ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ

ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΠΡΟΣ ΤΗΝ ΑΥΞΗΣΗ ΤΗΣ ΕΥΦΥΪΑΣ ΚΑΙ ΤΗΝ ΑΥΤΟΟΡΓΑΝΩΣΗ ΣΥΣΚΕΥΩΝ ΕΝΤΟΠΙΣΜΟΥ ΘΕΣΗΣ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

ΚΟΥΤΡΟΥΜΠΟΥΧΟΥ ΚΩΝΣΤΑΝΤΙΝΟΥ

Επιβλέπων : Κουκούτσης Ηλίας
Επίκ. Καθηγητής Ε.Μ.Π.

Αθήνα, Νοέμβριος 2017

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΈΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ
ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΠΡΟΣ ΤΗΝ ΑΥΞΗΣΗ ΤΗΣ ΕΥΦΥΪΑΣ ΚΑΙ ΤΗΝ ΑΥΤΟΟΡΓΑΝΩΣΗ ΣΥΣΚΕΥΩΝ ΕΝΤΟΠΙΣΜΟΥ ΘΕΣΗΣ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

ΚΟΥΤΡΟΥΜΠΟΥΧΟΥ ΚΩΝΣΤΑΝΤΙΝΟΥ

Επιβλέπων : Κουκούτσης Ηλίας
Επίκ. Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 7η Νοεμβρίου 2017.

.....	.....	.....
Ηλίας Κουκούτσης	Κωνσταντίνος Παπαοδυσσεύς	Γεώργιος Καμπουράκης
Επίκ. Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Αναπλ. Καθηγητής Ε.Μ.Π.

Αθήνα, Νοέμβριος 2017

.....

ΚΟΥΤΡΟΥΜΠΟΥΧΟΣ ΚΩΝΣΤΑΝΤΙΝΟΣ

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Στην εργασία αυτή έγινε μία πρώτη εξέταση της δυνατότητας αύξησης της συνολικής ευφυΐας των συστημάτων που χρησιμοποιούνται για την υποστήριξη εφαρμογών της πράξης που χρησιμοποιούν υπολογιστικά υποσυστήματα, τα οποία είναι ενσωματωμένα σε γεωγραφικά διεσπαρμένες, κινητές μονάδες, όπως είναι τα οχήματα μεταφορών, τα εμπορευματοκιβώτια κλπ. Σήμερα χρησιμοποιούνται ευρέως συστήματα εντοπισμού θέσης των κινητών αυτών μονάδων μέσω του συστήματος GPS και συστήματα παρουσίασης της θέσης των μονάδων αυτών σε χάρτες. Η αύξηση της ευφυΐας επιχειρήθηκε προς τρεις κύριους άξονες: (i) Την αύξηση της υπολογιστικής δυνατότητας των συστημάτων που είναι ενσωματωμένα στις κινητές μονάδες, (ii) την αυτοοργάνωση σε ασύρματα τηλεπικοινωνιακά υποδίκτυα θεματικά συσχετισμένων υποσυστημάτων που είναι ενσωματωμένα σε κοντινές κινητές μονάδες και (iii) τη μελέτη των αναγκών και των απαιτήσεων αρχιτεκτονικής δεδομένων και προγραμμάτων για τη δημιουργία ενός συστήματος-ομπρέλας, ικανού να παρακολουθήσει και να ελέγξει με πολύ αποτελεσματικό τρόπο ένα σημαντικό αριθμό από θεματικά διαφοροποιημένα, γεωγραφικά διεσπαρμένα συστήματα ενσωματωμένα στις προαναφερθείσες κινητές μονάδες. Στο πλαίσιο των τριών αυτών κατευθυντηρίων αξόνων: (α) Παρουσιάστηκε μία μεθοδολογία επιλογής ενσωματωμένων συστημάτων του εμπορίου, τα οποία βασίζονται σε μικροεπεξεργαστές σχετικά εξελιγμένους, αλλά χαμηλού κόστους και εξετάστηκαν οι δυνατότητες ενός πρώτου αριθμού από συστήματα του τύπου αυτού. (β) Εξετάστηκαν μερικές μεθοδολογίες και πρωτόκολλα αυτοοργάνωσης σε τηλεπικοινωνιακά υποδίκτυα ομοτίμων σταθμών (peer-to-peer networks) των συστημάτων που είναι ολοκληρωμένα σε κοντινές κινητές μονάδες και χρησιμοποιήθηκε το σύστημα Chord για την ανάπτυξη προγραμμάτων και για πρακτικές δοκιμές με έναν μικρό αριθμό από κινητά υποσυστήματα βασισμένα σε Arduino UNO και Arduino Nano. (γ) Έγινε μία πρώτη εξέταση προβλημάτων που προκύπτουν στην προσπάθεια ανάπτυξης συστήματος-ομπρέλας, το οποίο μπορεί να παρακολουθήσει ένα σημαντικό αριθμό από θεματικά διαφοροποιημένες και γεωγραφικά διεσπαρμένες κινητές μονάδες και μία κατ' αρχήν πρόταση χρήσης ειδικής, εξελιγμένης αρχιτεκτονικής δομής του συστήματος-ομπρέλας, η οποία έχει ονομαστεί “Τελεολογική Δομή” και βασίζεται στη χρήση κατάλληλων μεταδεδομένων, προκειμένου να αντιμετωπίσει πολλά από τα σχετικά προβλήματα που παρουσιάζονται στην πράξη.

Λέξεις κλειδιά: αυτοοργάνωση, συσκευές εντοπισμού θέσης, ευφυή υπολογιστικά συστήματα, γεωγραφικά διεσπαρμένες κινητές μονάδες, ευφυΐα συστημάτων με κινητές μονάδες

Abstract

In this thesis a first attempt at increasing the intelligence of systems used for monitoring and controlling a considerable number of thematically variant, geographically dispersed transport units (such as transport vehicles, TEUs etc.) is presented. For that purpose, microcontroller-based modules, integrated in the transport units and capable of communicating to a central Base Station, are necessary. Today, simple integrated systems, embedded in the transport units and used for providing the location of these units by using the Global Positioning System (GPS), are widely used. The approach for increasing the intelligence and capability of the corresponding supporting systems includes work towards the following: (i) Increasing the capability of the microcontroller-based systems embedded in the transport units, (ii) self-organizing the systems which are embedded in thematically correlated, adjacent transport units in peer-to-peer radio frequency networks and (iii) studying the architectural requirements for building a platform that can be used as an umbrella system for monitoring and controlling of geographically dispersed transport units in thematically diverse transport operations. In this context, the following work has been done: (a) A process for selecting embedded systems, based on more advanced and capable, but still affordable microcontrollers, has been proposed and the capabilities of a number of commercially available embedded systems of this type have been examined. (b) Methodologies and protocols for self-organizing peer-to-peer telecommunication networks, connecting nearby transport units through radio frequency links, have been studied and the Chord system has been initially implemented for testing a practical RF peer-to-peer network, which included an Arduino UNO- and three Arduino Nano-based subsystems. (c) The practical problems one faces in designing umbrella systems for monitoring and controlling a considerable number of geographically dispersed transport units for diverse practical transport operations have been initially considered. The complex, diverse and, in most cases, polymorphic data and program core of such systems dictates the use of a novel, advanced system architecture. Finally, the use of a methodology and a corresponding architecture, based on a metadata structure called “Teleological Structure” and suitable for such problems has been proposed.

Keywords: self-organization, positioning devices, intelligent microcontroller-based systems, geographically dispersed transport units, intelligence of transport unit systems

Περιεχόμενα

1 Εισαγωγή

1.1 Αντικείμενο της εργασίας

Η παρούσα εργασία επετελέστη στο εργαστήριο “Συστημάτων Πολυθεματικής και Γεωσυσχετισμένης Πληροφορίας”. Η εργασία αυτή, κάτα έννοια, είναι ανιχνευτική και αποσκοπεί στον προσδιορισμό γενικών και ειδικών παραμέτρων συσκευών και μεθοδολογιών, ώστε να είναι ευκολότερη η σχεδίαση στο μέλλον ευφυών δικτύων παρακολούθησης και ελέγχου γεωγραφικά διεσπαρμένων οντοτήτων. Η ανάγκη αυτή προκύπτει πολύ συχνά στην πράξη σήμερα. Θα αναφερθούν εδώ δύο μόνον σημαντικές περιοχές εφαρμογών της πράξης του τύπου αυτού (αν και αυξάνεται διαρκώς ο αριθμός ανάλογων εφαρμογών που ζητούνται στην πράξη σήμερα). Οι δύο αυτές σημαντικές περιοχές της πράξης είναι οι ακόλουθες:

A) Παρακολούθηση και έλεγχος κινητών υπομονάδων σε περιπτώσεις αποστολών, όπως μεταφορά ασθενών ή και μοσχευμάτων σε νοσοκομεία ή και ιατρικές μονάδες, αποστολές έρευνας και διάσωσης σε περίπτωση πτώσεων αεροπλάνων, κινδύνων ή βύθισης πλοίων, ατυχημάτων ή δυστυχημάτων που αφορούν οχήματα που κινούνται σε δρόμους, αποστολές που είναι αναγκαίες σε περιπτώσεις κρίσεων (όπως είναι οι σεισμοί, οι φυσικές καταστροφές κ.ά.), σε στρατιωτικές αποστολές κλπ.

