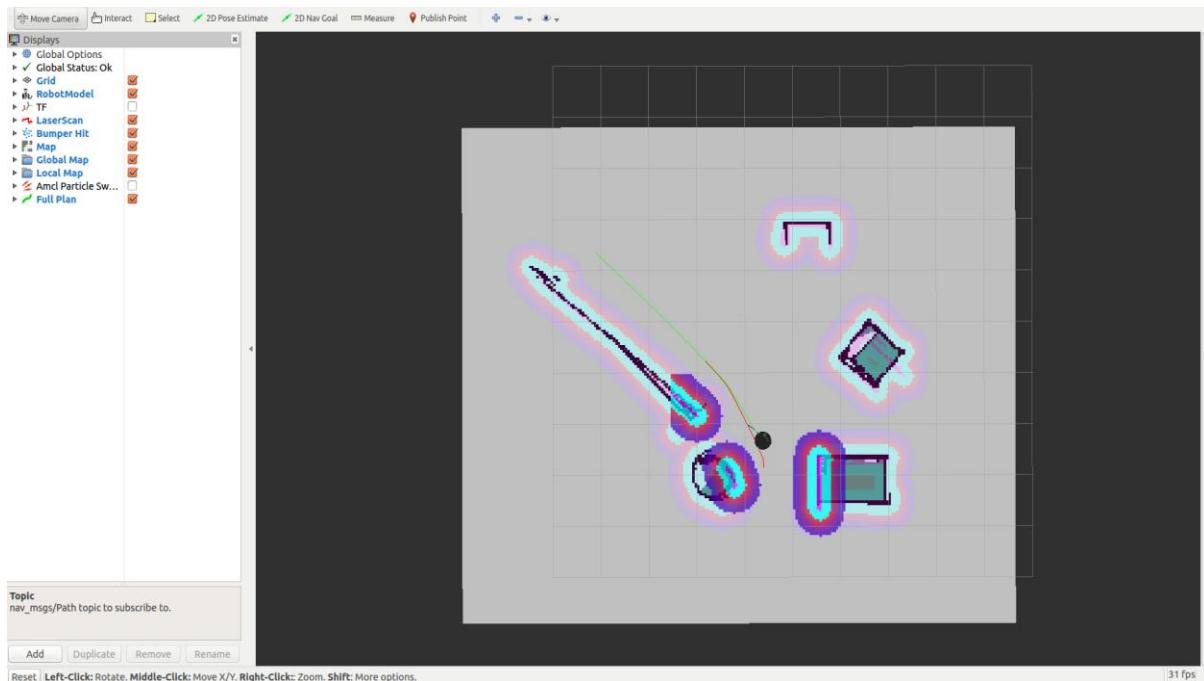
Στο διάγραμμα της εικόνας 5.5 φαίνονται τα μηνύματα που ανταλλάσσει ο αλγόριθμος κατά την εκτέλεσή του. Ο αλγόριθμος δέχεται μηνύματα από το /tf για τους μετασχηματισμούς που χρειάζεται, από το /initialpose για πιθανή αρχική εκτίμηση της θέσης, από το /map για το χάρτη και από το /scan για τη τρέχουσα σάρωση του λέιζερ. Ο χάρτης δημοσιεύεται στο /map από τον κόμβο map\_server<sup>1</sup>. Αφότου γίνει η εκτίμηση της θέσης του ρομπότ από τον αλγόριθμο, δημοσιεύει στο /amcl\_pose την εκτίμηση αυτή, στο /tf έναν μετασχηματισμό και στο /particlecloud το τρέχον φίλτρο σωματιδίων.

<sup>1</sup> [http://wiki.ros.org/map\\_server](http://wiki.ros.org/map_server)



Εικόνα 5.5: Διάγραμμα μηνυμάτων για τον εντοπισμό θέσης.

Όσον αφορά την πλοϊγηση, η λειτουργία του κόμβου move\_base που την υλοποιεί αναλύθηκε στην υποενότητα 4.3.3, η οποία φαίνεται και στην εικόνα 5.6.

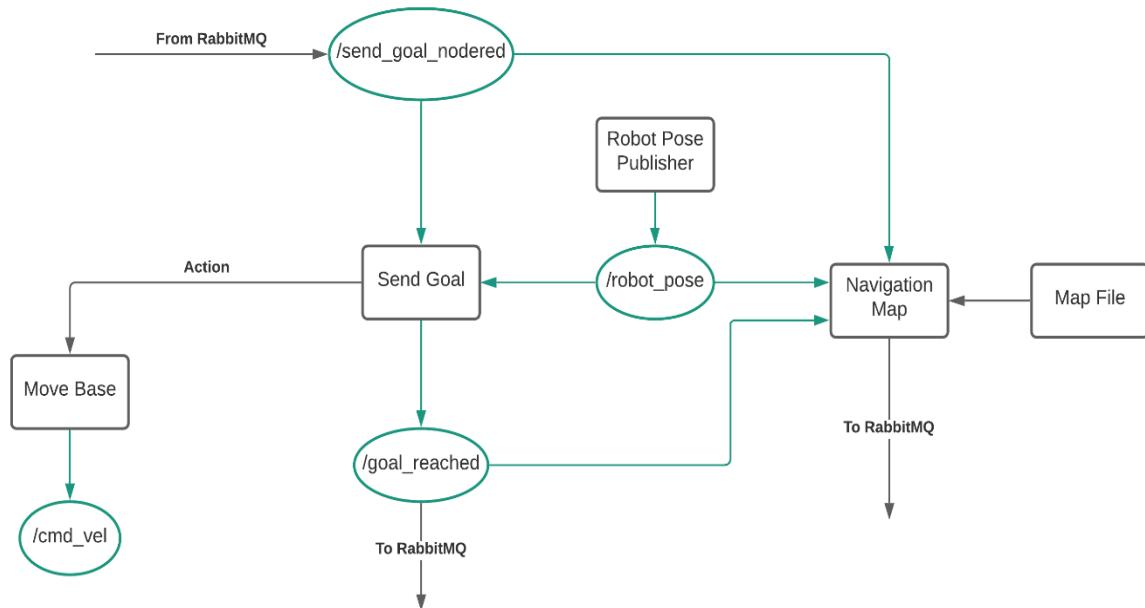


Εικόνα 5.6: Οι χάρτες κόστους και τα μονοπάτια κατά την πλοϊγηση

Ο τοπικός χάρτης κόστους φαίνεται με τα έντονα χρώματα πάνω στα εμπόδια που βρίσκονται κοντά στο ρομπότ, ενώ ο καθολικός χάρτης φαίνεται σε όλα τα εμπόδια. Βάσει αυτών των χαρτών, γίνεται ο υπολογισμός των εντολών ταχύτητας που θα σταλθούν στο ρομπότ. Επιπλέον, στην εικόνα απεικονίζεται και το μονοπάτι που έχει υπολογίσει ο αλγόριθμος για να φτάσει στην επιθυμητή θέση.

## ΚΕΦΑΛΑΙΟ 5. ΥΛΟΠΟΙΗΣΗ

Όπως και στο SLAM, έτσι και εδώ έχει υλοποιηθεί ένας κόμβος, ο navigation\_map, για το σχηματισμό της εικόνας του χάρτη, συμπεριλαμβανομένου του ρομπότ και του ενδεχόμενου επιθυμητού στόχου θέσης που του έχει δοθεί. Παράλληλα, δεδομένου ότι τα μηνύματα στόχων θέσης που δέχεται ο κόμβος move\_base πρέπει να περιέχουν μία κεφαλίδα (header) με πληροφορίες που δε μπορούν να παραχθούν εκτός του ROS, υλοποιήθηκε ένας επιπλέον κόμβος, ο send\_goal, που δέχεται τα μηνύματα στόχων θέσης από το Node-RED, προσθέτει τις απαραίτητες πληροφορίες και τα στέλνει στο move\_base. Στο διάγραμμα της εικόνας 5.7 φαίνονται οι ανταλλαγές μηνυμάτων μεταξύ των κόμβων αυτών.



Εικόνα 5.7: Διάγραμμα μηνυμάτων για την πλοήγηση

Όπως εξηγήθηκε και προηγουμένως, ο κόμβος send\_goal δέχεται το μήνυμα του στόχου θέσης από το /send\_goal\_nodered, και τη θέση του ρομπότ από το /robot\_pose και καλεί το action του κόμβου move\_base για την πλοήγηση στο στόχο. Οι δράσεις (actions) είναι παρόμοιες με τα services που προσφέρει το ROS, με κάποιες επιπλέον δυνατότητες. Πέρα από την αποστολή του στόχου θέσης, στέλνει και ένα μήνυμα επιβεβαίωσης στο /goal\_reached όταν το ρομπότ φτάσει με επιτυχία στο στόχο που του έχει τεθεί.

Ο κόμβος move\_base δέχεται με το action τον επιθυμητό στόχο και στέλνει τις εντολές ταχύτητας του ρομπότ στο /cmd\_vel. Δεν παρουσιάζεται αναλυτικά η δομή και τα topics με τα οποία επικοινωνεί ο κόμβος, για λόγους απλότητας. Η γενική δομή του φαίνεται στην εικόνα 4.3.

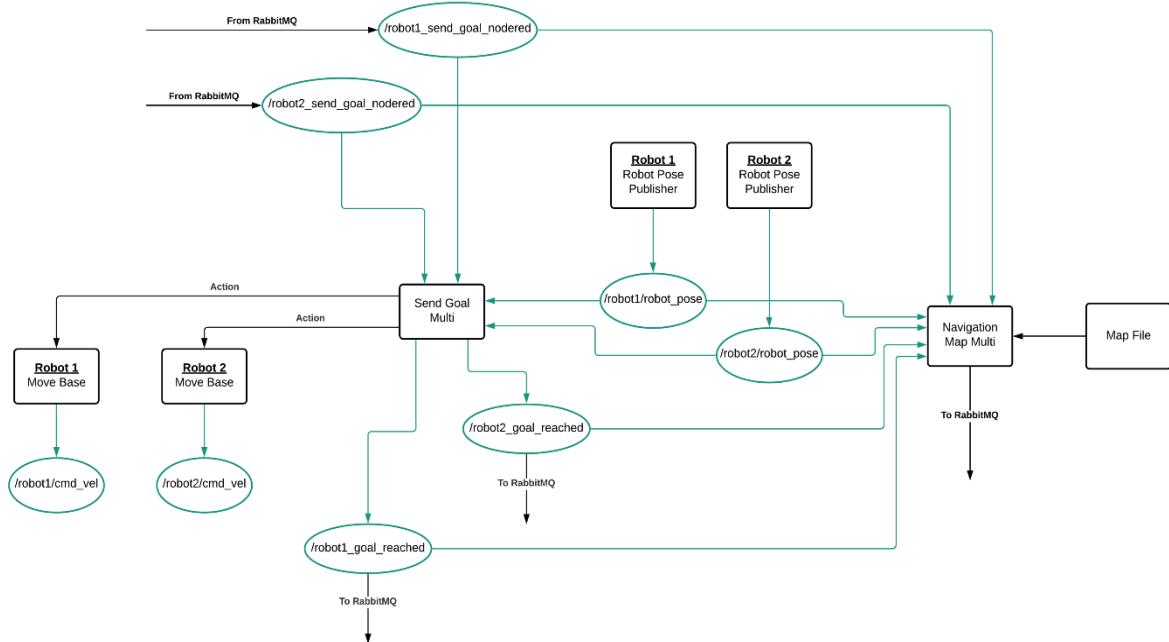
Τέλος, ο κόμβος navigation\_map είναι υπεύθυνος για το σχηματισμό της εικόνας του χάρτη κατά την πλοήγηση. Ο χάρτης που έχει δημιουργηθεί από το SLAM έχει αποθηκευτεί σε ένα αρχείο. Ο κόμβος φορτώνει το χάρτη από το αρχείο αυτό, δέχεται τη θέση του ρομπότ από το /robot\_pose, το στόχο θέσης από το /send\_goal\_nodered και το μήνυμα επιβεβαίωσης από το /goal\_reached για να ξέρει αν πρέπει να σχεδιάσει το στόχο, και με αυτές τις πληροφορίες σχηματίζει την εικόνα που στέλνει στο RabbitMQ.

### 5.2.1.2 Λειτουργία πολλαπλών ρομπότ

Το δεύτερο πακέτο είναι υπεύθυνο για την παροχή των λειτουργιών που πρέπει να προσφέρονται στο χρήστη, όταν χρησιμοποιούνται πολλά ρομπότ. Αυτές είναι ο εντοπισμός θέσης και η πλοήγηση. Στη συγκεκριμένη υλοποίηση, οι λειτουργίες παρέχονται για χρήση δύο ρομπότ, αλλά μπορούν εύκολα να επεκταθούν και για παραπάνω.

#### Εντοπισμός θέσης και πλοήγηση

Όπως και στο ένα ρομπότ, έτσι και εδώ υλοποιήθηκαν δύο κόμβοι, ο send\_goal\_multi που πάλι δέχεται τα μηνύματα στόχου, τα οποία τροποποιεί και στέλνει στη συνέχεια στους κόμβους move\_base των δύο ρομπότ, και ο navigation\_map\_multi που σχηματίζει την εικόνα του χάρτη μαζί με τις θέσεις των ρομπότ. Η διαφορά αυτή τη φορά είναι ότι τα ονόματα των topics που σχετίζονται με κάθε ρομπότ πρέπει να τροποποιηθούν για να μην υπάρχουν διαμάχες. Για αυτό το λόγο, στα topics κάθε ρομπότ εισάγεται και ο χώρος ονομάτων (namespace), όπως φαίνεται στο διάγραμμα μηνυμάτων για τη λειτουργία της πλοήγησης στην εικόνα 5.8.

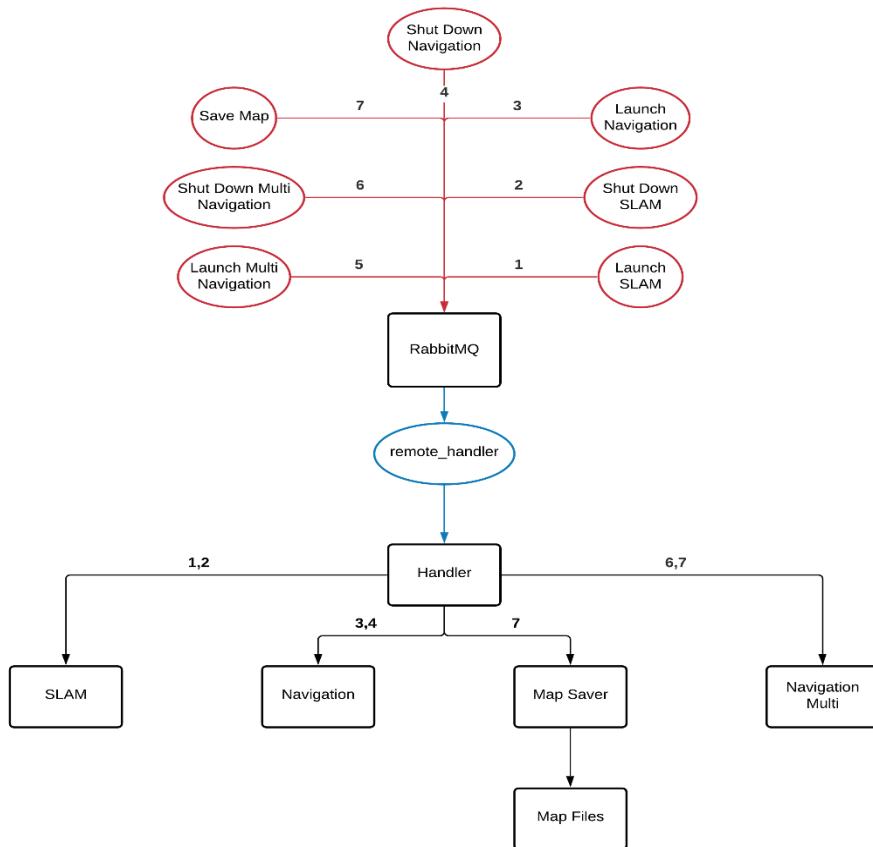


Εικόνα 5.8: Διάγραμμα μηνυμάτων για την πλοήγηση πολλαπλών ρομπότ

Όπως φαίνεται από το διάγραμμα, η λειτουργία των κόμβων είναι ίδια, απλά πλέον έχουμε δύο ρομπότ και έχουν αλλάξει τα ονόματα των topics.

### 5.2.2 Διαχείριση των λειτουργιών

Για την αρχικοποίηση και τον τερματισμό των λειτουργιών που αναλύθηκαν, ανάλογα με την εντολή που θα ληφθεί από το χρήστη μέσω του Node-RED, είναι υπεύθυνος ο handler. Οι εντολές που μπορεί να δώσει ο χρήστης, και οι λειτουργίες που θα εκκινήσει ή θα τερματίσει ο handler, φαίνονται στο διάγραμμα της εικόνας 5.9.



Εικόνα 5.9: Διαχείριση εντολών χρήστη από το handler

Οι εντολές που μπορεί να δώσει ο χρήστης για την αρχικοποίηση ή τον τερματισμό κάποιας λειτουργίας φαίνονται με κόκκινο χρώμα. Στη συνέχεια, μεταφέρονται μέσω του RabbitMQ στην AMQP ουρά `remote_handler`, που απεικονίζεται με μπλε χρώμα, στην οποία έχει συνδεθεί ο `handler` και τις λαμβάνει. Έπειτα, ανάλογα με την εντολή, εκκινεί ή τερματίζει την αντίστοιχη διαδικασία. Οι λειτουργίες SLAM και πλοιήγηση ενός ή πολλαπλών ρομπότ αναλύθηκαν προηγουμένως. Η αποθήκευση του χάρτη του SLAM γίνεται με τον κόμβο `map_saver` του πακέτου `map_server`.

### 5.2.3 Επικοινωνία με το RabbitMQ

Στην υποενότητα 2.1.5 έχουμε αναφερθεί στα πρωτόκολλα μηνυμάτων που υποστηρίζει το RabbitMQ, είτε άμεσα είτε μέσω plugin. Δυστυχώς, τα μηνύματα του ROS δεν είναι δομημένα σε μορφή που να υποστηρίζεται από το RabbitMQ. Επομένως, θα πρέπει να γίνει μετατροπή των μηνυμάτων αυτών σε κάποια συμβατή μορφή. Δεδομένου ότι το κύριο πρωτόκολλο του μεσολαβητή είναι το AMQP, αλλά και γιατί είναι αρκετά διαδεδομένο, είναι αυτό που επιλέχθηκε.

Το `ros2broker` είναι υπεύθυνο για τη γεφύρωση της επικοινωνίας μεταξύ του ROS και του RabbitMQ. Επειδή δεν υλοποιήθηκε στα πλαίσια αυτής της διπλωματικής εργασίας, δε θα αναλυθεί λεπτομερώς ο τρόπος με τον οποίο επιτυγχάνεται η διασύνδεση, απλά θα περιγραφεί ο τρόπος χρήσης του.

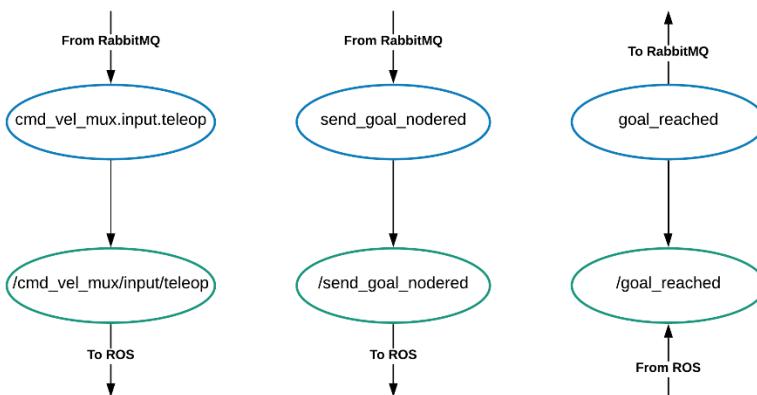
Ο πρώτος τρόπος που μπορεί να χρησιμοποιηθεί το ros2broker, είναι παρέχοντάς του ένα αρχείο του μοντέλου των επιθυμητών συνδέσεων. Σε αυτό το αρχείο εμπεριέχονται όλες οι πληροφορίες του μεσολαβητή, όπως η διεύθυνσή του και η θύρα στην οποία αναμένει τις συνδέσεις. Παράλληλα, ορίζονται αναλυτικά οι παράμετροι των επιθυμητών συνδέσεων που θέλουμε να δημιουργήσει το ros2broker. Οι παράμετροι αυτοί είναι:

- Το είδος της σύνδεσης, δηλαδή αν το ROS δρα ως subscriber ή publisher, ή αν θέλουμε να καλέσουμε κάποιο service οπότε πρόκειται για RPC επικοινωνία.
- Οι λεπτομέρειες του topic του ROS, δηλαδή το όνομα του topic ή του service, το είδος του μηνύματός του κ.ά.
- Οι λεπτομέρειες στη μεριά του μεσολαβητή, δηλαδή το όνομα του exchange που θα χρησιμοποιηθεί και το κλειδί με το οποίο θα συνδεθεί η ουρά στο exchange, το οποίο λειτουργεί όπως το όνομα του topic στο ROS.

Ο δεύτερος τρόπος είναι να κληθεί το ros2broker μέσω κώδικα. Και σε αυτήν την περίπτωση, θα πρέπει να οριστούν λεπτομερώς όλες οι πληροφορίες της επιθυμητής σύνδεσης, με τον ίδιο τρόπο με πριν. Ο πρώτος τρόπος είναι πιο βολικός στη χρήση και είναι αυτός που χρησιμοποιήθηκε. Στα πλαίσια του συστήματος υλοποιήθηκαν δύο μοντέλα συνδέσεων, ένα για λειτουργία ενός ρομπότ και ένα για πολλαπλά ρομπότ.

### 1. Λειτουργία ενός ρομπότ:

Το μοντέλο αυτό περιγράφει τρεις συνδέσεις. Η πρώτη αφορά τις εντολές ταχύτητας που δίνει ο χρήστης μέσω του Node-RED για τον τηλεχειρισμό του ρομπότ. Το ROS δρα ως subscriber και τα μηνύματα φτάνουν στο /cmd\_vel\_mux/input/teleop topic. Η δεύτερη αφορά τις εντολές των επιθυμητών στόχων θέσης που στέλνει ο χρήστης στο ρομπότ. Πάλι, το ROS λειτουργεί ως subscriber, ενώ τα μηνύματα φτάνουν στο /send\_goal\_nodered topic, όπου τα λαμβάνει ο κόμβος send\_goal. Η τρίτη και τελευταία σύνδεση αφορά το μήνυμα επιβεβαίωσης που στέλνεται όταν το ρομπότ φτάσει στον επιθυμητό στόχο. Αυτή τη φορά το ROS δρα ως publisher μέσω του topic /goal\_reached, και το μήνυμα μεταβιβάζεται προς το Node-RED μέσω του μεσολαβητή. Η αντιστοίχιση των ROS topics σε AMQP ουρές που έχουν συνδεθεί στο μεσολαβητή φαίνεται στην εικόνα 5.10.

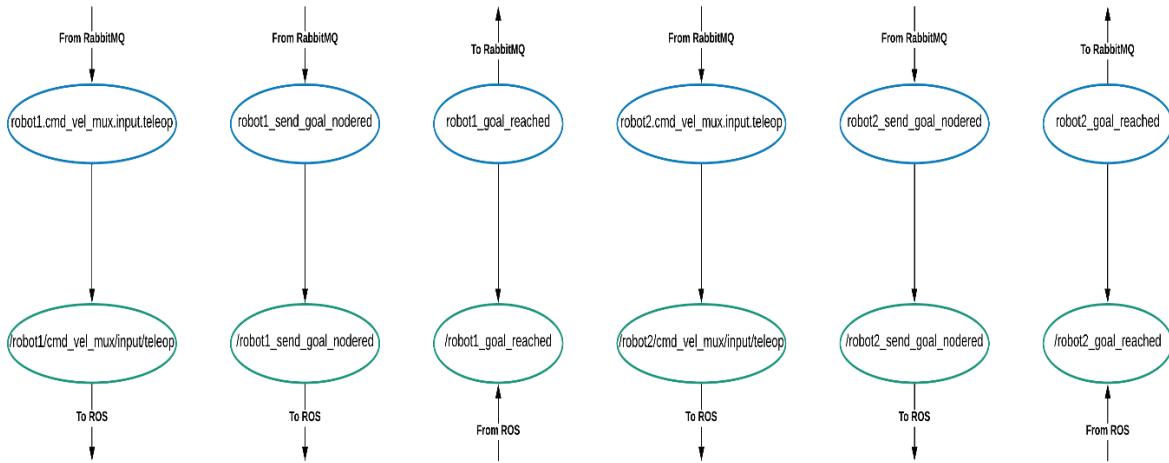


Εικόνα 5.10: Αντιστοίχιση ROS topics σε ουρές AMQP για λειτουργία ενός ρομπότ

Με πράσινο χρώμα απεικονίζονται τα ROS topics, ενώ με μπλε οι ουρές AMQP που έχουν συνδεθεί στο μεσολαβητή. Το ROS για το διαχωρισμό των λέξεων στα ονόματα των topics, χρησιμοποιεί το σύμβολο “ / ”, όπως και το πρωτόκολλο MQTT. Αντιθέτως, στο AMQP ο διαχωρισμός γίνεται με το σύμβολο της τελείας “ . ”.

## 2. Λειτουργία πολλαπλών ρομπότ

Το μοντέλο αυτό είναι παρόμοιο με το προηγούμενο, αλλά εδώ υπάρχουν έξι συνδέσεις, τρεις για κάθε ρομπότ. Και οι τρεις αυτές συνδέσεις είναι ίδιες με πριν, απλά έχει συμπεριληφθεί και το namespace του κάθε ρομπότ, για να γίνεται σωστά η σύνδεση με τα εκάστοτε topic. Οι αντιστοιχίεις φαίνονται στην εικόνα 5.11.



Εικόνα 5.11: Αντιστοιχίση ROS topics σε ουρές AMQP για λειτουργία πολλαπλών ρομπότ

## 5.3 Μεσολαβητής μηνυμάτων

Το δεύτερο κομμάτι του συστήματος είναι ο μεσολαβητής μηνυμάτων RabbitMQ. Λειτουργώντας ως server, πραγματοποιεί τη μεταφορά δεδομένων μεταξύ του ROS στο τοπικό τμήμα και του Node-RED στο απομακρυσμένο τμήμα. Όπως και το απομακρυσμένο τμήμα, έτσι και ο μεσολαβητής βρίσκεται εκτός του τοπικού δικτύου. Η γενική λειτουργία του RabbitMQ έχει αναλυθεί στην ενότητα 4.5.

Στη συνέχεια θα περιγραφούν όλες οι συνδέσεις με το RabbitMQ τόσο από τη μεριά του ρομπότ, όσο και από τη μεριά της εφαρμογής του χρήστη. Από τη μεριά του ρομπότ, χρησιμοποιείται το πρωτόκολλο μηνυμάτων AMQP και οι συνδέσεις με το ROS είναι οι εξής:

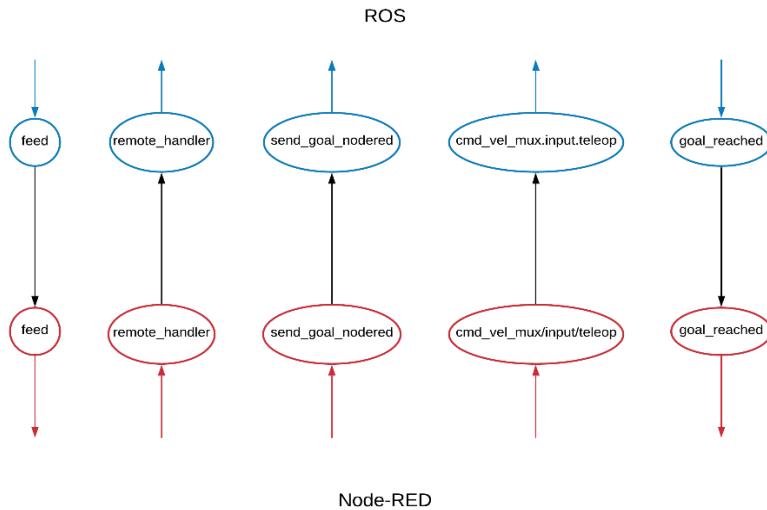
- Handler:** Ο handler δημιουργεί μία σύνδεση με το RabbitMQ για τη λήψη μηνυμάτων. Χρησιμοποιεί το default topic exchange που παρέχει ο μεσολαβητής, με ονομασία amq.topic και συνδέει σε αυτό μία ουρά με κλειδί “remote\_handler”. Επομένως, όποιο μήνυμα σταλθεί σε αυτό το exchange με κλειδί δρομολόγησης “remote\_handler”, θα μεταφερθεί στην ουρά που δημιούργησε, και κατ’ επέκταση σε αυτό.

- Αποστολή εικόνων:** Οι κόμβοι που πρέπει να στείλουν εικόνα στο Node-RED δημιουργούν μία σύνδεση με το RabbitMQ για την αποστολή μηνυμάτων. Χρησιμοποιούν πάλι το amq.topic exchange και στέλνουν το μήνυμα τους με κλειδί δρομολόγησης “feed”. Όσες ουρές έχουν συνδεθεί στο exchange με το αντίστοιχο κλειδί θα λάβουν την εικόνα.
- ros2broker:** Οι κόμβοι που λαμβάνουν τους επιθυμητούς στόχους που πρέπει να ακολουθήσει το ρομπότ, καθώς και τα topics που αναμένουν εντολές ταχύτητας από τον τηλεχειρισμό, λαμβάνουν τα δεδομένα από τις ουρές που δημιουργεί το ros2broker. Και εδώ χρησιμοποιείται το amq.topic exchange ενώ τα ονόματα των ουρών ορίζονται στο αντίστοιχο μοντέλο που χρησιμοποιεί το ros2broker.