B) Δεύτερη σημαντική περιοχή εφαρμογής της τεχνολογίας που θα περιγραφεί και των σχετικών μεθοδολογιών είναι οι εθνικές και διεθνείς μεταφορές εμπορευμάτων και επιβατών. Οι λόγοι για αυτό είναι οι εξής:

   1. Στις μεταφορές επιβατών η ομαδοποίηση σε μονάδες μεταφοράς είναι προφανής, δεδομένου ότι οι επιβάτες κινούνται υποχρεωτικά σε οχήματα. Ένα παράδειγμα υποομάδας επιβατικών οχημάτων που θα μπορούσε να θεωρηθεί ως μία ξεχωριστή οντότητα είναι ένα κομβόι από τέσσερα οχήματα τύπου πούλμαν που μεταφέρουν μία μεγάλη ομάδα επιβατών σε ένα τουριστικό ταξίδι σε διάφορες χώρες της Ευρώπης.

   2. Στις συντριπτικό αριθμό περιπτώσεων μεταφορών αγαθών σήμερα χρησιμοποιούνται μονάδες μεταφοράς σαν τα εμπορευματοκιβώτια, τα φορτηγά οχήματα, τα βαγόνια σιδηροδρόμων, τα πλοία, τα αεροπλάνα κλπ. Παράδειγμα υποομάδας μονάδων μεταφορών που θα μπορούσε να θεωρηθεί ως μία ξεχωριστή οντότητα είναι τα βαγόνια που είναι σε ένα συρμό τρένου, τα εμπορευματοκιβώτια που είναι σε ένα πλοίο, ένα κομβόι φορτηγών που μεταφέρουν ένα ή παραπάνω συγκεκριμένα φορτία σε έναν προορισμό κλπ.

Στην πράξη έχει αποδειχθεί ότι, όχι μόνο είναι επιθυμητή αλλά και, σε ιδιαίτερα μεγάλο αριθμό περιπτώσεων, αναγκαία η παρακολούθηση των επιμέρους κινητών μονάδων που αναφέρονται στις προηγούμενες δύο περιοχές εφαρμογών. Μία ομάδα του εργαστηρίου ασχολείται ήδη με τα θέματα της πρώτης από της δύο περιοχές εφαρμογών. Παρότι η παρούσα εργασία αναφέρεται σε ανάγκες και των δύο προαναφερθεισών περιοχών εφαρμογών, θα χρησιμοποιήσουμε παραδείγματα και περιπτώσεις από τη δεύτερη κατηγορία εφαρμογών, τις μεταφορές αγαθών και επιβατών, δεδομένου ότι αυτά θα γίνουν κατά τη γνώμη μας καλύτερα κατανοητά.

Στην παρούσα εργασία κινηθήκαμε προς την κατεύθυνση αύξησης της ευφυΐας του όλου υποστηρικτικού συστήματος παρακολούθησης και ελέγχου γεωγραφικά διεσπαρμένων οντοτήτων, με αρχική επιλογή των ακολούθων:

1. Έρευνα της δυνατότητας αύξησης της υπολογιστικής ισχύος των ανεξάρτητων, ολοκληρωμένων υπολογιστικών σταθμών (embedded systems) που τοποθετούνται συνήθως στις προς παρακολούθηση γεωγραφικά διεσπαρμένες οντότητες. Είναι σαφές ότι η αύξηση υπολογιστικής ισχύος των επιμέρους αυτών σταθμών επιτρέπει την εκτέλεση εξελιγμένων και πιο πολύπλοκων και ογκωδών προγραμμάτων και, επομένως, είναι παράγων που επιτρέπει την αύξηση της ευφυΐας των κινητών σταθμών. Στο πλαίσιο ανίχνευσης δυνατοτήτων του τύπου αυτού εξετάστηκαν συσκευές βασισμένες σε μικροεπεξεργαστές με επιπλέον κριτήρια προτίμησης την υπολογιστική ισχύ (προφανώς), την οικονομία κατανάλωσης ισχύος, τη δυνατότητα προσπέλασης στις λεπτομέρειες σχεδιασμού και λειτουργίας (open source hardware and code), την υποστήριξη μέσω διαδικτύου (user forums and communities) κλπ.

2. Τη δυνατότητα αυτοοργάνωσης κινητών σταθμών, τοποθετημένων σε ομάδες μονάδων μεταφοράς αγαθών ή επιβατών (όπως λεωφορεία, βαγόνια τρένων, εμπορευματοκιβώτια, φορτηγά αυτοκίνητα κλπ.). Όταν οι συγκεκριμένες ομάδες μονάδων μεταφοράς μπορούν να χαρακτηριστούν ως διακριτές οντότητες (τέτοια παραδείγματα αναφέρθηκαν στα αμέσως προηγούμενα) συμφέρει οι σταθμοί που είναι τοποθετημένοι στις μονάδες μεταφοράς της ομάδας να οργανώνονται σε ένα τοπικό τηλεπικοινωνιακό υποδίκτυο. Σκοπός του υποδικτύου αυτού είναι (i) η διευκόλυνση της παροχής υπηρεσιών σχετικών με τις κινητές μονάδες της ομάδας και (ii) η απλοποίηση της εργασίας ενός συνολικού "ευφυούς" δικτύου, το οποίο θα βρίσκεται σε κάποιο κεντρικό σταθμό βάσης και θα παρακολουθεί και ελέγχει σημαντικό αριθμό κινούμενων μονάδων μεταφοράς ή ομάδων μονάδων μεταφοράς.

   Προς αυτήν την κατεύθυνση κινηθήκαμε σε ένα κύριο κλάδο της παρούσας εργασίας όπου εξετάστηκαν πρωτόκολλα αυτοοργάνωσης σταθμών σε τηλεπικοινωνιακά υποδίκτυα. Κατ' αρχήν επελέγη για δοκιμή το πρωτόκολλο Chord, δεδομένου ότι αυτό αποτελεί τη βάση των περισσότερων δικτύων τύπου peer-to-peer στο διαδίκτυο σήμερα. Προκειμένου να αποκτήσουμε μία πρώτη εκτίμηση τέτοιων δικτύων υλοποιήθηκαν προγράμματα για την αυτοοργάνωση τεσσάρων σταθμών, εφοδιασμένων με πομποδέκτες RF συχνοτήτων. Ο ένας από τους σταθμούς αυτούς είναι βασίζεται σε ένα Arduino UNO, ο οποίος περιγράφεται στα επόμενα, ενώ οι υπόλοιποι τρεις σταθμοί βασίζονται σε Arduino Nano. Τα αποτελέσματα της δυνατότητας αυτοοργάνωσης που κατ' αρχήν είναι πολύ ικανοποιητικά και ενθαρρυντικά για τη μελλοντική συνέχεια της εργασίας στην περιοχή αυτή.

3. Εξετάστηκε, αλλά με πολύ απλό και εισαγωγικό τρόπο και χωρίς δοκιμές, η δυνατότητα κατασκευής ενός αντίστοιχου "σταθμού βάσης" για την παρακολούθηση και τον έλεγχο των γεωγραφικά διεσπαρμένων οντοτήτων, όπως αναφέρθηκε ήδη. Εξετάστηκαν κατ' αρχήν τα προβλήματα που παρουσιάζονται σε ένα τέτοιο σύστημα και πιθανή αρχιτεκτονική προγραμμάτων και δεδομένων του συστήματος αυτού που βοηθάει σημαντικά στην επίλυση των συγκεκριμένων προβλημάτων.

Τα προηγούμενα αναπτύσσονται στη συνέχεια της εργασίας με τον τρόπο που περιγράφεται στο επόμενο υποκεφάλαιο.

1.2 Οργάνωση της εργασίας

Η δομή της εργασίας έχει ως εξής.