Από τη μεριά της εφαρμογής χρησιμοποιείται το πρωτόκολλο MQTT. Από προεπιλογή το Node-RED στέλνει και λαμβάνει μηνύματα χρησιμοποιώντας το πρωτόκολλο αυτό. Για την υποστήριξη του πρωτοκόλλου AMQP έχουν δημιουργηθεί μερικά πακέτα από την κοινότητα που μπορούν να χρησιμοποιηθούν. Δυστυχώς, κατά την υλοποίηση του συστήματος κανένα από αυτά τα πακέτα δε διύλευε σωστά και παρουσίαζαν προβλήματα. Για αυτό το λόγο, και δεδομένου ότι όλες οι επιθυμητές λειτουργίες μπορούν να καλυφθούν από το MQTT, επιλέξαμε να χρησιμοποιήσουμε αυτό το πρωτόκολλο για την επικοινωνία μεταξύ του Node-RED και του RabbitMQ. Οι συνδέσεις με το Node-RED είναι οι εξής:

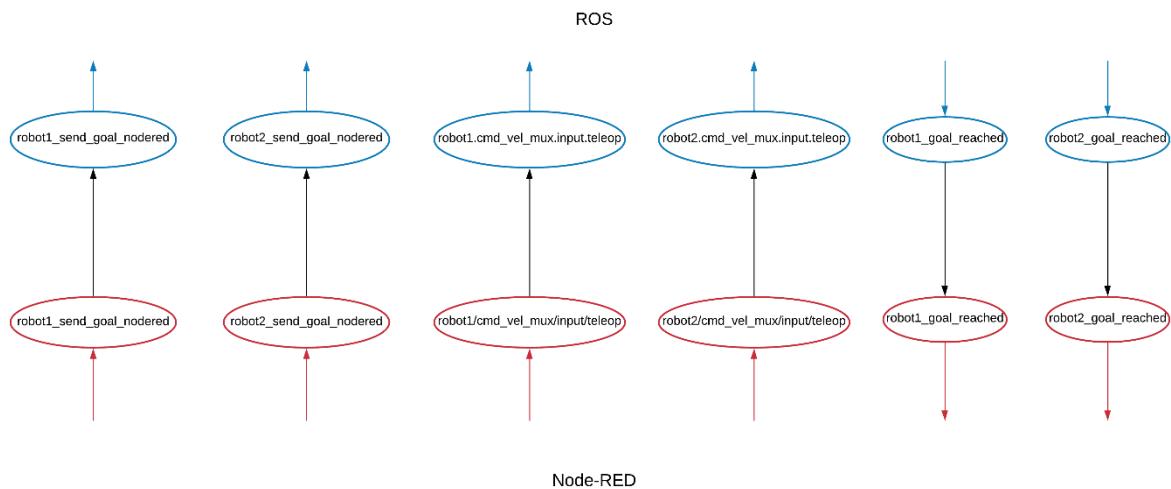
- Εντολές για handler:** Για την αποστολή των μηνυμάτων στο remote\_handler χρησιμοποιείται το topic “remote\_handler”. Στο MQTT πρωτόκολλο τα μηνύματα στέλνονται απευθείας σε κάποιο topic. Επομένως, το RabbitMQ δέχεται το MQTT μήνυμα και το topic του, το μετατρέπει σε μήνυμα AMQP και αντιστοιχίζει το topic του MQTT μηνύματος με το κλειδί δρομολόγησης του AMQP μηνύματος.
- Λήψη εικόνων:** Για τη λήψη των εικόνων, το Node-RED εγγράφεται στο topic “feed” και αναμένει τα μηνύματα που στέλνουν οι κόμβοι του ROS.
- Εντολές ταχύτητας:** Οι εντολές ταχύτητας για το τηλεχειρισμό του ρομπότ στέλνονται στο topic “cmd\_vel\_mux/input/teleop” αν πρόκειται για λειτουργία ενός ρομπότ, ενώ συμπεριλαμβάνεται και το namespace στην περίπτωση των πολλαπλών ρομπότ.
- Εντολές επιθυμητών στόχων:** Ομοίως, οι εντολές των επιθυμητών στόχων που πρέπει να ακολουθήσει το ρομπότ, στέλνονται στο topic “send\_goal\_nodered” στην περίπτωση του ενός ρομπότ, ενώ προστίθεται και το namespace για λειτουργία πολλαπλών ρομπότ.
- Λήψη επιβεβαιώσης στόχου:** Τέλος, για τα μηνύματα επιβεβαιώσης ότι το ρομπότ έφτασε επιτυχώς στο στόχο, το Node-RED εγγράφεται στο topic “goal\_reached” όταν πρόκειται για ένα ρομπότ, και προστίθεται και το namespace όταν πρόκειται για πολλαπλά ρομπότ.

Όλα τα MQTT topics από τη μεριά του Node-RED, αλλά και οι AMQP ουρές από τη μεριά του τοπικού τμήματος για τη λειτουργία ενός ρομπότ φαίνονται στο διάγραμμα της εικόνας 5.12.



Εικόνα 5.12: Μεταφορά μηνυμάτων από το RabbitMQ για λειτουργία ενός ρομπότ

Με μπλε χρώμα απεικονίζονται οι AMQP ουρές από τη μεριά του ROS, ενώ με κόκκινο τα MQTT topics από τη μεριά του Node-RED. Η λειτουργία πολλαπλών ρομπότ φαίνεται στην εικόνα 5.13.



Εικόνα 5.13: Μεταφορά μηνυμάτων από το RabbitMQ για λειτουργία πολλαπλών ρομπότ

Αξίζει να σημειωθεί, ότι όλες οι συνδέσεις που δημιουργούνται εμπεριέχονται σε virtual host. Συνεπώς, είναι πολύ εύκολο να υποστηριχθούν πολλαπλοί χρήστες, αφού τα εκάστοτε περιβάλλοντα είναι απομονωμένα και δεν επικοινωνούν μεταξύ τους.

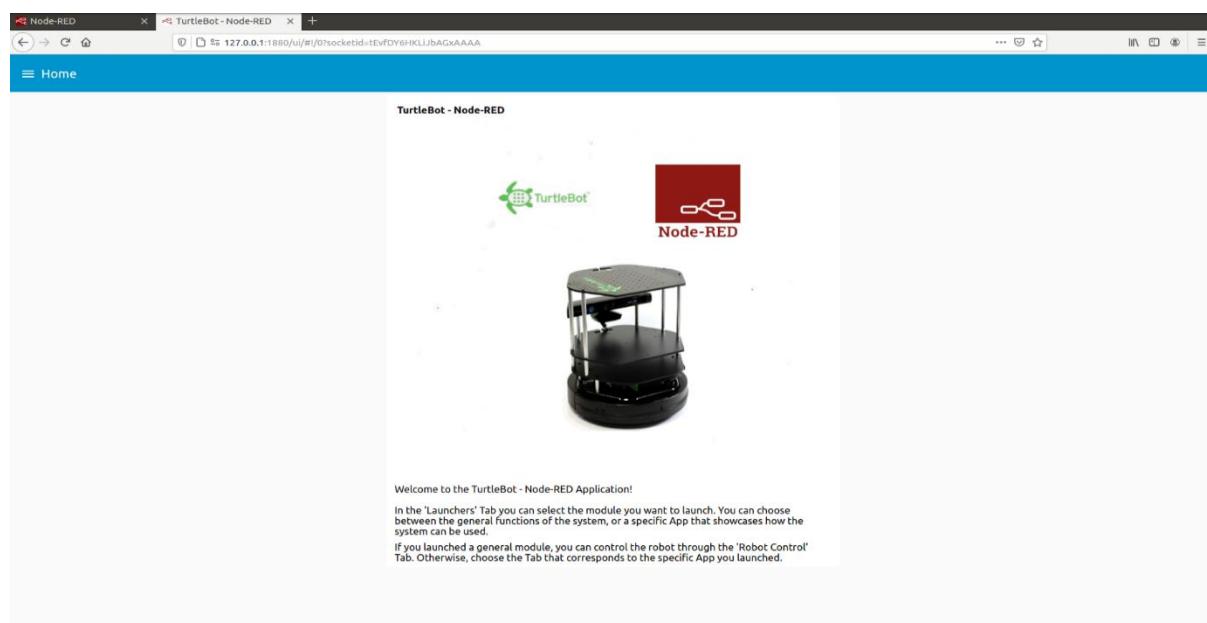
## 5.4 Απομακρυσμένο τμήμα του συστήματος

Στον απομακρυσμένο υπολογιστή του χρήστη γίνεται η ανάπτυξη των εφαρμογών, μέσω του Node-RED. Το τμήμα αυτό είναι εντελώς ανεξάρτητο από το τοπικό τμήμα, που βρίσκεται το ρομπότ, και η μεταξύ τους επικοινωνία γίνεται μέσω του RabbitMQ. Στο χρήστη παρέχονται οι λειτουργίες εντοπισμού θέσης, πλοιήγησης και τηλεχειρισμού ανεξαρτήτως του αριθμού των ρομπότ, και SLAM για χρήση μόνο ενός ρομπότ.

Για την οπτικοποίηση όλων αυτών των λειτουργιών, αλλά και για ευκολία στη χρήση, χρησιμοποιήθηκε το Node-RED Dashboard, μία γραφική διεπαφή όπου μπορούν να παρουσιαστούν γρήγορα και εύκολα πληροφορίες από το Node-RED. Το Dashboard είναι πρόσθετο πακέτο και μπορεί να εγκατασταθεί πολύ εύκολα. Το γεγονός ότι είναι ενσωματωμένο στο Node-RED, διευκολύνει πολύ τη διαδικασία, αφού δε χρειάζεται να αναπτυχθεί εκ νέου μία διεπαφή ιστού που θα επικοινωνεί με το Node-RED.

Ένα ακόμα πολύ χρήσιμο χαρακτηριστικό του Dashboard είναι ότι ο χρήστης μπορεί να έχει πρόσβαση σε αυτό και από οποιαδήποτε συσκευή βρίσκεται στο τοπικό του δίκτυο. Το μόνο που έχει να κάνει είναι να πληκτρολογήσει σε οποιοδήποτε φυλλομετρητή, τη διεύθυνση τοπικού δικτύου του υπολογιστή στον οποίο τρέχει το Node-RED, τη θύρα που χρησιμοποιεί το Node-RED και την κατάληξη /ui που αντιστοιχεί στο Dashboard. Με αυτόν τον τρόπο, ο χρήστης μπορεί εύκολα να ελέγξει την κατάσταση του συστήματος και να κάνει οποιαδήποτε τροποποίηση εύκολα, για παράδειγμα από το κινητό του τηλέφωνο.

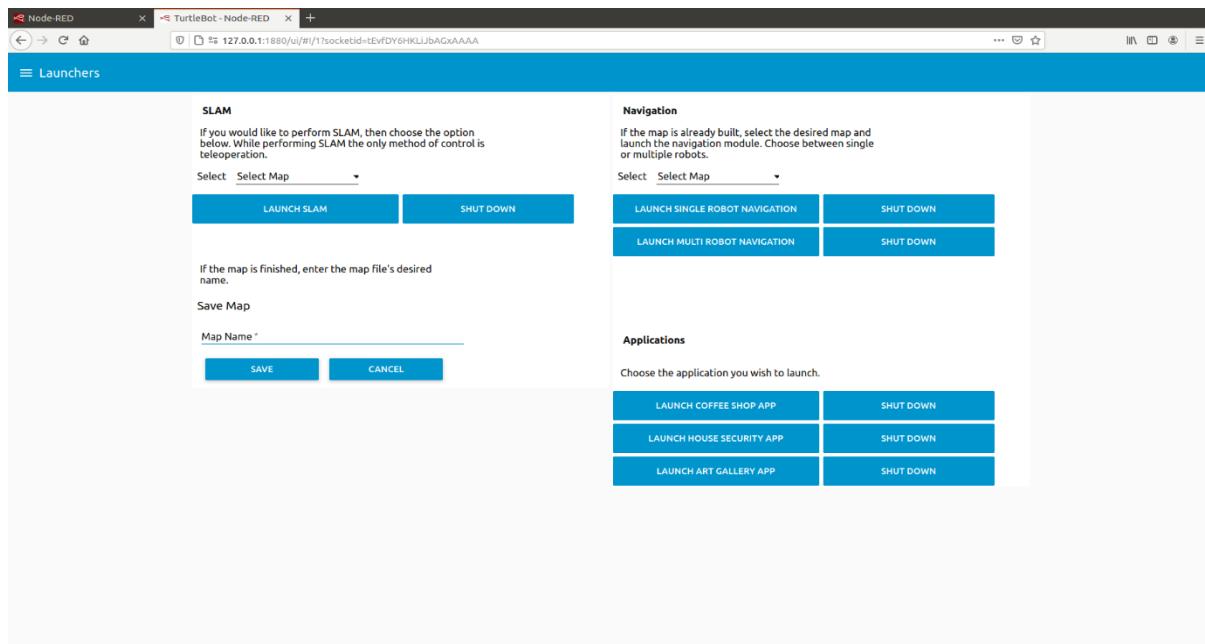
Το Dashboard χωρίζει τη διεπαφή σε καρτέλες (tabs) και κάθε καρτέλα σε ομάδες (groups). Η διεπαφή του συστήματος αποτελείται από τρεις καρτέλες για τη γενική λειτουργία του συστήματος, και άλλες τρεις για τα σενάρια χρήσης, τα οποία θα αναλυθούν στο επόμενο κεφάλαιο. Η καρτέλα “Home” (εικόνα 5.14) είναι η πρώτη καρτέλα, η οποία καλωσορίζει το χρήστη στην εφαρμογή και του εξηγεί συνοπτικά τη λειτουργία των υπόλοιπων καρτελών.



Εικόνα 5.14: Η καρτέλα Home του Dashboard.

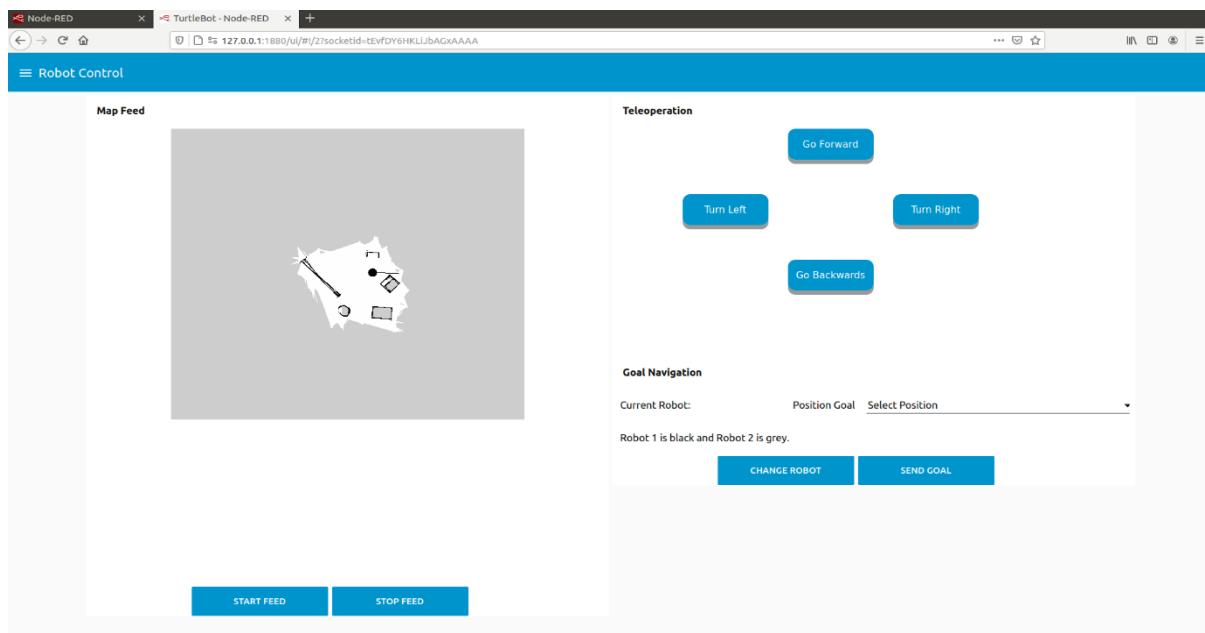
Η καρτέλα “Launchers” (εικόνα 5.15) είναι υπεύθυνη για την αρχικοποίηση και των τερματισμό όλων των διαθέσιμων λειτουργιών. Επίσης, παρέχεται και η επιλογή της αποθήκευσης του χάρτη που έχει δημιουργηθεί από τη διαδικασία του SLAM, με το όνομα που επιθυμεί ο χρήστης. Τα αρχεία αυτά δημιουργούνται στον υπολογιστή του τοπικού τμήματος του συστήματος. Για την εκκίνηση των διαδικασιών ο χρήστης πρέπει πρώτα να επιλέξει το χάρτη τον οποίο θέλει να χρησιμοποιήσει. Έπειτα, ανάλογα με τη λειτουργία που επιθυμεί να εκκινήσει, πατάει το αντίστοιχο κουμπί, και σχηματίζεται στο Node-RED το σωστό μήνυμα που θα σταλθεί στο handler.