Στο 2ο κεφάλαιο περιέχονται πληροφορίες σχετικά με το τεχνολογικό, επιστημονικό και θεωρητικό προσκήνιο των αντικειμένων της παρούσας εργασίας, κυρίως πληροφορίες που έχουν βρεθεί μετά από αναζήτηση και έρευνα στο διαδίκτυο. Αρχικά, αναλύεται ο τομέας βελτίωσης ευφυΐας μέσω πιο ισχυρών υπολογιστικών συσκευών (hardware) που θα τοποθετηθούν στις κινητές μονάδες μεταφορών. Συνεχίζουμε με την παρουσίαση σημαντικών ή/και ενδιαφερουσών εργασιών σχετικών με την αυτοοργάνωση σταθμών σε θεματικά υποδίκτυα. Στο τέλος του κεφαλαίου αυτού, εξετάζουμε τη δυνατότητα της αύξησης ευφυΐας μέσω της ολοκλήρωσης των κινητών υπολογιστικών σταθμών σε ένα συνολικό σύστημα-ομπρέλα που, πέραν της παρακολούθησης της κατάστασης των κινητών σταθμών, μπορεί να αποτελέσει την πλατφόρμα για την ανάπτυξη θεματικών, επιμέρους συστημάτων υποστήριξης λήψης αποφάσεων από τους εμπλεκόμενους στις μεταφορές.

Στο 3ο κεφάλαιο παρουσιάζονται το συνολικό πρόβλημα και οι άξονες ανάπτυξης μέσω των οποίων επιχειρείται η αύξηση της ευφυΐας του συνολικού υποστηρικτικού υποσυστήματος και η σχετική βελτίωση των υπηρεσιών στις περιπτώσεις παρακολούθησης και ελέγχου κινητών μονάδων μεταφορών.

Στο 4ο κεφάλαιο δίνεται μία προσέγγιση για βελτίωση παροχής υπηρεσιών στις μεταφορές μέσω της αύξησης της ευφυΐας των υποστηρικτικών συστημάτων, όπου πρώτα περιγράφονται αρχικά κριτήρια επιλογής των υπολογιστικών συσκευών των κινητών σταθμών, μέσω των οποίων μπορεί να γίνει η επιλογή συγκεκριμένου υλικού (hardware). Στο κεφάλαιο αυτό παρουσιάζονται ακόμα οι συσκευές που χρησιμοποιήθηκαν για την υλοποίηση στην πράξη.

Στο 5ο κεφάλαιο γίνεται μία υλοποίηση απλού δικτύου αυτοοργάνωσης με χρήση των δοκιμαστικών συσκευών. Περιγράφεται αρχικά η δομή του δικτύου αυτού και οι βασικές ανάγκες διαδικασιών που απαιτούνται για τη σωστή λειτουργία του. Στη συνέχεια παρουσιάζεται ένας συνολικός αλγόριθμος για την επίλυση του προβλήματος, μετά από πιο λεπτομερή περιγραφή των ιδιαίτερων προβλημάτων του συγκεκριμένου δικτύου και των σχετικών λύσεων. Τέλος, δίνεται μία παρουσίαση μέσω αποτύπωσης επιλεγμένων οθονών (screenshots) ενός πραγματικού, δοκιμαστικού τρεξίματος του αναπτυχθέντος αλγορίθμου. Οι οθόνες αυτές δίνουν στο χρήστη μία εικόνα των τεκταινομένων κατά τη λειτουργία του δικτύου.

Στο 6ο κεφάλαιο δίνεται καταγραφή στοιχείων διαφορετικών πειραματικών τρεξιμάτων με σκοπό να εξεταστούν οι επιδόσεις του όλου συστήματος και οι πιθανές αδυναμίες του αλγορίθμου που υλοποιήθηκε.

Στο 7ο κεφάλαιο παρουσιάζονται τα συμπεράσματα της εργασίας και προτείνονται μελλοντικές εργασίες προς την υλοποίηση των συνιστωσών του συνολικού υποστηρικτικού συστήματος των σχετικών εφαρμογών της πράξης.

2 Προσκήνιο της έρευνας και της τεχνολογίας σήμερα στο συγκεκριμένο τομέα

Στα επόμενα παρατίθενται σύντομα ορισμένα σημαντικά προβλήματα που εμφανίζονται σήμερα σε δίκτυα μεταφορών (σιδηροδρομικές μεταφορικές αλυσίδες, σειρά φορτηγών μεταφορών κλπ.):

- Οι ανάγκες των χρηστών αυξάνονται συνεχώς, αφού η παγκοσμιοποίηση συνεχίζει να επεκτείνεται, γεωγραφικά και θεματικά, με αποτέλεσμα να ζητούνται όλο και περισσότερες διεθνείς και παγκόσμιες μεταφορές αγαθών και επιβατών. Επίσης, δημιουργούνται νέα προϊόντα τα οποία μπορεί να χρειάζονται ειδική μεταχείριση, διαφορετική από αυτές που υπάρχουν μέχρι τώρα.

- Υπάρχουν πολλοί εμπλεκόμενοι φορείς (απλοί επιβάτες, πελάτες που μεταφέρουν τα προϊόντα τους κλπ.), καθένας από τους οποίους έχει διαφορετικές ανάγκες από τη μεταφορά.

Χρειάζεται σε αυτές, αλλά και άλλες περιπτώσεις, να είναι δυνατό κανείς να μπορεί να ικανοποιήσει σε ικανοποιητικό βαθμό κάθε χρήστη των μεταφορικών μέσων. Για να γίνει δυνατό κάτι τέτοιο, πρέπει, μεταξύ άλλων, να κατασκευαστούν κατάλληλα υπολογιστικά συστήματα για την υποστήριξη των αυξημένων απαιτήσεων των χρηστών των μεταφορικών μέσων. Τα συστήματα αυτά πρέπει να έχουν αυξημένη ευφυΐα (δηλαδή να ενσωματωθεί σε αυτά με κατάλληλο τρόπο σημαντικό μέρος της ευφυΐας των σχεδιαστών μηχανικών), προκειμένου να γίνει δυνατή η ικανοποίηση των προαναφερθεισών απαιτήσεων. Σύμφωνα με τις τεχνολογικές τάσεις που υπάρχουν σήμερα και μετά από μία πρώτη εξέταση των απαιτήσεων των χρηστών, προτείνονται οι ακόλουθες μέθοδοι βελτίωσης της ευφυΐας των υποστηρικτικών, υπολογιστικών συστημάτων των δικτύων αυτών:

1. Αύξηση της ισχύος των ολοκληρωμένων υπολογιστικών συσκευών που τοποθετούνται στα επιμέρους οχήματα ή μονάδες μεταφορών (φορτηγά, βαγόνια τρένων, εμπορευματοκιβώτια, κλπ.). Είναι η πρώτη προσέγγιση την οποία κανείς θα μπορούσε να επιχειρήσει, αφού είναι, κατά μία έννοια, η πλέον προφανής. Χρειάζεται, όμως, να επιλεγούν συσκευές που είναι ικανές να ανταποκριθούν στις απαιτήσεις των χρηστών των υπηρεσιών μεταφορών και, ταυτόχρονα, να μην υπερβαίνουν κάποιο όριο κόστους.

2. Αυτοοργάνωση υποομάδων ολοκληρωμένων (στα οχήματα/μονάδες μεταφορών) συσκευών σε τοπικά υποδίκτυα. Με τον τρόπο αυτό υποσύνολα του συνολικού δικτύου ολοκληρωμένων υπολογιστικών συσκευών, τα οποία έχουν παρόμοιες ανάγκες ή απαιτήσεις, μπορούν μέσω μιας ομαδοποίησης, να αντιμετωπιστούν ως ανεξάρτητες οντότητες με ιδιαίτερα χαρακτηριστικά και απαιτήσεις. Αυτό θα επιτρέψει την αποδοτικότερη συνολική οργάνωση του δικτύου, όπως η πράξη έχει αποδείξει: η κάθε επιμέρους υποομάδα/οντότητα αντιμετωπίζεται μεν ως μία οντότητα, αφορά όμως στην πραγματικότητα μία ομάδα οχημάτων/μονάδων μεταφορών. Παραδείγματα: κομβόι φορτηγών οχημάτων που κινούνται σε αυτοκινητοδρόμους και πρέπει να αντιμετωπίζονται ως μία σύνθετη μεταφορική μονάδα, ομάδα βαγονιών τρένων που αφορούν την ίδια επιχείρηση μεταφοράς αγαθών ή επιβατών, ομάδα εμπορευματοκιβωτίων φορτωμένων σε πλοίο και αφορώντων επίσης μία συγκεκριμένη επιχείρηση μεταφοράς κλπ.

3. Ύπαρξη σταθμού βάσης, ο οποίος θα επικοινωνεί με τις συσκευές που είναι ολοκληρωμένες στα μεταφορικά οχήματα/μονάδες μεταφορών και θα αποτελεί πλατφόρμα γεωσυσχετισμένων πληροφοριών, πάνω στην οποία θα εκτελούνται συστήματα προγραμμάτων για τη διαχείριση των στόλων οχημάτων και μεταφορικών μονάδων και την υποστήριξη λήψης αποφάσεων σχετικών με τα οχήματα/μεταφορικές μονάδες. Είναι προφανές ότι ο σταθμός βάσης μπορεί (συνήθως πρέπει) να έχει αυξημένη υπολογιστική ισχύ, ώστε να εκτελεί ταυτόχρονα εξελιγμένα προγράμματα για την υποστήριξη πολύ μεγάλου αριθμού ανεξάρτητων οχημάτων/μονάδων μεταφορών ή σύνθετων μεταφορικών μονάδων (που αποτελούνται από περισσότερα οχήματα/μεταφορικές μονάδες και αντιμετωπίζονται ως διακριτές οντότητες). Ο σταθμός βάσης:

    a) Να αλλάζει, όταν χρειάζεται, τα επιμέρους προγράμματα, τα οποία εκτελούνται στα επιμέρους υπολογιστικά μικροσυστήματα που είναι ολοκληρωμένα στα οχήματα ή μεταφορικές μονάδες, ώστε αυτά να μπορούν, χωρίς τοπική μνήμη μεγάλου μεγέθους, να ικανοποιούν κατά περίπτωση διαφορετικές ανάγκες.

    b) Να μπορεί να αντιμετωπίσει ταυτόχρονα σημαντικό αριθμό προγραμμάτων, τα οποία χειρίζονται δεδομένα που είναι μεγάλου όγκου, πολύπλοκα και πολυσχιδή, και σε πολλές περιπτώσεις πολύμορφα. Είναι προφανές, ότι απαιτείται ειδική οργάνωση και αρχιτεκτονική των δεδομένων και των προγραμμάτων του σταθμού βάσης, προκειμένου να αντιμετωπίσει το είδος αυτό των δεδομένων.

Μία ιδιαίτερα αποδοτική οργάνωση (αρχιτεκτονική δεδομένων και προγραμμάτων), η οποία καλύπτει με πολύ ικανοποιητικό τρόπο την ανάγκη (3b) που αφορά τον σταθμό βάσης, είναι η χρήση της επονομαζόμενης “Τελεολογικής Δομής”. Η δομή αυτή έχει αναπτυχθεί από ομάδα που έχει εργαστεί σε εθνικά και ευρωπαϊκά προγράμματα, και παρουσιάζεται στις εργασίες [1][2][3][4][5][6]. Λόγω όμως των περιορισμένων δυνατοτήτων όγκου εργασίας της διπλωματικής, στην παρούσα εργασία δεν προχωρήσαμε στη συγκεκριμένη εφαρμογή, δεδομένου ότι η περαιτέρω αυτή εργασία είναι ιδιαίτερα εκτεταμένη, αλλά συνοπτικά περιγράφουμε τη δυνατή χρήση των τελεολογικών δομών στις εφαρμογές που μας αφορούν.

2.1 Ολοκληρωμένες, αυτόνομες συσκευές με μικροελεγκτές που επιτρέπουν αυξημένη ευφυΐα σε αντικατάσταση απλών συσκευών εντοπισμού θέσης.

2.1.1 Σύντομη παρουσίαση επιλεγμένων συσκευών βασισμένων σε μικροελεγκτή

Στα επόμενα αναφέρονται ορισμένες ευρέως χρησιμοποιούμενες στην πράξη συσκευές με μικροελεγκτή, οι οποίες θεωρήθηκαν οι πιο χρήσιμες και σχετικές για τον τομέα που ασχοληθήκαμε. Γενικά, στην πράξη, χρησιμοποιείται ο όρος Development Boards (πλακέτες ανάπτυξης) για το χαρακτηρισμό τους, επειδή με αυτές μπορεί να γίνει σχεδιασμός μεγαλύτερων και αυτόνομων συστημάτων. Εμείς προτιμήσαμε τον όρο μικροελεγκτές για να χαρακτηρίσουμε τα development boards με όλες τις αναγκαίες επεκτάσεις υλικού που επιτρέπουν τη χρήση στις εφαρμογές που μας ενδιαφέρουν. Επεκτάσεις του τύπου αυτού (στην πράξη καλούμενες και shields) είναι το υποσύστημα εντοπισμού θέσης μέσω της χρήσης του GPS (Global Positioning System), διάφορα τηλεπικοινωνιακά υποσυστήματα (με χρήση GPS/GPRS, με δορυφορική ζεύξη, με ασύρματη ζεύξη με περιορισμένη εμβέλεια με ανάλογες συσκευές με μικροελεγκτή που βρίσκονται στο εγγύς περιβάλλον κλπ.) και άλλες. Η σειρά παρουσίασης δεν έχει γίνει με κάποιο ειδικό κριτήριο. Στις ακόλουθες παρουσιάσεις δίνονται βασικές πληροφορίες σχετικά με τις επιδόσεις των αντίστοιχων μικροελεγκτών, όπως επίσης και ειδικά χαρακτηριστικά που τις κάνουν να ξεχωρίζουν.

1. Arduino UNO

Η σχετική πλακέτα χρησιμοποιεί το μικροελεγκτή ATmega328 της Atmel [7], ο οποίος έχει τα εξής χαρακτηριστικά:

- 32KB flash program memory
- SRAM 2KB data memory
- EEPROM 1KB data memory
- I/O pins: 23
- clock speed: 16 MHz
- 8-bit αρχιτεκτονική

Η σχετική πλακέτα έχει: [8]

- 14 digital I/O pins
- 6 analog input pins
- operating voltage: 5V
- DC current per I/O pin: 40mA
- DC current: 46.5mA

Το περιβάλλον ανάπτυξης που χρησιμοποιείται για τον προγραμματισμό της πλακέτας αυτής είναι το Arduino IDE.

Πλεονεκτήματα:

- Το πλέον συζητημένο development board. Υπάρχει πολύ υλικό στο διαδίκτυο για αυτό (πολλά λυμένα προβλήματα/εφαρμογές, βιβλιοθήκες με έτοιμες συναρτήσεις για τρόπους και συσκευές επικοινωνίας, όπως το RF24, που είναι βιβλιοθήκη που αφορά το module nRF24L01 2.4GHz wireless transceiver.)
  Αυτό, σε συνδυασμό με το γεγονός ότι η γλώσσα με την οποία προγραμματίζεται κατά κόρον η πλακέτα αυτή είναι η C/C++, καθιστά το συγκεκριμένο σύστημα πολύ χρήσιμο για γρήγορη ανάπτυξη προγραμμάτων/εφαρμογών.

- Ο σχεδιασμός του είναι ανοιχτός, οπότε υπάρχουν Arduino UNO Compatible boards τα οποία έχουν χαμηλότερη τιμή από το αυθεντικό Arduino UNO, της τάξεως των 4-5 ευρώ [9], ενώ το αρχικό κοστίζει περίπου 20 ευρώ [10].

Μειονεκτήματα:

- Η πλακέτα αυτή χρησιμοποιεί επεξεργαστή με σχετικά χαμηλή ταχύτητα λόγω της 8-bit αρχιτεκτονικής του.

2. Raspberry PI 3

Περαιτέρω πληροφορίες για το Raspberry PI 3 υπάρχουν στο [11].

- CPU: 4x 1.2GHz
- αρχιτεκτονική 64-bit
- RAM: 1GB 900MHz
- storage: microSD (<32GB)
- GPIO (General Purpose Input Output): 40pins
- Built-in WiFi, bluetooth 4.1 & bluetooth le (low energy)
- Operating voltage: 5V
- Typical current consumption: ~200-400mA

Από γλώσσες προγραμματισμού οι οποίες χρησιμοποιούνται για την ανάπτυξη εφαρμογών για τη συσκευή αυτή, κυρίαρχη είναι η Python, αλλά μπορούν να χρησιμοποιηθούν και άλλες, όπως η HTML, Jacascript, Java, C, C++, PERL, ERLANG.

Περιβάλλοντα ανάπτυξης που χρησιμοποιούνται για τον προγραμματισμό του Raspberry PI 3 είναι τα εξής:

- Το Operating System Raspbian (ειδικό για τη συσκευή αυτή), το οποίο είναι λειτουργικό βασισμένο στο Debian (Unix-like OS, λειτουργεί σαν UNIX, χωρίς απαραίτητα να έχει πιστοποιηθεί ως προς κάποια προδιαγραφή UNIX). Το Raspbian παρέχει στον χρήστη πολλά εργαλεία για ανάπτυξη προγραμμάτων σε διάφορες γλώσσες.