## ΚΕΦΑΛΑΙΟ 5. ΥΛΟΠΟΙΗΣΗ



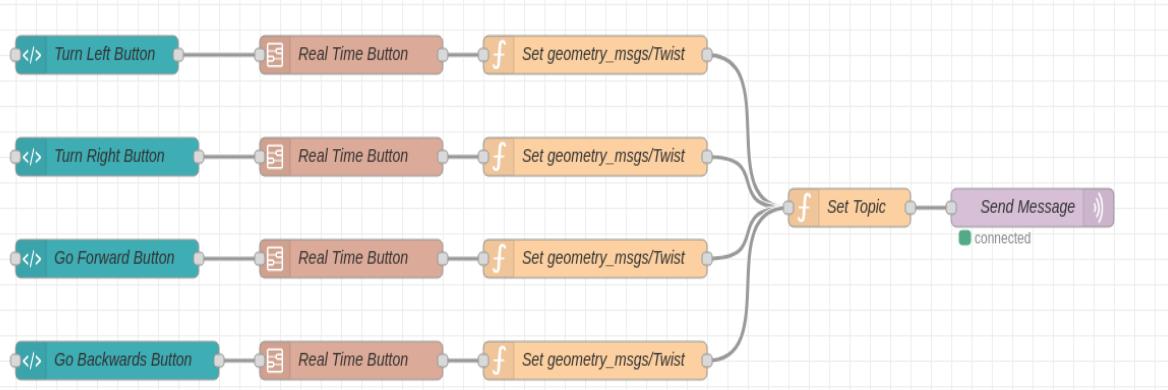
*Εικόνα 5.15: Η καρτέλα Launchers του Dashboard*

Τέλος, μέσω της καρτέλας “Robot Control” (εικόνα 5.16) πραγματοποιείται ο έλεγχος του ρομπότ. Στο αριστερό τμήμα εμφανίζεται είτε ο τρέχων χάρτης κατά την εκτέλεση του SLAM, συμπεριλαμβανομένης της θέσης του ρομπότ και των σημείων που σαρώθηκαν από το λέιζερ, είτε ο χάρτης που έχει δημιουργηθεί από το SLAM, μαζί με τη θέση του ρομπότ και τον ενδεχόμενο επιθυμητό στόχο που του έχει τεθεί. Παρόλο που το Dashboard δε παρέχει κάποιο κόμβο για την απεικόνιση εικόνων, δίνεται η δυνατότητα χρήσης HTML για να υποστηριχθεί όποια επιπλέον λειτουργικότητα επιθυμεί ο χρήστης, οπότε με αυτόν τον τρόπο απεικονίζεται και η εικόνα στο Dashboard. Ανά πάσα στιγμή, ο χρήστης μπορεί να διακόψει και να εκκινήσει την αναπαραγωγή του χάρτη με τα αντίστοιχα κουμπιά που βρίσκονται στο κάτω μέρος.



*Εικόνα 5.16: Η καρτέλα Robot Control του Dashboard*

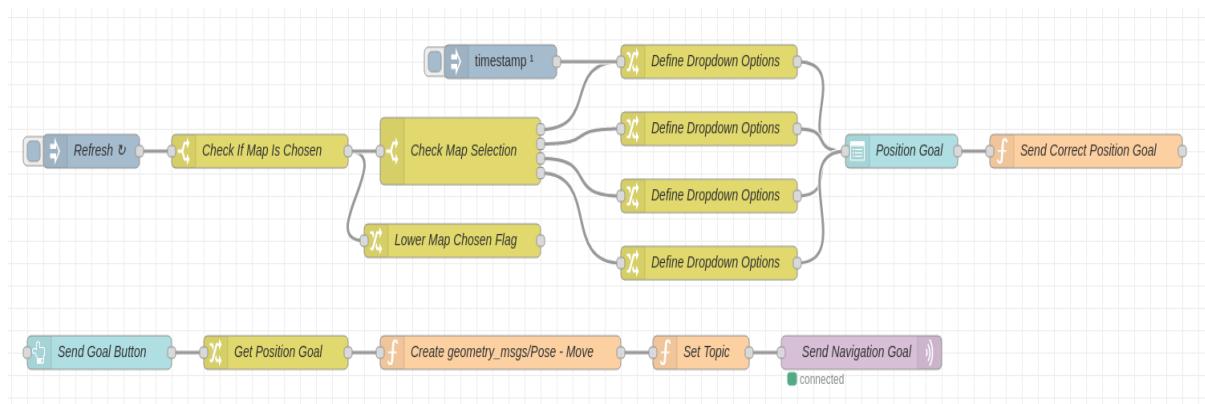
Στο δεξί τμήμα δίνονται οι δύο τρόποι ελέγχου του ρομπότ, ο τηλεχειρισμός και η αποστολή επιθυμητών στόχων θέσης. Ο τηλεχειρισμός γίνεται μέσω των τεσσάρων κουμπιών, όπως φαίνεται στην εικόνα. Ανάλογα με το κουμπί, το ρομπότ είτε πηγαίνει μπροστά / πίσω, είτε περιστρέφεται αριστερά / δεξιά. Όσο ο χρήστης κρατάει πατημένο το κουμπί, η εντολή συνεχίζει να στέλνεται. Στην εικόνα 5.17 φαίνεται το flow για την υλοποίηση αυτής της συμπεριφοράς.



Εικόνα 5.17: To Node-RED flow για τον τηλεχειρισμό

Επειδή ο κόμβος “button” που παρέχεται από το Dashboard δεν αναγνωρίζει αν ο χρήστης κρατάει πατημένο το κουμπί, υλοποιήθηκε ένας ειδικός κόμβος που λειτουργεί ως κουμπί, αλλά παρέχει και την πληροφορία αν είναι πατημένο ή όχι. Αυτήν την πληροφορία επεξεργάζεται ο επόμενος κόμβος και στέλνει σήμα όταν το κουμπί είναι πατημένο. Αφότου έρθει αυτό το σήμα, σχηματίζεται το μήνυμα κίνησης για το ρομπότ, επιλέγεται το σωστό topic ανάλογα με το αν χρησιμοποιούνται ένα ή πολλαπλά ρομπότ, και τέλος στέλνεται το MQTT μήνυμα στο RabbitMQ.

Για την αποστολή στόχων θέσης στο ρομπότ, ο χρήστης μπορεί να επιλέξει ανάμεσα στους διαθέσιμους στόχους που έχουν οριστεί και να στείλει την εντολή. Επιπλέον, για τη λειτουργία πολλαπλών ρομπότ, ο χρήστης μπορεί εύκολα να αλλάξει το ρομπότ το οποίο χειρίζεται με το αντίστοιχο κουμπί. Το flow για την αποστολή στόχων θέσης φαίνεται στην εικόνα 5.18.



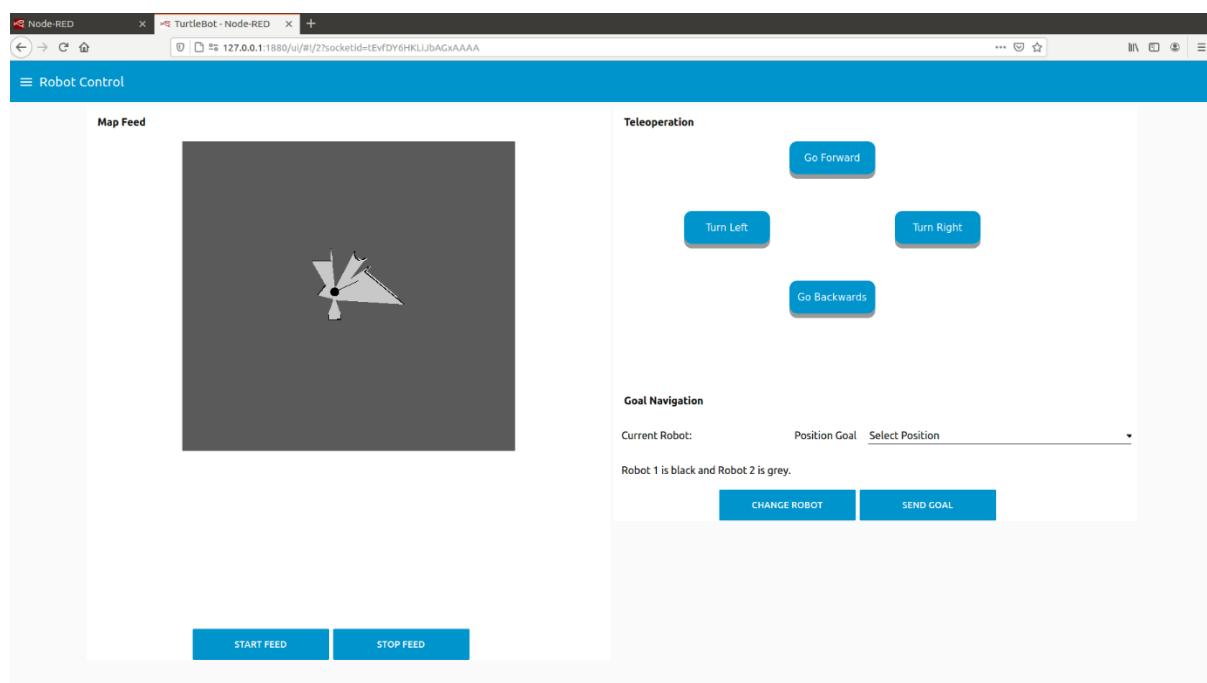
Εικόνα 5.18: To Node-RED flow για την πλοιήγηση

## ΚΕΦΑΛΑΙΟ 5. ΥΛΟΠΟΙΗΣΗ

Αρχικά, στο πάνω τμήμα γίνεται έλεγχος για το αν έχει επιλεχθεί κάποιος χάρτης από το χρήστη, και αν έχει επιλεχθεί τότε φορτώνονται οι διαθέσιμοι στόχοι θέσης στο Dashboard. Όταν επιλέξει ο χρήστης τον επιθυμητό του στόχο, τότε αυτός αποθηκεύεται προσωρινά. Στο κάτω τμήμα, μόλις ο χρήστης πατήσει το κουμπί για την αποστολή του στόχου, διαβάζεται ο αποθηκευμένος στόχος, σχηματίζεται το μήνυμα και το σωστό topic και τέλος αποστέλλεται στο RabbitMQ.

Συνεπώς, ο χρήστης μέσω αυτών των σχετικά απλών flows μπορεί να ελέγχει το ρομπότ είτε μέσω τηλεχειρισμού είτε μέσω επιθυμητών στόχων θέσης, απευθείας από το Dashboard. Αν όμως επιθυμεί ο έλεγχος να γίνεται άμεσα από το Node-RED, δεν έχει παρά να αντικαταστήσει τους κόμβους για τα κουμπιά του Dashboard, με έναν απλό κόμβο εισαγωγής μηνύματος (inject) που θα πυροδοτήσει τη διαδικασία.

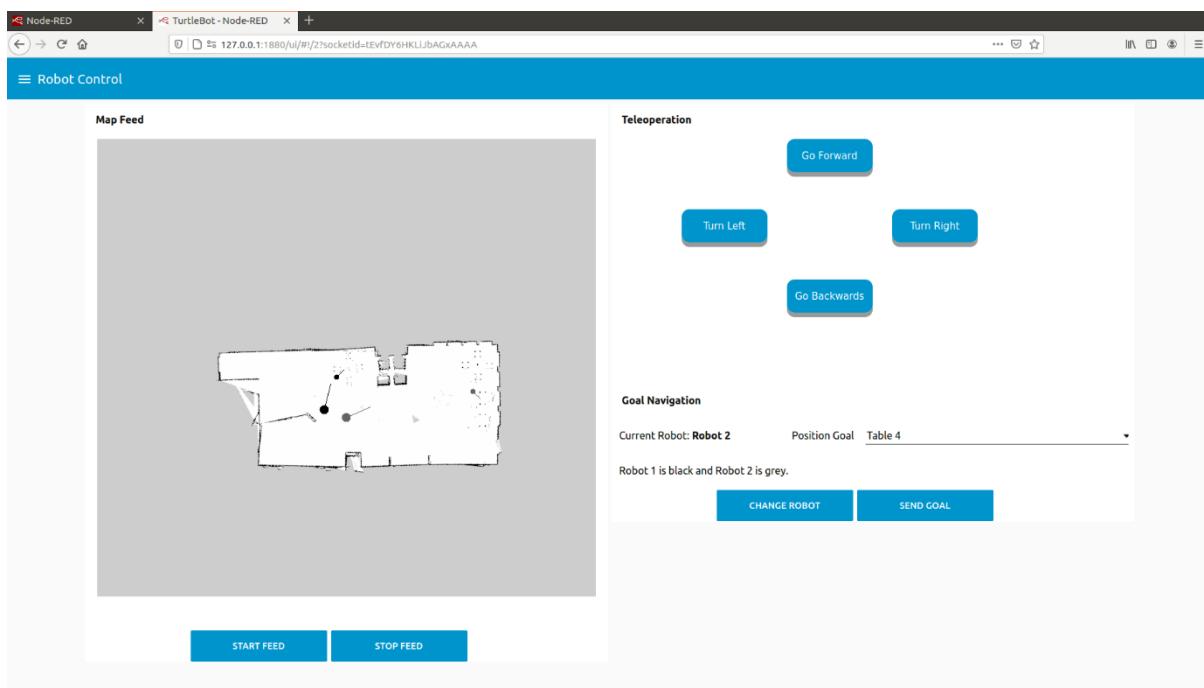
Τέλος, θα παρουσιαστούν οι δύο βασικές λειτουργίες του SLAM και της πλοήγησης από την οπτική γωνία του χρήστη. Κατά τη διαδικασία του SLAM, ο μόνος τρόπος ελέγχου του ρομπότ είναι μέσω τηλεχειρισμού, αφού δεν έχει σχηματιστεί ακόμα ο χάρτης για να μπορούν να δοθούν στόχοι θέσης. Στο ακόλουθο στιγμιότυπο του Dashboard (εικόνα 5.19), φαίνεται η εικόνα του χάρτη που φτάνει στο Node-RED όσο τρέχει το SLAM. Πέρα από τη θέση και τον προσανατολισμό του ρομπότ, φαίνονται με μαύρο τα εμπόδια που έχουν εντοπιστεί, με άσπρο η τρέχουσα σάρωση του λέιζερ και με ανοιχτό γκρι ο ελεύθερος χώρος.



Εικόνα 5.19: Λειτουργία SLAM μέσω Dashboard

Κατά τη διαδικασία της πλοήγησης, και οι δύο τρόποι ελέγχου του ρομπότ είναι διαθέσιμοι. Στο παρακάτω στιγμιότυπο (εικόνα 5.20), φαίνεται η εικόνα του χάρτη για λειτουργία πλοήγησης δύο ρομπότ. Το ένα ρομπότ απεικονίζεται με μαύρο χρώμα, ενώ το άλλο με γκρι, για να ξεχωρίζουν, ενώ πληροφορείται και ο χρήστης ποιο από τα δύο ελέγχει κάθε στιγμή. Παράλληλα, φαίνεται και ο τρόπος που απεικονίζεται ο επιθυμητός στόχος που έχει δοθεί στο ρομπότ, δηλαδή με ένα μικρό κυκλικό δίσκο και μια γραμμή για τον επιθυμητό προσανατολισμό.