Εκτός από αυτό, υπάρχουν και άλλα περιβάλλοντα τρίτων, τα οποία κάλλιστα μπορούν να χρησιμοποιηθούν για τον προγραμματισμό της πλακέτας. Μερικά γνωστά τέτοια περιβάλλοντα είναι τα παρακάτω:

- BlueJ, το οποίο είναι ένα δωρεάν ολοκληρωμένο περιβάλλον ανάπτυξης (IDE – Integrated development environment) για προγραμματισμό σε γλώσσα Java.

- Geany, το οποίο αποτελεί IDE για πολλές γλώσσες προγραμματισμού (C, Java, HTML, Python, Perl κ.ά.). Είναι μικρό σε όγκο μνήμης και ελαφρύ όσον αφορά τις απαιτήσεις σε επιδόσεις του υπολογιστή, ο οποίος το τρέχει.
- Node-RED: είναι εφαρμογή που επιτρέπει τον προγραμματισμό βασισμένο στη ροή, με κύριο σκοπό τον προγραμματισμό για το λεγόμενο Internet of Things.
- IDLE, το οποίο αποτελεί περιβάλλον ανάπτυξης για τη γλώσσα προγραμματισμού Python (περιέχεται στο Rapsbian).

Πλεονεκτήματα της πλακέτας αυτής είναι τα εξής:

- Είναι πολύ πιο γρήγορη από το Arduino, παρέχοντας τετραπύρηνο 1.2GHz επεξεργαστή σε συνδυασμό με την 64 bit αρχιτεκτονική. Αυτό την καθιστά ως ίσως τη βέλτιστη λύση αν υπάρξει ανάγκη επίδοσης.
- Έχει επίσης πληθώρα πληροφορίας στο διαδίκτυο, όπως έτοιμες βιβλιοθήκες, βίντεο με επεξηγήσεις διαφόρων εφαρμογών, πειράματα και έτοιμα προγράμματα.

Μειονεκτήματα της πλακέτας αυτής είναι τα εξής:

- Είναι πιο ακριβό από το Arduino, με κόστος της τάξεως των 40ευρώ [12].
- Δεν είναι ανοιχτός ο σχεδιασμός της πλακέτας, οπότε δεν υπάρχουν πιο φθηνές εκδοχές κατασκευασμένες από άλλες εταιρίες.
- Η πλακέτα έχει σχετικά μεγάλη κατανάλωση ρεύματος (~400mA).

3. Raspberry PI Zero

- CPU: 1GHz
- RAM: 512MB
- storage: microSD
- αρχιτεκτονική 32-bit
- 40 GPIO pins
- typical current: 80-200mA

Η σχετική πλακέτα αποτελεί μία πιο φθηνή αλλά πιο αργή έκδοση του Raspberry PI 3.

Πλεονεκτήματά της είναι:

- Η μικρότερη κατανάλωση ρεύματος σε σχέση με το Raspberry PI 3 (το οποίο καταναλώνει ρεύμα της τάξεως των 400mA, ενώ το Raspberry PI Zero καταναλώνει στην περιοχή των 80-200mA).
- Η χαμηλότερη τιμή, ειδικά σε σχέση με το Raspberry PI 3, της τάξεως των 5 ευρώ [13].

Δεν αναφέρονται περισσότερες πληροφορίες για την πλακέτα αυτή, διότι έχουν καταγραφεί για την Raspberry PI 3, η οποία περιγράφηκε αμέσως προηγουμένως.

4. BeagleBone Black

- CPU: 1GHz
- αρχιτεκτονική 32-bit
- RAM: 512MB 800MHz
- flash storage: 4GB 8-bit

- 2x microcontrollers on-board 32-bit
- typical current draw: 300mA

Πλεονέκτημα της BeagleBone Black είναι τα εξής:

- Έχει 2 ενσωματωμένους μικροελεγκτές, οι οποίοι ονομάζονται Programmable Realtime Units (PRUs). Ο κύριος επεξεργαστής της πλακέτας μπορεί να επικοινωνήσει με τους μικροελεγκτές, με αποτέλεσμα να έχουμε από τη μία τη δυνατότητα του άμεσου ελέγχου που προσφέρεται από τους μικροελεγκτές, ενώ ταυτόχρονα έχουμε και χαρακτηριστικά "πραγματικού υπολογιστή", όπως πολλές γλώσσες και WiFi.

- Η πλακέτα δέχεται όλες τις γνωστές γλώσσες προγραμματισμού.

Η τιμή της πλακέτας αυτής είναι της τάξεως των 50 ευρώ [14].

5. pcDuino8 Uno

- CPU: 2GHz 8-core
- αρχιτεκτονική 32-bit
- RAM: 1GB
- storage: microSD
- power supply: 5V / 2A

Η τιμή της πλακέτας αυτής είναι της τάξεως των 42 ευρώ [15].

Πλεονέκτημα του pcDuino8 Uno είναι:
- Έχει τα ίδια pins με το arduino, και μέσω διαδικασίας μπορεί να προγραμματιστεί σαν αυτό, οπότε αποτελεί μία πιο δυνατή έκδοσή του.

Μειονεκτήματα της πλακέτας είναι:
- Παρόλο που μπορεί τελικά να θεωρηθεί ως arduino, δεν παύει να είναι μία ξεχωριστή πλακέτα οπότε μπορεί να παρουσιάσει δικά της μοναδικά προβλήματα. Όμως, το διαδίκτυο δεν είναι αρκετά ενεργό ώστε να υπάρχει πληροφορία με λύσεις σε πολλά τέτοια θέματα.

6. freetronics Goldilocks

- CPU: 20MHz
- αρχιτεκτονική 8-bit
- SRAM: 16KB
- flash: 128KB
- eeprom: 4KB
- current consumption: ~51mA

Η πλακέτα αυτή είναι ένα arduino uno compatible development board, δηλαδή έχει ίδιο σχεδιασμό και εξόδους με την πλακέτα Arduino UNO, στην οποία έχει αλλαχθεί ο μικροελεγκτή με τον ATmega1284P, ο οποίος προσφέρει καλύτερη επίδοση από τον ATmega328, ο οποίος υπάρχει στο Arduino UNO.

Η τιμή της είναι της τάξεως των 70 ευρώ [16].

7. Teensy 3.6

- CPU: 180MHz
- αρχιτεκτονική 32-bit
- flash: 1MB
- ram: 256KB
- eeprom: 4KB
- current consumption: 50mA

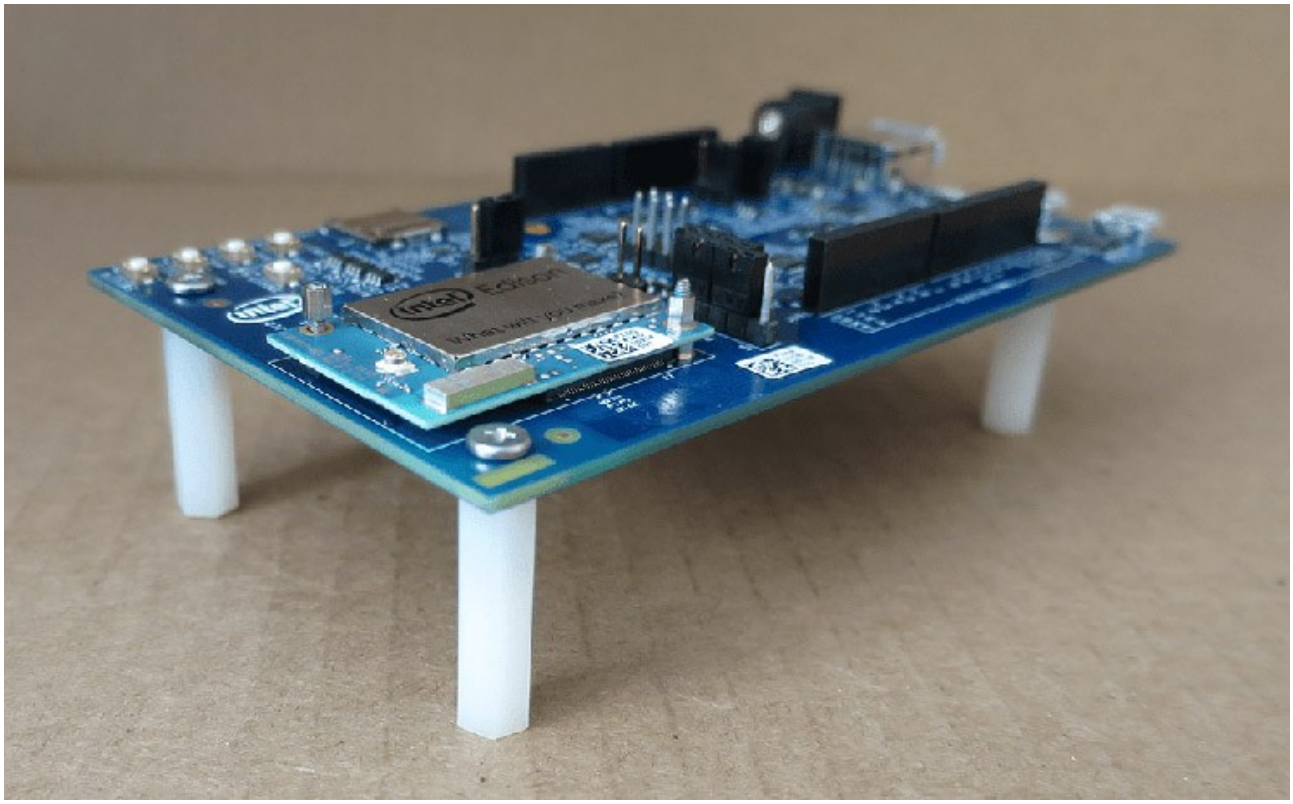
Η σχετική πλακέτα προγραμματίζεται είτε με χρήση συντάκτη (editor) σε γλώσσα προγραμματισμού C, είτε μπορεί να εγκατασταθεί το Teensyduino add-on στον υπολογιστή για το Arduino IDE, το οποίο επιτρέπει τον προγραμματισμό της σαν να ήταν Arduino UNO.

Η τιμή του Teensy 3.6 είναι της τάξεως των 30 ευρώ [17].

8. Intel Edison

CPU: 500MHz
αρχιτεκτονική 32-bit
microcontroller 32-bit, 100MHz
flash storage: 4GB
current consumption: 50mA
Η τιμή της είναι της τάξεως των 100 ευρώ, στην οποία περιέχεται και ο προσαρμογέας ο οποίος φαίνεται στη συνέχεια σε εικόνα [18].

Η πλακέτα αυτή συνδέεται με breakout board ώστε να έχει τα ίδια pins με Arduino. Παρακάτω είναι εικόνα, όπου φαίνεται το breakout board το οποίο προσφέρει εξωτερική δομή των pins, η οποία είναι ίδια με αυτή του Arduino UNO, και στην κοντινή γωνία βιδωμένο το intel edison.



9. Arduino Due

- clock speed: 84MHz
- αρχιτεκτονική **32-bit**
- user applications flash memory: 512KB
- SRAM: 96KB
- 54 digital IO pins
- 12 analog inputs
- operating voltage: 3.3V
- current consumption: 130mA

Η σχετική πλακέτα είναι η πρώτη πλακέτα τύπου Arduino, της οποίας ο μικροελεγκτής είναι βασισμένος σε 32-bit αρχιτεκτονική. Το γεγονός αυτό, σε συνδυασμό με τη μεγαλύτερη RAM και μνήμη που προσφέρονται από το μικροελεγτή της πλακέτας, την καθιστά μία πιο δυνατή εναλλακτική, αν δεν αρκεί η επίδοση του Arduino Uno.

Όπως και για το Arduino Uno, ο σχεδιασμός του Due είναι ανοιχτός, οπότε υπάρχουν φθηνότερες εκδόσεις του, με κόστος της τάξεως των 16 ευρώ [19], ενώ το αυθεντικό κοστίζει περίπου 36 ευρώ [20].

10. Arduino Nano

- microcontroller: ATmega328
- flash memory: 32KB
- SRAM: 2KB
- clock speed: 16 MHz
- EEPROM: 1KB
- operating voltage: 5V

- DC current per I/O pin: 40mA
- Analog I/O pins: 8
- Digital I/O pins: 22
- power consumption: 19mA

Το Arduino Nano αποτελεί μία συμπαγή έκδοση του γνωστού Arduino Uno. Μία από τις σημαντικές διαφορές, σε σχέση με το Uno, είναι ότι δεν παρέχει έξοδο 5V, αλλά μόνο 3.3V. Αυτό στην εργασία μας δεν ενοχλεί, διότι τουλάχιστον για το nRF24L01 η προτεινόμενη τάση είναι 3V οπότε η έξοδος των 3.3V είναι επαρκής. Τα pins είναι αρσενικά (σε αντίθεση με το Arduino Uno, το οποίο έχει θηλυκά pins), γεγονός το οποίο μπορεί να δυσκολέψει τον πειραματισμό και την ανάπτυξη κάποιου προγράμματος, όμως με απλή σύνδεσή του σε ένα breadboard έχουμε την ίδια δομή με το Arduino Uno. Το nRF24L01 που πρόκειται να χρησιμοποιηθεί για την υλοποίηση απλού αυτοοργανούμενου δικτύου έχει επίσης αρσενικά pins, οπότε στη συγκεκριμένη περίπτωση χρησιμοποιούμε απλά female to female jumper cables, χωρίς την ανάγκη προσθήκης breadboard.

Όπως και τα άλλα Arduinos, έχει ανοιχτό σχεδιασμό, με αποτέλεσμα να βρίσκονται πολύ πιο φθηνές εναλλακτικές, της τάξεως των 3 ευρώ [21], ενώ το αυθεντικό κοστίζει περίπου 20 ευρώ [22].

2.1.2 Σύντομη παρουσίαση επιλεγμένων σχετικών πομποδεκτών

Παρακάτω καταγράφονται ορισμένες συσκευές πομποδέκτη οι οποίες θεωρήθηκαν ως καλύτερες επιλογές. Γενικά, οι πληροφορίες που προσφέρονται για κάθε συσκευή μπορεί αρχικά να φαίνονται ελλιπείς σε ορισμένα σημεία, όπως στη κατηγορία της εμβέλειας μπορεί να μην αναφέρεται λεπτομερώς η ισχύς πομπού, η ευαισθησία δέκτη κλπ. Αυτό συνέβη διότι για τις συσκευές αυτές δεν μπόρεσαν να βρεθούν αναλυτικά φύλλα δεδομένων από τα οποία μπορούν να παρθούν τέτοιες πληροφορίες. Όμως, υπάρχει παντού η απαραίτητη πληροφορία για την αξιολόγηση της συσκευής για εφαρμογές που μας αφορούν, δηλαδή οι καταναλώσεις, η τιμή, η τελική εμβέλεια και ο ρυθμός δεδομένων.

- NRF24L01L Long Range Wireless Module R2 (1.1KM, PA+SMA)

Η τιμή της συσκευής αυτής είναι της τάξεως των 3.85 ευρώ [23].

Ταχύτητα ρυθμού μετάδοσης δεδομένων με χρήση αυτού του πομποδέκτη:
[250Kbps – 2Mbps], ανάλογα με το Receiver sensitivity.

Χαρακτηριστικά εμβέλειας της συσκευής αυτής:
Transmit power = 20 dBm
Receiver sensitivity = [-86 dBm , -104 dBm]
Antenna gain = 2 dBi
Range = [200m – 2km]

Κατανάλωσή της (η τιμή "battery life" έχει υπολογιστεί με χρήση μίας μπαταρίας τυπικού μεγέθους 1000mAh. Η χωρητικότητα της μπαταρίας χρησιμοποιήθηκε κυρίως για να υπάρχει μία αρχική εικόνα της αυτονομίας, και για να υπάρχει μία πιο εύκολα κατανοητή σύγκριση μεταξύ των συσκευών πομποδεκτών):
Transmit mode current = 115mA => Battery life = 7h
Receive mode current = 45mA => 22h
Power-down mode current = 4.2μA

Παίρνοντας μέσο όρο των ρευμάτων, έχουμε 53mA, που με μπαταρία των **1000mAh**, μας δίνει περίπου 18h διάρκεια ζωής για τη λειτουργία του πομποδέκτη.

Αναλόγως με το ποσοστό λειτουργίας σε emission/receive mode και power-down, η κατανάλωση του πομποδέκτη θα κυμαίνεται. Γενικά, το power-down mode ή το idle mode είναι όταν δε λειτουργεί η συσκευή ως πομπός ή δέκτης. Το ρεύμα που καταναλώνεται σε αυτήν την κατάσταση είναι αρκετές τάξεις μεγέθους μικρότερο σε σχέση με τις καταστάσεις λειτουργίας. Άρα, χρήσιμο είναι, γενικά, να λειτουργεί όσον το δυνατό λιγότερο γίνεται. Αυτό επιτυγχάνεται με το να γίνονται οι απαραίτητες επεξεργασίες των δεδομένων στο κάθε σταθμό, ώστε να στέλνονται λιγότερα δεδομένα αλλά πιο πυκνά ως προς την πληροφορία.