## 5.4 Απομακρυσμένο τμήμα του συστήματος



Εικόνα 5.20: Λειτουργία πλοιήγησης για δύο ρομπότ μέσω Dashboard

## Κεφάλαιο 6

# ΣΕΝΑΡΙΑ ΧΡΗΣΗΣ

Στο κεφάλαιο αυτό θα παρουσιαστούν τρία σενάρια χρήσης του συστήματος, όπου αναδεικνύονται οι δυνατότητες του συστήματος σε διάφορες εφαρμογές. Στην Ενότητα 5.1 περιγράφεται το πρώτο σενάριο χρήσης, όπου το ρομπότ λειτουργεί ως σερβιτόρος σε μία καφετέρια. Το δεύτερο σενάριο χρήσης αναλύεται στην Ενότητα 5.2, με το ρομπότ αυτή τη φορά να δρα ως φύλακας σε μία οικία. Τέλος, στην Ενότητα 5.3 παρουσιάζεται το τελευταίο σενάριο χρήσης, όπου δύο ρομπότ χρησιμοποιούνται ως ξεναγοί για δύο διαφορετικές εκθέσεις σε μία γκαλερί.

### 6.1 Coffee Shop

Αρχικά, πρέπει να αναφερθεί ότι ο κόσμοι του Gazebo για κάθε σενάριο χρήσης δημιουργήθηκαν από την αρχή. Δυστυχώς, τα περισσότερα μοντέλα αντικειμένων που παρέχονται από το προσόμοιωτή είναι για εξωτερικά περιβάλλοντα και δεν καλύπτουν τις ανάγκες μας. Για το λόγο αυτό χρησιμοποιήθηκαν δύο dataset μοντέλων αντικειμένων για εσωτερικούς χώρους. Το πρώτο<sup>1</sup> είναι του York University του Τορόντο και περιέχει διάφορα μοντέλα αντικειμένων, όπως ηλεκτρονικών συσκευών, έπιπλα και διακοσμητικά [42]. Το δεύτερο<sup>2</sup> είναι του AWS RoboMaker και περιέχει έπιπλα, συσκευές και άλλα αντικείμενα ενός σπιτιού. Ο κόσμος που δημιουργήθηκε για αυτό το σενάριο χρήσης είναι μία καφετέρια και φαίνεται στην εικόνα 6.1.

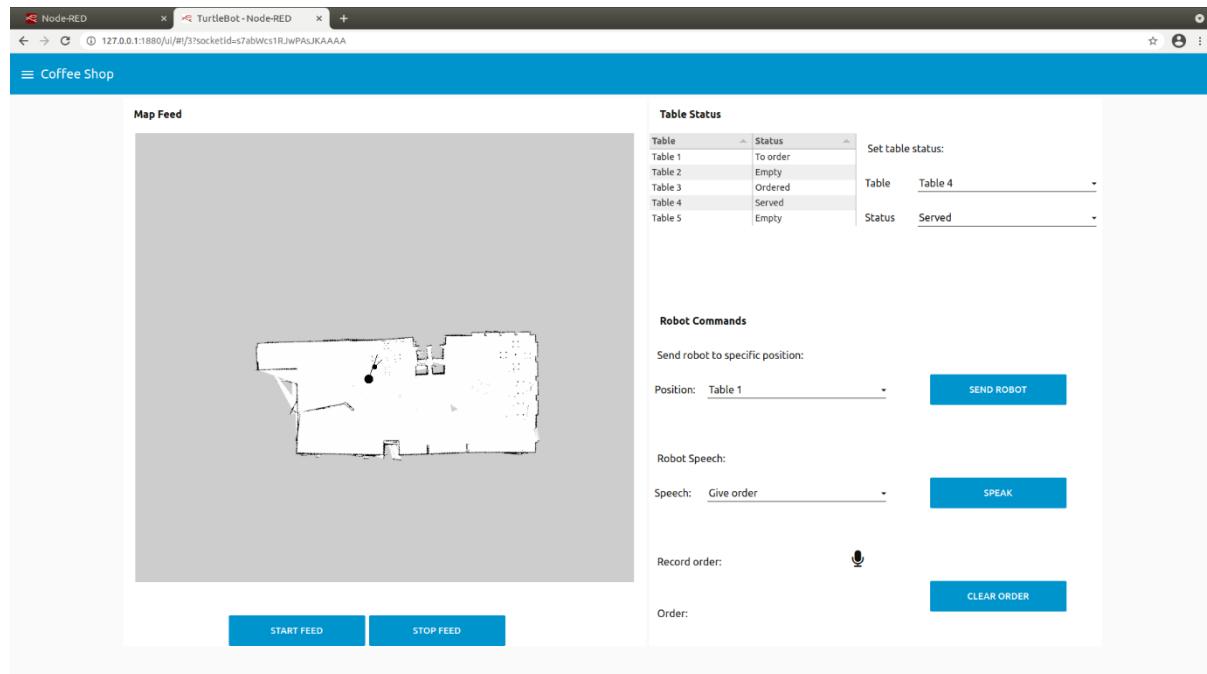


Εικόνα 6.1: Η καφετέρια στο περιβάλλον του Gazebo

<sup>1</sup> <http://data.nvision2.eecs.yorku.ca/3DGEMS/>

<sup>2</sup> <https://github.com/aws-robotics/aws-robomaker-small-house-world>

Σε αυτό το σενάριο, το ρομπότ λειτουργεί ως σερβιτόρος στην καφετέρια. Μέσω του Node-RED Dashboard γίνεται ο χειρισμός του ρομπότ από το χρήστη. Επιπλέον, στο Dashboard εμφανίζεται ένας πίνακας με τις πληροφορίες για την κατάσταση κάθε τραπέζιού, δηλαδή σε ποιο στάδιο της εξυπηρέτησής τους βρίσκονται, για παράδειγμα αν είναι άδειο ή αν ο πελάτης περιμένει να παραγγείλει. Ο χρήστης μέσω του Dashboard στέλνει το ρομπότ στα τραπέζια για να εξυπηρετεί τους πελάτες και μεταβάλλει την κατάσταση του τραπέζιού ανάλογα. Στην εικόνα 6.2 φαίνεται ένα στιγμιότυπο κατά τη διάρκεια της χρήσης.



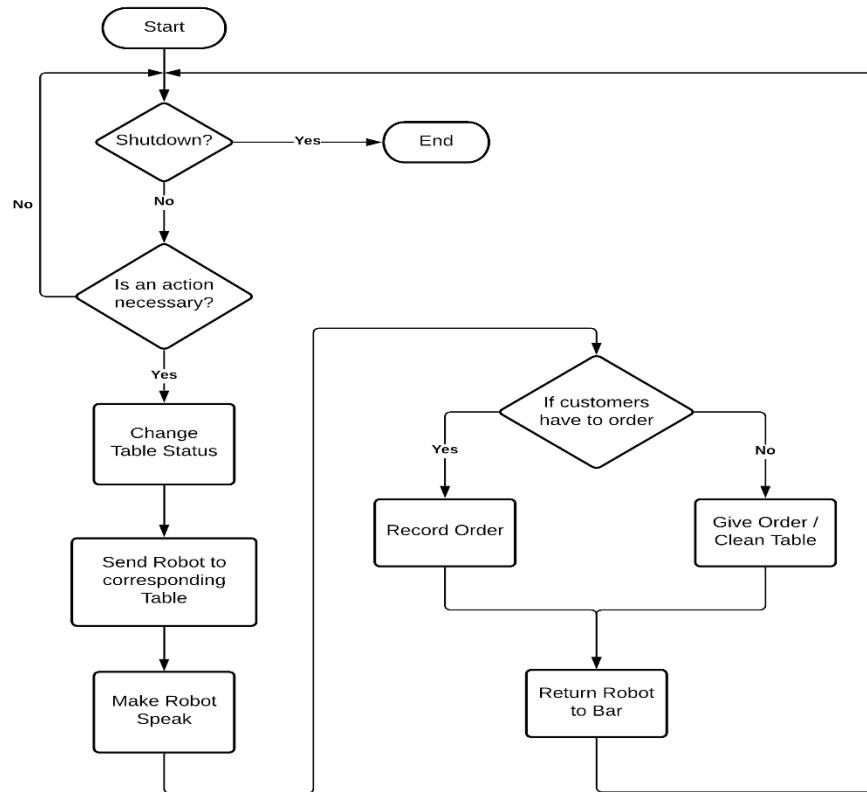
Εικόνα 6.2: Σενάριο χρήσης Coffee Shop

Για την εξυπηρέτηση των πελατών, ο χρήστης μπορεί να στείλει το ρομπότ στο τραπέζι, είτε για να καταγράψει την παραγγελία τους, είτε για να παραδώσει την παραγγελία, είτε για να μαζέψει τα πράγματα. Η παραγγελία καταγράφεται από το μικρόφωνο, με το οποίο είναι εξοπλισμένο το ρομπότ και στέλνεται απευθείας στο Node-RED. Παράλληλα, μπορεί να δώσει εντολή στο ρομπότ να εκφωνήσει κάποια ομιλία με το ηχείο του, για να επικοινωνήσει με τους πελάτες. Η διαδικασία χειρισμού της εφαρμογής για την εξυπηρέτηση των πελατών φαίνεται στο διάγραμμα ροής της εικόνας 6.3.

Πρέπει να σημειωθεί ότι επειδή το Gazebo δεν έχει αισθητήρες ηχείου και μικροφώνου, οι λειτουργίες αυτές προσομοιώνονται μέσω των αντίστοιχων κόμβων του Dashboard.

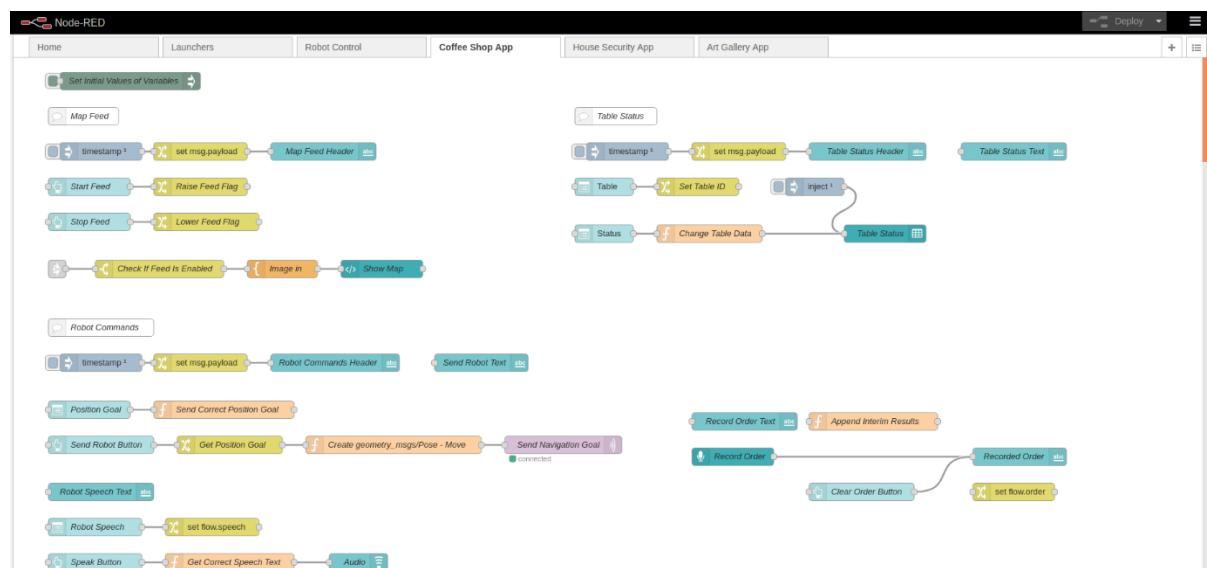
Συνεπώς, με τη βοήθεια του συστήματος η διαδικασία εξυπηρέτησης των πελατών μπορεί να γίνει μέσω του ρομπότ. Σε πολλά μαγαζιά χρησιμοποιούνται ήδη ρομπότ ως σερβιτόροι και ιδιαίτερα στην Ασία, με ικανοποιητικά αποτελέσματα, χωρίς βέβαια να ξεπερνάνε τις επιδόσεις ενός ανθρώπινου σερβιτόρου. Με κάποιες περαιτέρω βελτιώσεις, το σύστημα μπορεί να αυτοματοποιηθεί ακόμα περισσότερο, σε βαθμό που ο χρήστης δε θα χρειάζεται καν να παρεμβαίνει στη λειτουργία, όπως για παράδειγμα με τη χρήση ενός κουμπιού που θα βρίσκεται στο τραπέζι, που όταν πατιέται το ρομπότ θα πηγαίνει αυτόματα στο τραπέζι, για να εξυπηρετήσει τους πελάτες.

## ΚΕΦΑΛΑΙΟ 6. ΣΕΝΑΡΙΑ ΧΡΗΣΗΣ



Εικόνα 6.3: Διάγραμμα ροής της εφαρμογής Coffee Shop

Στην εικόνα 6.4 φαίνεται το Node-RED flow για την εφαρμογή του Coffee Shop. Το flow είναι χωρισμένο σε τρία μέρη και το καθένα αντιστοιχεί σε ένα group του Dashboard. Το πάνω αριστερά τμήμα λαμβάνει την εικόνα του χάρτη και την απεικονίζει, το πάνω δεξιά λαμβάνει την εικόνα του ρομπότ και την εκτελεί τις εντολές που δίνει ο χρήστης στο ρομπότ.



Εικόνα 6.4: To Node-RED flow της εφαρμογής Coffee Shop

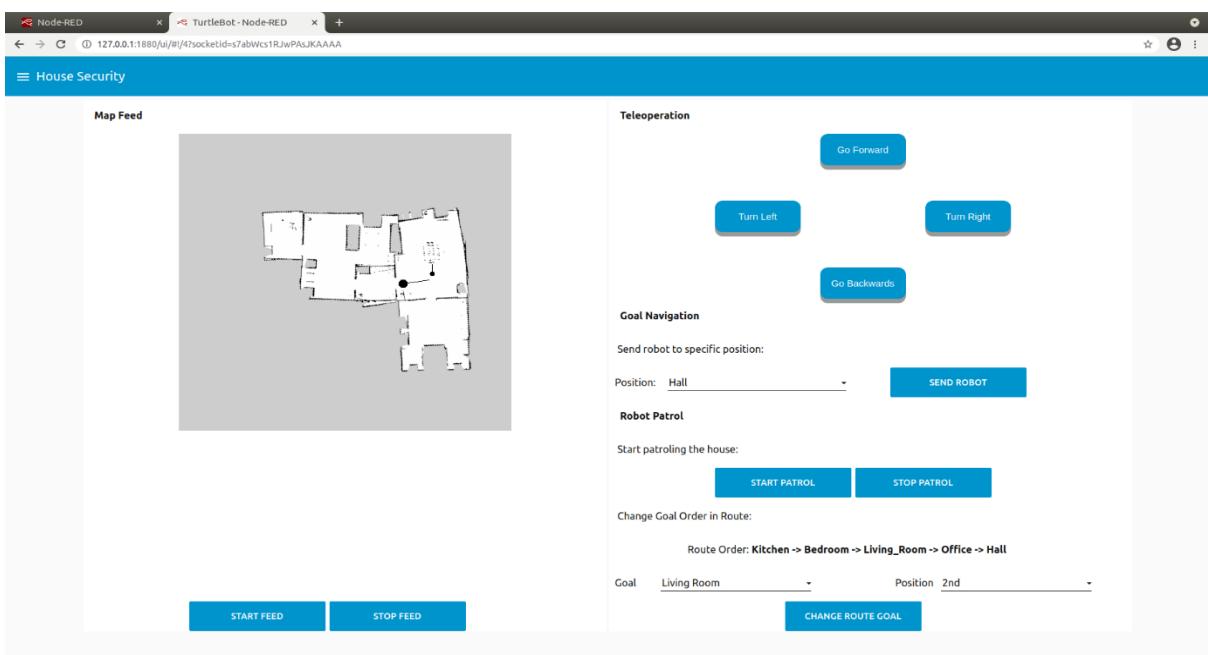
## 6.2 House Security

Ο κόσμος που δημιουργήθηκε για αυτό το σενάριο χρήσης είναι ένα σπίτι και φαίνεται στην εικόνα 6.5.



Εικόνα 6.5: Το σπίτι στο περιβάλλον του Gazebo

Ο ρόλος του ρομπότ σε αυτό το σενάριο είναι να περιπολεί το σπίτι όσο λείπει ο χρήστης και να εντοπίζει τους διαρρήκτες. Η βασική του λειτουργία είναι να ακολουθεί μία διαδρομή και αν εντοπίσει διαρρήκτη να ενημερώνει αμέσως το χρήστη. Στην εικόνα 6.6 φαίνεται ένα στιγμιότυπο κατά τη διάρκεια της χρήσης.



Εικόνα 6.6: Σενάριο χρήσης House Security

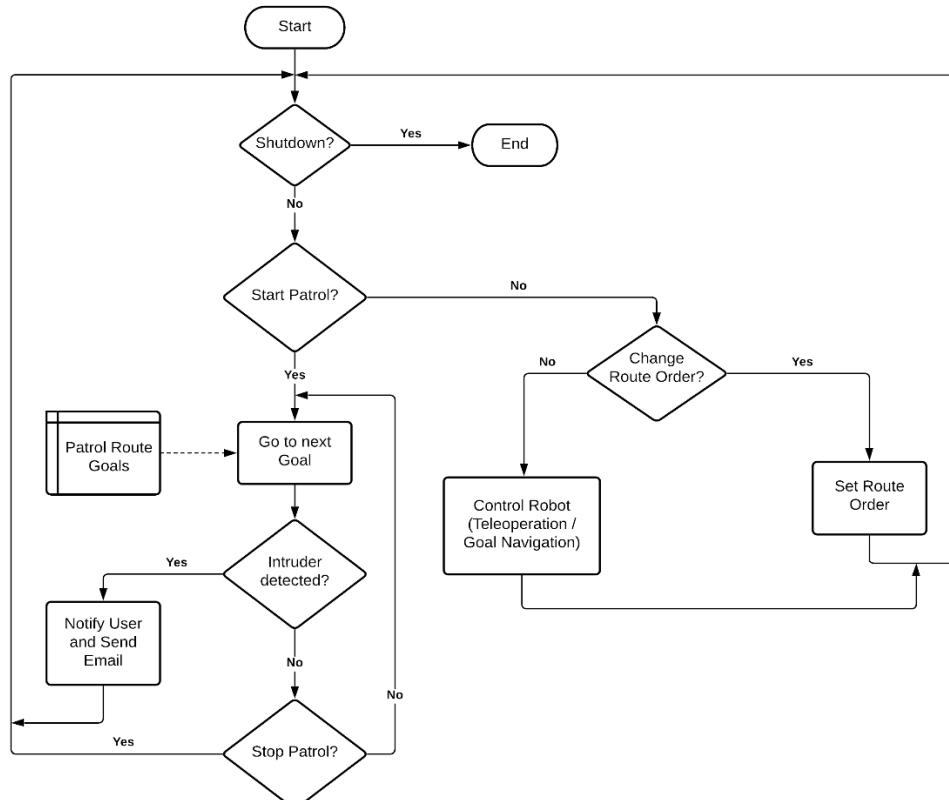
## ΚΕΦΑΛΑΙΟ 6. ΣΕΝΑΡΙΑ ΧΡΗΣΗΣ

Η παρακολούθηση μιας διαδρομής, δηλαδή μιας ακολουθίας στόχων θέσης, είναι μία πρόσθετη λειτουργία που υλοποιήθηκε για αυτό το σενάριο χρήσης. Ουσιαστικά, οι στόχοι της διαδρομής είναι αποθηκευμένοι στο Node-RED και μόλις το ρομπότ φτάσει επιτυχώς στον τωρινό του στόχο, το Node-RED λαμβάνει το μήνυμα ότι έφτασε το ρομπότ επιτυχώς στο στόχο του, και δίνεται ο επόμενος στόχος στο ρομπότ.

Στην περίπτωση που το ρομπότ εντοπίσει κάποιο διαρρήκτη, εμφανίζεται ειδοποίηση στο Dashboard, ενώ συγχρόνως στέλνεται και email στο χρήστη, όπου αναφέρεται ότι εντοπίστηκε διαρρήκτης, και η ακριβής ημερομηνία και ώρα του εντοπισμού. Παράλληλα, ο χρήστης μπορεί αν επιθυμεί, να αλλάξει την σειρά των στόχων στη διαδρομή, ή αν θέλει να ελέγξει κάποιο συγκεκριμένο σημείο του σπιτιού, μπορεί να αναλάβει τον πλήρη έλεγχο του ρομπότ μέσω τηλεχειρισμού ή με το να δώσει συγκεκριμένο στόχο στο ρομπότ. Η διαδικασία χειρισμού της εφαρμογής φαίνεται στο διάγραμμα ροής εικόνας 6.7.

Ο εντοπισμός του διαρρήκτη προσομοιώνεται μέσω κουμπιού στο Node-RED. Στην πραγματικότητα θα έπρεπε να λαμβάνεται η εικόνα από την κάμερα του ρομπότ, να εκτελείται αναγνώριση εικόνας (image recognition) για τον εντοπισμό ανθρώπου και στην περίπτωση αυτή να στελνόταν ένα μήνυμα στο Node-RED.

Επομένως, μέσω αυτού του σεναρίου χρήσης γίνεται εμφανές ότι χρησιμοποιώντας το σύστημά μας, το ρομπότ μπορεί να λειτουργεί ως φύλακας σε ένα σπίτι. Αν συνδυαστεί και με άλλες συσκευές συστημάτων ασφαλείας IoT που ενσωματώνονται πολύ εύκολα στο Node-RED, όπως συναγερμούς και κάμερες, τότε το συνολικό σύστημα θα καλύπτει σε ικανοποιητικό βαθμό την ασφάλεια ενός σπιτιού.



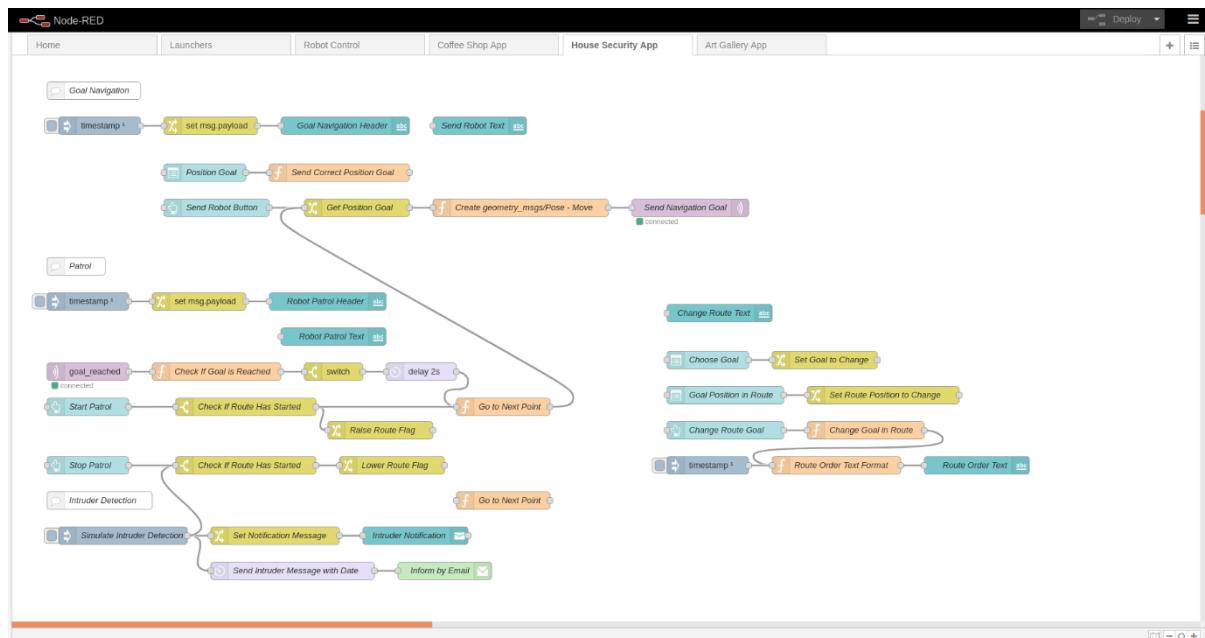
Εικόνα 6.7: Διάγραμμα ροής της εφαρμογής House Security

Στις εικόνες 6.8 και 6.9 φαίνεται το Node-RED flow για την εφαρμογή House Security. Το τμήμα που είναι υπεύθυνο για τη λήψη και την απεικόνιση του χάρτη, αλλά και το τμήμα για την αποστολή των εντολών τηλεχειρισμού του ρομπότ φαίνονται στην εικόνα 6.8.



Εικόνα 6.8: To Node-RED flow της εφαρμογής House Security (1/2)

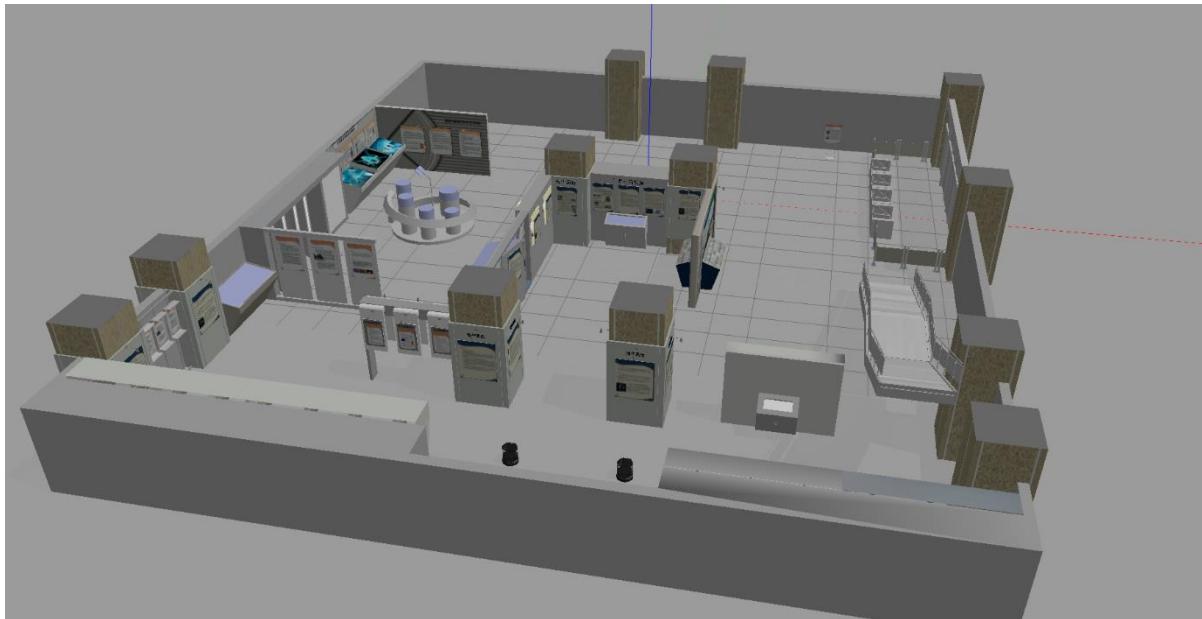
Στην εικόνα 6.9 απεικονίζεται το τμήμα του flow που είναι υπεύθυνο για την αποστολή των επιθυμητών στόχων θέσης στο ρομπότ, το τμήμα που υλοποιεί τη λειτουργία για την παρακολούθηση μιας διαδρομής από το ρομπότ, καθώς επίσης και το τμήμα που ειδοποιεί το χρήστη σε περίπτωση που εντοπιστεί διαρρήκτης.



Εικόνα 6.9: To Node-RED flow της εφαρμογής House Security (2/2)

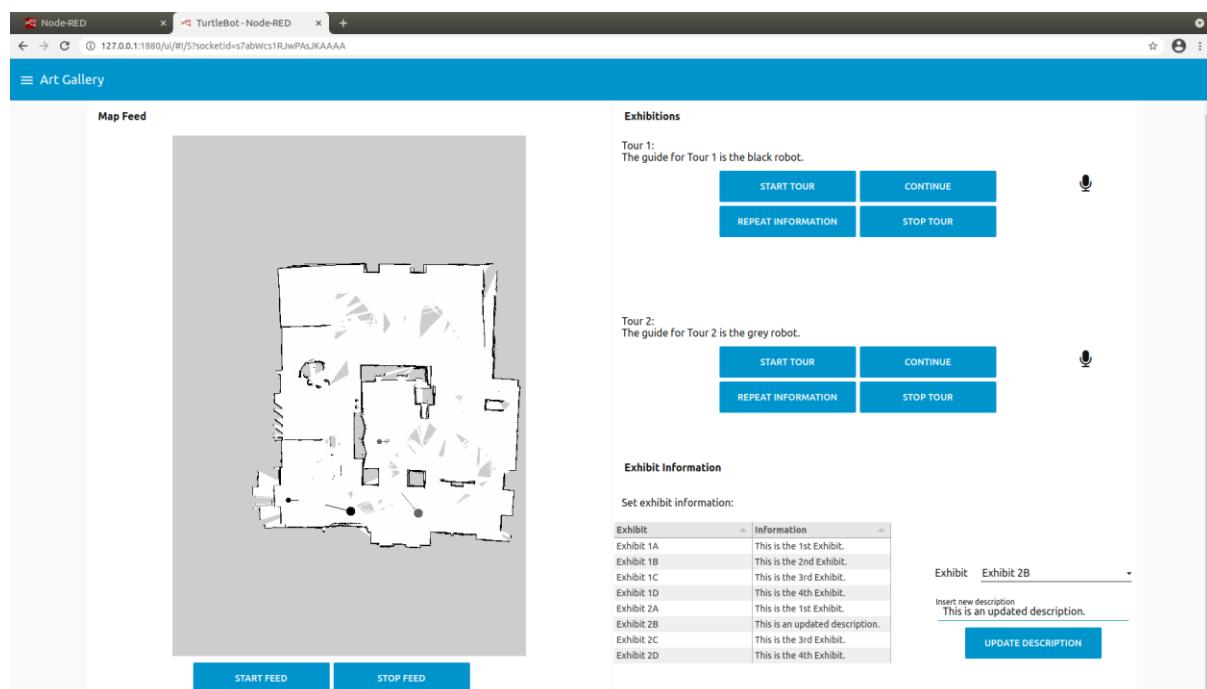
### 6.3 Art Gallery

Για το τελευταίο σενάριο χρήσης δημιουργήθηκε μία γκαλερί, η οποία φαίνεται στην εικόνα 6.10.



Εικόνα 6.10: Η γκαλερί στο περιβάλλον του Gazebo

Σε αυτό το σενάριο, ο χρήστης έχει μία γκαλερί τέχνης, όπου φιλοξενεί συχνά διάφορες εκθέσεις, ενώ μπορεί ταυτόχρονα να υποστηρίξει δύο διαφορετικές. Τα δύο ρομπότ, ένα για κάθε έκθεση, έχουν το ρόλο των ξεναγών και καθοδηγούν τους επισκέπτες στις εκθέσεις. Στην εικόνα 6.11 φαίνεται ένα στιγμιότυπο κατά τη διάρκεια της χρήσης.