Σχόλιο:
Για τα nRF24L01 modules, τουλάχιστον για τις συσκευές βασισμένες σε μικροελεγτή του τύπου Arduino, υπάρχουν έτοιμες βιβλιοθήκες που εκμεταλλεύονται τη λειτουργία τους [24].

- NRF24L01 2.4GHz Wireless Transceiver Module

Η τιμής της συσκευής αυτής είναι της τάξεως των 1.89 ευρώ [25].

Η ταχύτητα μετάδοσης δεδομένων είναι έως και 2Mbps. Η εμβέλειά της είναι της τάξεως των 10m.

Κατανάλωση:
Transmit mode current = 11.3mA (at 0dBm output power, μπορεί να ρυθμιστεί ανάλογα με την επιθυμητή εμβέλεια.) => Battery life = 88.5h
Receive mode current = 12.3mA (at 2000bps data rate) => Battery life = 81.3h
Power down mode current = 0.9mA

Ο σχετικός πομποδέκτης αποτελεί ίδια συσκευή, όσον αφορά το βασικό σχεδιασμό, με την προηγούμενη, όμως έχει εκτυπωμένη μικρή κεραία πάνω στο chip. Αυτό, από τη μία προσφέρει εξοικονόμηση χώρου, όμως μειώνει δραστικά την εμβέλεια. Επίσης, είναι πιο φθηνή λύση σε σχέση με την προηγούμενη.

- UM801 passthrough (UART) SX1276 wireless module LoRa

Η τιμή της συσκευής αυτές είναι της τάξεως των 5-25 ευρώ [26].

Ταχύτητα μετάδοσης δεδομένων:
[300bps – 31.2kbps]

Εμβέλεια της συσκευής:
Receiver sensitivity = -137dBm @ 300bps, -121dBm @12.5kbps
Transmit Power = [5-20dBm] (όσο πιο μικρό, τόσο λιγότερο range αλλά και λιγότερη κατανάλωση)
Range = [1.7km – 6km]

Κατανάλωση:
Typical transmission current = 120mA => Battery life = 8.3h

Receive mode current = 13mA => 77h
Sleep current = 1.13μA

- RFM95 wireless transceiver module/LoRa spread spectrum communication 868/915mhz SX1276

Η τιμή του σχετικού πομποδέκτη είναι της τάξεως των 6.5 ευρώ [27].

Περαιτέρω πληροφορίες σχετικές με τη συσκευή μπορούν να βρεθούν στα φύλλα δεδομένων της [28].

Ταχύτητα:
Μέχρι 300kbps

Εμβέλεια:
Receiver Sensitivity >-148dBm
Transmit power 20 dBm
Typical range 2km

Κατανάλωση:
TX current draw: <120mA => Battery life = 8.3h
RX current draw: ~10.3mA => 97h
Idle mode current draw: 0.2μA

Σχόλιο:
Τέτοιο module έχει χρησιμοποιηθεί σε μία σειρά βίντεο όπου κατασκευάζεται δίκτυο lora και lora gateway [29] [30], το οποίο περιγράφεται σε επόμενα κεφάλαιο.

- Adafruit feather 32u4 RFM95 LoRa radio

Η τιμή της συσκευής αυτής είναι της τάξεως των 30 ευρώ. Περαιτέρω πληροφορίες για αυτήν μπορούν να βρεθούν στο [31].

Αυτή η συσκευή αποτελεί development board το οποίο έχει ενσωματωμένο το προηγούμενο module. Το board περιέχει μικροελεγκτή ATmega32u4.
Δε δίνονται πληροφορίες για τον ενσωματωμένο πομποδέκτη, αφού αυτές καταγράφηκαν προηγουμένως. Όσον αφορά την πλακέτα και το μικροελεγκτή που χρησιμοποιεί, έχουμε τα παρακάτω χαρακτηριστικά
Clock speed: 8MHz
Αρχιτεκτονική 8-bit
20 GPIO (general purpose input-output) pins

- Digi XBee-PRO s2c ZigBee

Η τιμή της συσκευής είναι της τάξεως των 28.5 ευρώ [32]. Περαιτέρω πληροφορίες για αυτήν μπορούν να βρεθούν στο [33].

Ταχύτητα μετάδοσης δεδομένων:
Μέχρι 1Mbps

Εμβέλεια:
Range = [90-3200m]
Transmit Power = 18dBm
Receiver Sensitivity = -101dBm

Κατανάλωση:
Transmit Current = 120mA => battery life = 8.3h
Receive Current = 45mA => 22h
Power-down Current = 1.5μA

-Xbee ZigBee s2c

Η τιμή για τη συσκευή αυτή είναι της τάξεως των 19.4 ευρώ [34].

Ταχύτητα:
Μέχρι 1Mbps

Εμβέλεια:
Range = [60-1200m]
Transmit power = 5-8dBm
Receiver Sensitivity = -100dBm

Κατανάλωση:
Transmit current = 45mA => 22h
Receive current = 31mA => 32h
Power-down current = 1.5μA

Σχόλιο:
Ο πομποδέκτης αυτός είναι ο ίδιος σε βασικό σχεδιασμό με τον προηγούμενο, αλλά μειώνεται η εμβέλεια για να μειωθεί και η κατανάλωση ρεύματος.

-RFM12B

Η τιμή της συσκευής είναι της τάξεως των 4-5 ευρώ [35]. Περαιτέρω πληροφορίες μπορούν να βρεθούν στα φύλλα δεδομένων της συσκευής [36].

Ταχύτητα:
Μέχρι 115.2kbps

Εμβέλεια:
Transmit power = [-17 , 4dBm]
Receiver sensitivity = -105dBm
Range = ~150m

Κατανάλωση:
Transmit Current draw=18-25mA => battery life = [40h ,55h]
Receive current draw = 14mA => 71h
Idle current draw = 0.6mA

Σχόλιο:

Ο πομποδέκτης αυτός έχει ως πλεονέκτημα ότι έχει πολύ χαμηλή κατανάλωση κατά τη λειτουργία. Υπάρχει έτοιμη βιβλιοθήκη για χρήση του με πλακέτα τύπου Arduino [37].

-RFM69HW

Η τιμή της σχετικής συσκευής είναι της τάξεως των 8 ευρώ [38]. Περισσότερες πληροφορίες για αυτήν μπορούν να βρεθούν στο [39].

Ταχύτητα:
Μέχρι 300kbps

Εμβέλεια:
Transmit power = [-18 , 20dBm]
Receiver sensitivity = -120dBm
Range = ~400-500m

Κατανάλωση:
Transmit Current draw=16mA(για -1dBm) – 130mA(για 20dBm) => battery life = [7h ,62h]
Receive current draw = 16mA => 62h
Idle current draw = 1.2mA

Σχόλιο:
Υπάρχει επίσης βιβλιοθήκη σχετική με αυτήν για προγραμματισμό σε σύνδεση με πλακέτα τύπου Arduino [40].

2.1.3 Πληροφορίες για αρχιτεκτονικές και πρωτόκολλα επικοινωνίας των σχετικών πομποδεκτών

Σε αυτό το κεφάλαιο αναφέρονται αρχιτεκτονικές και πρωτόκολλα που βρέθηκαν μετά από αναζήτηση στο Διαδίκτυο και τα οποία χρησιμοποιούν οι πομποδέκτες που παρουσιάστηκαν, με σχετικά σύντομη επεξήγηση στο καθένα, αρκετή ώστε να υπάρχει μία βασική κατανόησή τους.

LoRa Technology πληροφορίες

LoRa (Long Range) είναι modulation technique που χρησιμοποιεί τεχνική spread spectrum.