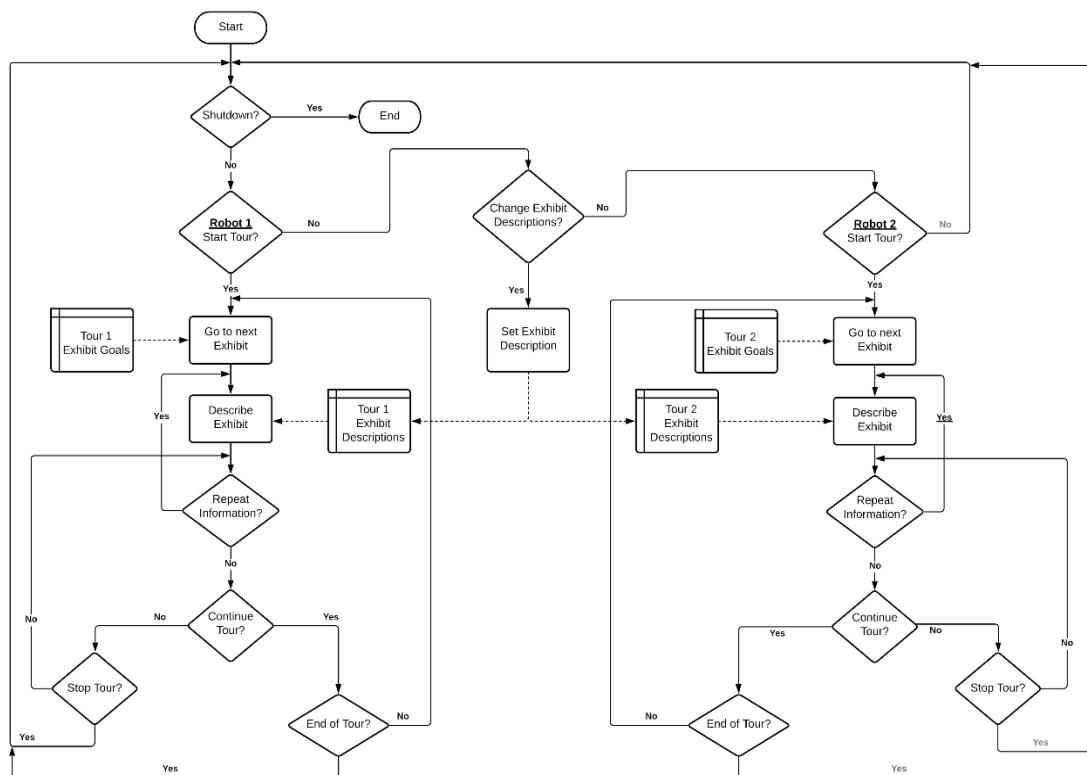
Εικόνα 6.11: Σενάριο χρήσης Art Gallery

Ο χρήστης μέσω του Dashboard βλέπει στο χάρτη τη θέση και τον στόχο κίνησης των ρομπότ, για να γνωρίζει σε ποιο στάδιο της ξενάγησης βρίσκονται, έχει τη δυνατότητα να ελέγξει την επόμενή τους κίνηση, καθώς επίσης και να αλλάξει το κείμενο που εκφωνούν τα ρομπότ για κάθε έκθεμα των δύο εκθέσεων.

Κάθε ρομπότ δέχεται φωνητικές εντολές από τους επισκέπτες για την επόμενή του κίνηση. Αρχικά, θα λάβει την εντολή “Start” για να ξεκινήσει η ξενάγηση. Έπειτα, οι επισκέπτες το ακολουθούν σε κάθε έκθεμα, όπου τους εκφωνεί πληροφορίες σχετικά με αυτό. Στη συνέχεια, οι επισκέπτες μπορούν να του δώσουν εντολή να επαναλάβει τις πληροφορίες για το έκθεμα, να συνεχίσουν στο επόμενο έκθεμα της ξενάγησης ή να τερματίσει η ξενάγηση. Η διαδικασία χειρισμού της εφαρμογής φαίνεται στο διάγραμμα ροής της εικόνας 6.12.

Όπως και στο πρώτο σενάριο, τα μικρόφωνα και τα ηχεία προσομοιώνονται μέσω του Node-RED. Ο κόμβος που υλοποιεί το μικρόφωνο παρόλο που υποστηρίζει λειτουργία αναγνώρισης ομιλίας, δεν παρουσιάζει αρκετά καλά αποτελέσματα, και για αυτό παρέχεται και η δυνατότητα ελέγχου μέσω του Dashboard.

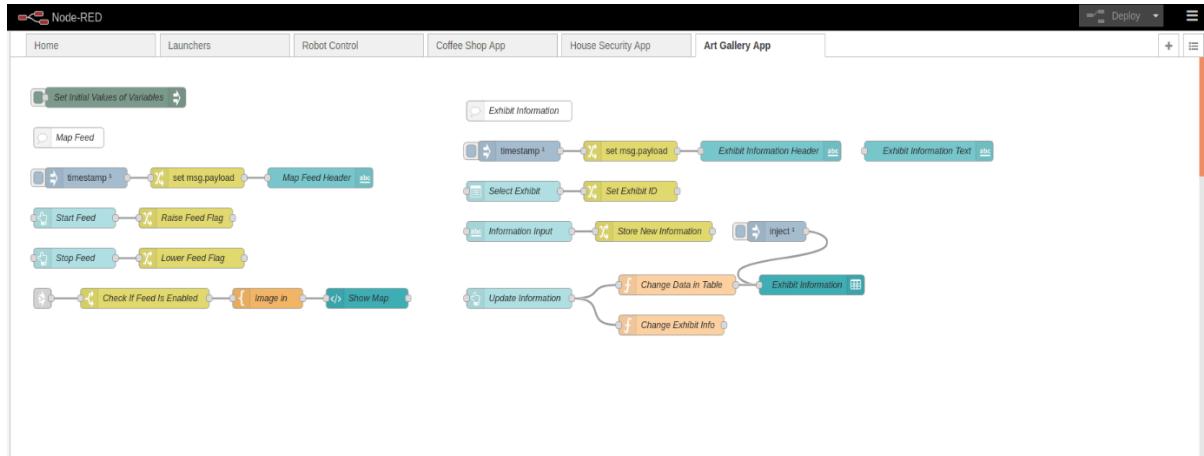
Με αυτό το σενάριο χρήσης γίνεται εμφανές, ότι το σύστημα είναι εξίσου βολικό είτε η εφαρμογή χρησιμοποιεί ένα ρομπότ είτε περισσότερα. Τα ρομπότ μπορούν να λειτουργούν ως ξεναγοί αποτελεσματικά και χωρίς να απαιτείται συνεχής έλεγχος από το χρήστη, ειδικά στην περίπτωση που χρησιμοποιείται κάποια πιο αποδοτική μέθοδος για την αναγνώριση της ομιλίας των επισκεπτών.



Εικόνα 6.12: Διάγραμμα ροής της εφαρμογής Art Gallery

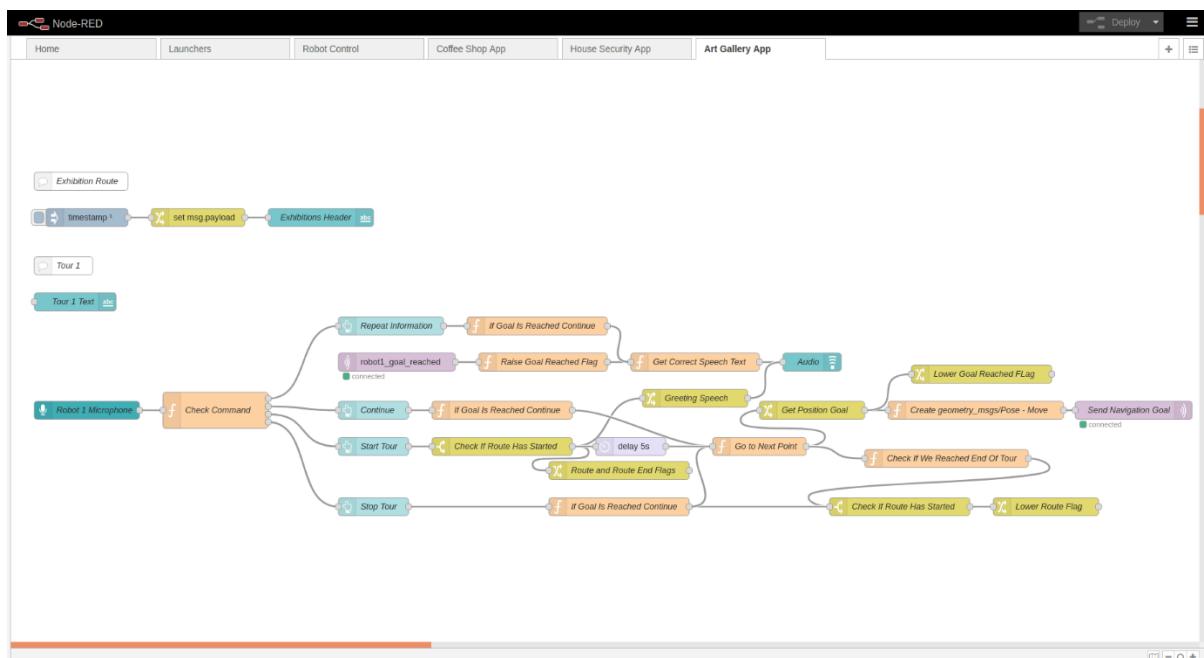
## ΚΕΦΑΛΑΙΟ 6. ΣΕΝΑΡΙΑ ΧΡΗΣΗΣ

Στις εικόνες 6.13 και 6.14 φαίνεται το Node-RED flow για την εφαρμογή Art Gallery. Συγκεκριμένα, στην εικόνα 6.13 φαίνεται το τμήμα που είναι υπεύθυνο για τη λήψη και την απεικόνιση του χάρτη, αλλά και το τμήμα που διαχειρίζεται το κείμενο που θα εκφωνεί το ρομπότ για κάθε έκθεμα.



Εικόνα 6.13: To Node-RED flow της εφαρμογής Art Gallery (1/2)

Στην εικόνα 6.14 απεικονίζεται το τμήμα του flow που είναι υπεύθυνο για την εκτέλεση όλων των εντολών που μπορεί να λάβει το ρομπότ είτε από το μικρόφωνο είτε από τα κουμπιά του Dashboard. Στην εικόνα φαίνεται το μέρος του flow που αφορά τη λειτουργία του ενός ρομπότ, διότι και για τα δύο οι κόμβοι είναι ίδιοι.



Εικόνα 6.14: To Node-RED flow της εφαρμογής Art Gallery (2/2)

Σε αυτό το σημείο θα γίνει μία σύντομη αναφορά στα πρόσθετα πακέτα του Node-RED που χρησιμοποιήθηκαν τόσο στην ανάπτυξη του γενικού συστήματος, όσο και στην υλοποίηση των σεναρίων χρήσης. Όλα τα πακέτα αυτά μπορούν να εγκατασταθούν πολύ εύκολα μέσω του περιβάλλοντος του Node-RED, απλά εισάγοντας το όνομα τους στην αντίστοιχη επιλογή αναζήτησης. Τα πακέτα είναι τα εξής:

- **node-red-contrib-config:** Ένας κόμβος για τον ορισμό μεταβλητών κατά την αρχικοποίηση των ροών.
- **node-red-contrib-moment:** Κόμβοι για την μορφοποίηση ημερομηνίας και ώρας από μηνύματα χρονικής σήμανσης (timestamp).
- **node-red-dashboard:** Κόμβοι για τη δημιουργία ενός dashboard που είναι ενσωματωμένο στο Node-RED.
- **node-red-node-base64:** Ένας κόμβος για την ακωδικοποίηση και αποκωδικοποίηση μηνυμάτων σε μορφή Base64.
- **node-red-node-email:** Κόμβοι για την αποστολή και λήψη email.
- **node-red-node-ui-microphone:** Ένας κόμβος που παρέχει τη λειτουργία μικροφώνου μέσω του Dashboard.
- **node-red-node-ui-table:** Ένας κόμβος που παρουσιάζει δεδομένα σε μορφή πίνακα μέσω του Dashboard.

## Κεφάλαιο 7

# ΣΥΜΠΕΡΑΣΜΑΤΑ & ΜΕΛΛΟΝΤΙΚΗ ΕΡΓΑΣΙΑ

Στο κεφάλαιο αυτό, παρουσιάζονται τα συμπεράσματα που μπορούν να προκύψουν από την υλοποίηση και χρήση του συστήματος της διπλωματικής εργασίας, καθώς και μελλοντικές επεκτάσεις για τη βελτίωση της διαδικασίας ανάπτυξης των εφαρμογών αλλά και της εμπειρίας του χρήστη. Στην Ενότητα 6.1 αναλύονται τα συμπεράσματα που εξάγονται από την ανάπτυξη και χρήση του συστήματος. Τα προβλήματα που προέκυψαν κατά την υλοποίηση του συστήματος περιγράφονται στην Ενότητα 6.2. Τέλος, στην Ενότητα 6.3 γίνεται μία αναφορά στην πιθανή μελλοντική εργασία, με σκοπό τη βελτίωση του συστήματος.

### 7.1 Συμπεράσματα

Τα κυβερνοφυσικά συστήματα, δηλαδή συστήματα που συνδυάζουν υπολογιστικές δυνατότητες και αλληλεπίδραση με το περιβάλλον, βιώνουν μία δραστική ανάπτυξη τα τελευταία χρόνια. Παρόμοια ανάπτυξη εντοπίζεται και στο Internet of Things, όπου καθημερινές συσκευές συνδέονται στο Διαδίκτυο και μπορούν να ανταλλάξουν δεδομένα. Η ενσωμάτωση των δύο αυτών τεχνολογιών θα επιφέρει σημαντικά πλεονεκτήματα, αφού πέρα από τα δεδομένα που συλλέγουν οι συσκευές του IoT, θα υπάρχει και σημαντική επεξεργαστική ισχύς, αλλά και επίδραση στο περιβάλλον, δυνατότητες που προσφέρουν τα κυβερνοφυσικά συστήματα.

Στα πλαίσια της παρούσας διπλωματικής εργασίας, υλοποιήθηκε ένα σύστημα που έχει ως στόχο την ενσωμάτωση ρομποτικών συστημάτων στο IoT. Τα προβλήματα που ακλήθηκε να αντιμετωπίσει για να επιτευχθεί αυτό ήταν δύο.

Το ROS είναι το πιο διαδεδομένο μεσολειτουργικό σύστημα για ανάπτυξη ρομποτικών εφαρμογών. Παρόλο που το ROS παρέχει πολλές διευκολύνσεις για την υλοποίηση εφαρμογών, περιορίζει τη διαχείριση του ρομπότ σε τοπικό δίκτυο και δεν υπάρχει δυνατότητα ελέγχου μέσω Διαδικτύου. Για να επιτευχθεί η επικοινωνία μεταξύ του ROS και του απομακρυσμένου υπολογιστή, χρησιμοποιείται ο μεσολαβητής μηνυμάτων RabbitMQ. Δεδομένου ότι τα μηνύματα του ROS δεν είναι συμβατά με κάποιο από τα πρωτόκολλα μηνυμάτων που χρησιμοποιεί το RabbitMQ, τα μηνύματα αυτά μετατρέπονται σε μηνύματα AMQP, που μπορεί να μεταβιβάσει ο μεσολαβητής, αλλά και αντιστρόφως, στην περίπτωση της λήψης ενός μηνύματος που προορίζεται για το ROS. Με αυτόν τον τρόπο, όλες οι λειτουργίες του ROS γίνονται διαθέσιμες και εκτός του τοπικού δικτύου.

Το δεύτερο πρόβλημα έγκειται στο γεγονός ότι για να ενσωματωθεί επιτυχώς ένα ρομποτικό σύστημα στο IoT, είναι σημαντικό ο χρήστης να έχει τη δυνατότητα να αναπτύξει την εφαρμογή του, χωρίς να διαθέτει εκτεταμένες τεχνικές γνώσεις ρομποτικής ή προγραμματισμού. Για αυτό το λόγο, για την ανάπτυξη των εφαρμογών το σύστημα

χρησιμοποιεί το Node-RED, ένα εργαλείο που επιτρέπει τη δημιουργία εφαρμογών για IoT συστήματα μέσω γραφικής διεπαφής. Επομένως, με τη χρήση του Node-RED απλοποιείται σημαντικά η διαδικασία, ενώ παράλληλα το ρομπότ μπορεί να επικοινωνεί πολύ εύκολα με διάφορες συσκευές IoT.

Η αρχιτεκτονική του συστήματος στηρίζεται σε ένα μοντέλο client-server τριών επιπέδων. Η εφαρμογή που θα αναπτύξει ο χρήστης, καθώς και το ρομπότ, αποτελούν τους client, ενώ ο μεσολαβητής μηνυμάτων παίρνει το ρόλο του server, και καθιστά δυνατή την επικοινωνία μεταξύ των δύο client. Ως αποτέλεσμα, το σύστημα είναι κλιμακούμενο και μπορεί να υποστηρίξει εύκολα πολλούς χρήστες, στο οποίο συμβάλλει βέβαια και το RabbitMQ, αφού κάθε χρήστης εμπεριέχεται σε διαφορετικό virtual host και το εκάστοτε περιβάλλον είναι απομονωμένο από τα υπόλοιπα.

Τέλος, μέσω των σεναρίων χρήσης που υλοποιήθηκαν γίνονται ξεκάθαρές οι δυνατότητες που προσφέρει το σύστημα για την απομακρυσμένη ανάπτυξη εφαρμογών. Ο χρήστης μπορεί εύκολα μέσω του Node-RED Dashboard να δημιουργήσει το χάρτη του χώρου του με χρήση της SLAM λειτουργίας. Έπειτα, μέσω των λειτουργιών του τηλεχειρισμού και της αποστολής επιθυμητών στόχων θέσης, μπορεί να χειρίζεται με ευκολία το ρομπότ του. Παράλληλα, με τις διευκολύνσεις που παρέχει το Node-RED, μπορεί να υλοποιήσει οποιαδήποτε λειτουργία επιθυμεί, συνδυάζοντας τόσο το ρομπότ, όσο και άλλες συσκευές IoT, και όλα αυτά χωρίς να βρίσκεται στο ίδιο δίκτυο με το ρομπότ.

## 7.2 Προβλήματα

Κατά τη διάρκεια εκπόνησης της διπλωματικής εργασίας, εμφανίστηκαν διάφορα προβλήματα που καθυστέρησαν σημαντικά τη διαδικασία υλοποίησης του συστήματος. Ένα μεγάλο μέρος του χρόνου σπαταλήθηκε στην εγκατάσταση του λειτουργικού συστήματος Ubuntu 16.04 το οποίο εμφάνιζε προβλήματα συμβατότητας με ένα εξάρτημα του υπολογιστή, καθώς επίσης και στα εργαλεία λογισμικού που χρησιμοποιούνται από το σύστημα, τα περισσότερα από τα οποία, εμφάνισαν κάποιο πρόβλημα συμβατότητας κατά την εγκατάστασή τους.

Ένα άλλο πρόβλημα προέκυψε κατά την υλοποίηση της λειτουργίας πολλαπλών ρομπότ. Το ROS έχει αναλυτική περιγραφή για τις περισσότερες λειτουργίες που προσφέρει, ενώ παράλληλα υπάρχουν και πολλές αναρτήσεις χρηστών για την επίλυση διάφορων ζητημάτων. Παρ' όλα αυτά, στην προκειμένη περίπτωση το documentation για την αρχικοποίηση κόμβων εντός namespace και το σωστό ορισμό των topics, που απαιτούνται για να λειτουργούν συγχρόνως πολλαπλά ρομπότ, ήταν ελλιπές, και οι ελάχιστες αναρτήσεις ήταν από προηγούμενες εκδόσεις του ROS, οπότε δε μπορούσαν να βοηθήσουν. Ως αποτέλεσμα δαπανήθηκε αρκετός χρόνος για την υλοποίηση της συγκεκριμένης λειτουργίας.

Τέλος, ένα ακόμα πρόβλημα ήταν ότι δε μπορούσε να χρησιμοποιηθεί το πρωτόκολλο μηνυμάτων AMQP, που χρησιμοποιεί ο μεσολαβητής RabbitMQ, για την επικοινωνία με το Node-RED, όπως αναφέρθηκε και στην Ενότητα 4.3. Δυστυχώς, παρόλο που έχουν δημιουργηθεί μερικά πακέτα από την κοινότητα, όσα δοκιμάστηκαν κατά την υλοποίηση του συστήματος παρουσίαζαν πρόβλημα. Για αυτό το λόγο, επιλέχθηκε η χρήση του MQTT, το οποίο υποστηρίζεται από το ίδιο το Node-RED, ενώ μπορεί και εύκολα να υποστηριχθεί

από το RabbitMQ. Η μόνη λειτουργία που δε θα μπορούσε να υλοποιηθεί είναι αν έπρεπε να κληθεί κάποια υπηρεσία του ROS μέσω RPC, αλλά στα πλαίσια της υλοποίησης του συστήματος δε χρειάστηκε κάτι τέτοιο.

### 7.3 Μελλοντικές Επεκτάσεις

Στη συνέχεια παρουσιάζονται μερικές επεκτάσεις για τη περαιτέρω βελτίωση της διαδικασίας ανάπτυξης των εφαρμογών αλλά και της εμπειρίας του χρήστη.

#### 1. Βελτίωση του τρόπου επιλογής επιθυμητών στόχων θέσης

Με την τρέχουσα κατάσταση του συστήματος, η επιλογή στόχων θέσης γίνεται από μία λίστα στόχων που έχουν οριστεί στο Node-RED. Η λογική είναι, ότι ο χρήστης κατά το στήσιμο του συστήματος θα εισάγει μερικές επιθυμητές θέσεις που θέλει να πηγαίνει συχνά το ρομπότ, και αν θέλει να το στείλει σε κάποια άλλη θέση, θα χρησιμοποιήσει τη μέθοδο του τηλεχειρισμού. Παρόλο που στα πλαίσια της διπλωματικής εργασίας με αυτόν τον τρόπο καλύπτονται οι ανάγκες του χειρισμού του ρομπότ, μία αλλαγή που θα βελτίωνε την εμπειρία του χρήστη είναι η επιλογή του επιθυμητού στόχου θέσης απευθείας μέσω του χάρτη που εμφανίζεται στο Dashboard. Ο χρήστης απλά θα επέλεγε τη θέση στο χάρτη που θα ήθελε να στείλει το ρομπότ και η εντολή θα μεταφερόταν απευθείας στο ROS.

#### 2. Απεικόνιση των εικόνων της κάμερας του ρομπότ

Μία πρόσθετη λειτουργία που θα βοηθούσε τόσο κατά τη διαδικασία του SLAM, όσο και σε διάφορες εφαρμογές των χρηστών, είναι η απεικόνιση των εικόνων που τραβάει η κάμερα του ρομπότ στο Dashboard. Με αυτόν τον τρόπο, ο χρήστης θα έχει καλύτερη αντίληψη του χώρου στον οποίο βρίσκεται το ρομπότ, και δε θα βασίζεται μόνο στην πληροφορία που του παρέχει ο χάρτης του Dashboard.

#### 3. Γλοποίηση πακέτου Node-RED για χρήση του πρωτοκόλλου AMQP

Όπως εξηγήθηκε παραπάνω, στο σύστημα χρησιμοποιείται το πρωτόκολλο μηνυμάτων MQTT για την επικοινωνία μεταξύ του Node-RED και του μεσολαβητή. Για την πλήρη υποστήριξη των λειτουργιών που παρέχει το ROS, θα ήταν χρήσιμη η υλοποίηση ενός πακέτου Node-RED, που θα επιτρέπει την αποστολή και λήψη μηνυμάτων χρησιμοποιώντας το πρωτόκολλο AMQP. Η χρήση του AMQP θα προσφέρει πολλά πλεονεκτήματα, όπως για παράδειγμα τη δυνατότητα απευθείας κλήσης κάποιας υπηρεσίας του ROS μέσω RPC, καλύτερη ασφάλεια, περισσότερη αξιοπιστία στη μεταφορά των μηνυμάτων κ.ά.

# Βιβλιογραφία

- [1] 5 Ways the Internet of Things Impacts Your Daily Life. 2016. Online at <https://www.business2community.com/big-data/5-ways-internet-things-impacts-daily-life-01447717>
- [2] Minerva, Roberto, Abyi Biru, and Domenico Rotondi. "Towards a definition of the Internet of Things (IoT)." *IEEE Internet Initiative* 1.1 (2015): 1-86.
- [3] Ray, Partha Pratim. "Internet of robotic things: Concept, technologies, and challenges." *IEEE Access* 4 (2016): 9489-9500.
- [4] Atzori, Luigi, Antonio Iera, and Giacomo Morabito. "Understanding the Internet of Things: definition, potentials, and societal role of a fast evolving paradigm." *Ad Hoc Networks* 56 (2017): 122-140.
- [5] Patel, Keyur K., and Sunil M. Patel. "Internet of things-IOT: definition, characteristics, architecture, enabling technologies, application & future challenges." *International journal of engineering science and computing* 6.5 (2016).
- [6] Sikder, Amit Kumar, et al. "A survey on sensor-based threats to internet-of-things (iot) devices and applications." *arXiv preprint arXiv:1802.02041* (2018).
- [7] Billard, A., Calinon, S., Dillmann, R., Schaal, S.: Robot programming by demonstration. In: Siciliano, B., Khatib, O. (eds.) Springer Handbook of Robotics, pp. 1371–1394. Springer, Heidelberg (2008).
- [8] Chong, Jonathan Wun Shiung, et al. "Robot programming using augmented reality: An interactive method for planning collision-free paths." *Robotics and Computer-Integrated Manufacturing* 25.3 (2009): 689-701.
- [9] Gordon, Michal, Edith Ackermann, and Cynthia Breazeal. "Social robot toolkit: Tangible programming for young children." *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction Extended Abstracts*. 2015.
- [10] Salman, Tara, and Raj Jain. "A survey of protocols and standards for internet of things." *arXiv preprint arXiv:1903.11549* (2019).
- [11] Yuan, Michael. 'Getting to know MQTT'. IBM Developer. 2017. Online at <https://developer.ibm.com/articles/iot-mqtt-why-good-for-iot/>
- [12] 'AMQP 0-9-1 Model Explained'. VMWare, Inc. Online at <https://www.rabbitmq.com/tutorials/amqp-concepts.html>
- [13] Boshernitsan, Marat, and Michael Sean Downes. *Visual programming languages: A survey*. Computer Science Division, University of California, 2004.
- [14] Bézivin, Jean. "In search of a basic principle for model driven engineering." *Novatica Journal, Special Issue* 5.2 (2004): 21-24.

- [15] "The Nao4All platform: Program a robot without technical knowledge". R4A. 2019. Online at <https://r4a.issel.ee.auth.gr/the-nao4all-platform/>
- [16] Quigley, Morgan, et al. "ROS: an open-source Robot Operating System." *ICRA workshop on open source software*. Vol. 3. No. 3.2. 2009
- [17] Fox, Dieter. "Kld-sampling: Adaptive particle filters and mobile robot localization." *Advances in Neural Information Processing Systems (NIPS)* (2002).
- [18] Breazeal, Cynthia. "Social robots for health applications." *2011 Annual international conference of the IEEE engineering in medicine and biology society*. IEEE, 2011.
- [19] Devetiyarova, M., E. Zueva, and S. Ostreiko. "Medical robots." (2017).
- [20] Tierney, Michael J., et al. "Surgical robotic tools, data architecture, and use." U.S. Patent No. 6,331,181. 18 Dec. 2001.
- [21] Sinharay, Arijit, et al. "A novel approach to unify robotics, sensors, and cloud computing through IoT for a smarter healthcare solution for routine checks and fighting epidemics." *International Internet of Things Summit*. Springer, Cham, 2015.
- [22] Chen, Min, et al. "Cp-robot: Cloud-assisted pillow robot for emotion sensing and interaction." *International Conference on Industrial IoT Technologies and Applications*. Springer, Cham, 2016.
- [23] Lekova, Anna, et al. "Braininspired IoT Controlled Walking Robot Big-Foot." *Advances in Science, Technology and Engineering Systems Journal* 4.3, 2019: 220-226.
- [24] Katona, Jozsef, et al. "Electroencephalogram-based brain-computer interface for internet of robotic things." *Cognitive Infocommunications, Theory and Applications*. Springer, Cham, 2019. 253-275.
- [25] Leeb, Robert, et al. "Towards independence: a BCI telepresence robot for people with severe motor disabilities." *Proceedings of the IEEE* 103.6 (2015): 969-982.
- [26] Xiong, Zhang, et al. "Intelligent transportation systems for smart cities: a progress review." *Science China Information Sciences* 55.12 (2012): 2908-2914.
- [27] Meyer, Jonas, et al. "Autonomous vehicles: The next jump in accessibilities?" *Research in transportation economics* 62 (2017): 80-91.
- [28] Roy Chowdhury, Ankur. "IoT and Robotics: a synergy." *PeerJ Preprints* 5 (2017): e2760v1
- [29] Gerla, Mario, et al. "Internet of vehicles: From intelligent grid to autonomous cars and vehicular clouds." *2014 IEEE world forum on internet of things (WF-IoT)*. IEEE, 2014.
- [30] Kumar, Swarun, et al. "Carspeak: a content-centric network for autonomous driving." *ACM SIGCOMM Computer Communication Review* 42.4 (2012): 259-270.
- [31] Ghazzai, Hakim, Hamid Menouar, and Abdullah Kadri. "On the placement of UAV docking stations for future intelligent transportation systems." *2017 IEEE 85th Vehicular Technology Conference (VTC Spring)*. IEEE, 2017.

- [32] Menouar, Hamid, et al. "UAV-enabled intelligent transportation systems for the smart city: Applications and challenges." *IEEE Communications Magazine* 55.3 (2017): 22-28.
- [33] Crick, Christopher, et al. "Rosbridge: Ros for non-ros users." *Robotics Research*. Springer, Cham, 2017. 493-504.
- [34] Koubaa, Anis, Maram Alajlan, and Basit Qureshi. "ROSLink: bridging ROS with the internet-of-things for cloud robotics." *Robot Operating System (ROS)*. Springer, Cham, 2017. 265-283.
- [35] Zhao, Rui, et al. "A Cross-Platform Communication Mechanism for ROS-Based Cyber-Physical System." (2020).
- [36] "Introducing ROStful: ROS over RESTful web services". Ben Kehoe. 2014. Online at <http://www.ros.org/news/2014/02/introducing-rostful-ros-over-restful-web-services.html>
- [37] Zubrycki, Igor, Marcin Kolesiński, and Grzegorz Granosik. "Graphical programming interface for enabling non-technical professionals to program robots and internet-of-things devices." *International Work-Conference on Artificial Neural Networks*. Springer, Cham, 2017.
- [38] Rahul, R., Ashwin Whitchurch, and Madhav Rao. "An open source graphical robot programming environment in introductory programming curriculum for undergraduates." *2014 IEEE International Conference on MOOC, Innovation and Technology in Education (MITE)*. IEEE, 2014.
- [39] Brunner, Sebastian G., et al. "RAFCON: A graphical tool for engineering complex, robotic tasks." *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016.
- [40] Strawhacker, Amanda, and Marina U. Bers. "'I want my robot to look for food': Comparing Kindergartner's programming comprehension using tangible, graphic, and hybrid user interfaces." *International Journal of Technology and Design Education* 25.3 (2015): 293-319.
- [41] Sapounidis, Theodosios, Stavros Demetriadis, and Ioannis Stamelos. "Evaluating children performance with graphical and tangible robot programming tools." *Personal and Ubiquitous Computing* 19.1 (2015): 225-237.
- [42] A.Rasouli, J.K. Tsotsos. "The Effect of Color Space Selection on Detectability and Discriminability of Colored Objects." *arXiv preprint arXiv:1702.05421* (2017).