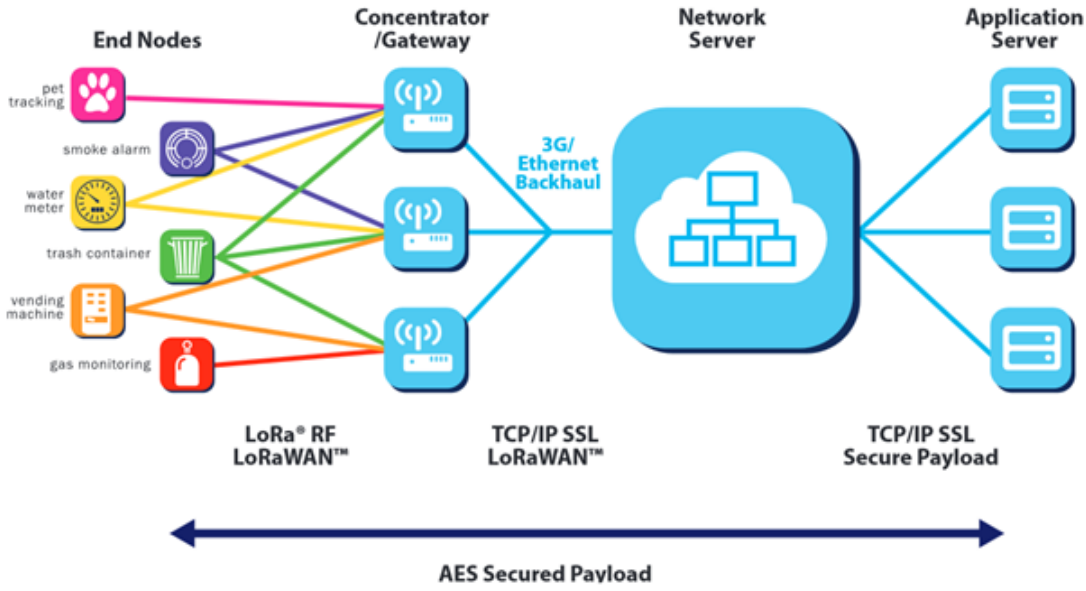
Αυτή είναι διαμόρφωση χρήσιμη για τη μείωση παρεμβολών όταν στέλνονται μηνύματα από τον πομποδέκτη που τη χρησιμοποιεί. Το σήμα που αποστέλνεται μεταδίδεται σε εύρος ζώνης πολύ μεγαλύτερό από τη συχνότητα του περιεχομένου της αρχικής πληροφορίας. Περισσότερες πληροφορίες στα [41][42]. Το LoRaWAN architecture συνήθως χρησιμοποιεί star of stars topology. LoRa είναι το Physical Layer, δηλαδή το modulation που γίνεται στα σήματα, ενώ το LoRaWAN είναι το πρωτόκολλο επικοινωνίας μεταξύ των συσκευών του δίκτυου.

Στο LoRaWAN έχουμε τα εξής μέρη του δικτύου:

- End-devices: Κατά κύριο λόγο είναι κάποιος αισθητήρας με επεξεργαστική ισχύ.

- Gateways: Αποτελούν γέφυρες επικοινωνίας μεταξύ των end-nodes και κάποιου Network Server.

Στη συνέχεια, ο Network Server είναι υπεύθυνος να μεταφέρει τη πληροφορία εκεί που έχει θέσει το end-device.



Με σκοπό τη βελτίωση της αυτονομίας, όσον αφορά την κατανάλωση ρεύματος, και για να αυξηθεί η χωρητικότητα του δικτύου, ο server χειρίζεται τα data rates των end-devices με λογική adaptive data rate. Σύμφωνα με αυτή τη λογική, ανάλογα με κάποιες πληροφορίες (το μέγεθος της πληροφορίας που πρόκειται να σταλεί, κατανάλωση, επίδοση) θέτει την τιμή του data rate για κάθε end-device. Στη διαδικασία αυτή υπάρχει encryption.

Τα end-devices μπορούν να κατηγοριοποιηθούν σε ορισμένες κλάσεις ανάλογα με την ανάγκη τους να έχουν συνεχή ή όχι επικοινωνία με άλλες συσκευές ή με το gateway και με το ρυθμό δεδομένων που επιθυμούν να έχουν. Έτσι, προκύπτουν οι εξής κλάσεις:

- Class A – Bidirectional (battery powered): Για τις συσκευές που βρίσκονται σε αυτήν την κατηγορία, γίνεται 1 uplink transmission με ακόλουθο 2 downlink receptions. Uplink είναι η επικοινωνία που γίνεται από κάτω προς τα πάνω, δηλαδή από τα end-devices προς τα gateways. Downlink, αντίθετα, είναι η επικοινωνία που γίνεται από πάνω προς τα κάτω. Όλα τα end-devices στέλνουν όποτε θέλουν με λογική ALOHA [43]. Σύμφωνα με τη λογική αυτή, αν το end-device έχει δεδομένα να στείλει, τότε τα στέλνει. Αν, ενώ στέλνει δεδομένα, λαμβάνει και άλλα δεδομένα από κάποια άλλη πηγή σήματος, τότε σημαίνει ότι δύο συσκευές έστειλαν ταυτόχρονα και έγινε collision σημάτων άρα και δεδομένων. Προκειμένου τελικά να γίνουν επιτυχώς όλες οι μεταφορές δεδομένων, όλοι οι σταθμοί μετάδοσης ξαναπροσπαθούν να τα στείλουν “αργότερα”, το οποίο ορίζεται από τον αλγόριθμο. Ο αλγόριθμος προσπαθεί να εξασφαλίσει ότι θα σταλθούν σε διαφορετικές χρονικές περιόδους, ώστε τελικά να μην γίνει σύγκρουση σημάτων. Αν γίνει ξανά τέτοια σύγκρουση είτε με τις ίδιες συσκευές είτε με άλλες, τότε επαναλαμβάνεται η διαδικασία μέχρι να γίνουν όλες οι μεταφορές επιτυχώς.
- Class B (low latency): Οι συσκευές αυτές είναι Class A με επιπρόσθετο ότι ανοίγει receive windows σε προγραμματισμένες χρονικές στιγμές, σύμφωνα με ένα συγχρονισμένο beacon, το οποίο στέλνει το gateway. Receive windows είναι χρονικά παράθυρα, στα οποία η συσκευή είναι ενεργοποιημένη και λειτουργεί

συνεχώς ως δέκτης. Beacon (μεταφράζεται ως φάρος) είναι ένα σήμα το οποίο απλά περιέχει την πληροφορία ότι το end-device που την δέχθηκε πρέπει να ανάψει ως δέκτης και να ακούει για σήματα. Έτσι ο server γνωρίζει πότε το end-device ακούει.

- Class C (no latency): Έχει συνέχεια ανοιχτό το receiver, εκτός αν κάνει transmit.

Για να ενεργοποιηθεί ένα end-device χρειάζεται τις εξής πληροφορίες:

- Device Address (DevAddr): Είναι 32bit identifier που συμπεριλαμβάνεται σε κάθε data frame. Το γνωρίζουν το node, ο network server και ο application server.
- Network Session Key (NwkSKey): Αποτελεί 128bit AES key (advanced encryption standard, αποτελεί προσδιορισμό για την κρυπτογράφηση ηλεκτρονικών δεδομένων και καθιερώθηκε από το Εθνικό Ινστιτούτο Πρότυπων και Τεχνολογίας – National Institute of Standards and Technology – NIST των ΗΠΑ), μοναδικό για κάθε end-device. Το γνωρίζουν το end-device και ο network server. Σκοπός του κλειδιού αυτού είναι να υπάρχει ασφάλεια πληροφορίας μεταξύ end-device και network server.
- Application Session Key (AppSKey): Είναι 128bit AES key, μοναδικό για κάθε end-device. Το γνωρίζουν το end-device και ο application server. Ο σκοπός του κλειδιού αυτού είναι να κρυπτογραφούνται τα μηνύματα μεταξύ των δύο.

Υπάρχουν δύο μέθοδοι για να ενεργοποιηθεί ένα end-device, και αυτοί είναι οι εξής:

i. Over the air activation (OTAA) (OTA handshaking):

To end-device εκτελεί τις εξής διαδικασίες:

- Στέλνει Join request στο application server με τις εξής πληροφορίες: globally unique end-device identifier (DevEUI), application identifier (AppEUI) και γίνεται η αυθεντικοποίηση μέσω του AppKey.
- Δέχεται Join accept από το server.
- Κάνει αυθεντικοποίηση του Join accept.
- Αποκρυπτογραφεί το Join accept.
- Εξάγει από τα περιεχόμενα του πακέτου Join accept και αποθηκεύει τα DevAddrs και παράγει μέσω αυτών τα NekSKey, AppSKey.

ii. Activation by personalization (ABP):

Κατά την κατασκευή της συσκευής, τίθενται οι τιμές των DevAddr, NwkSKey, AppSKey. Η συσκευή είναι έτοιμη να επικοινωνήσει χωρίς περαιτέρω διαδικασίες.

Μπορεί να γίνει και επικοινωνία μεταξύ των συσκευών τύπου LoRa χωρίς το LoRaWAN protocol, αλλά με χρήση πιο απλών πρωτόκολλων. Έτσι, μπορεί να παραληφθεί το gateway και να γίνει η επικοινωνία ανάμεσα στα end-devices. Αυτό, από τη μία απλουστεύει το δίκτυο, επειδή δεν υπάρχει η διαδικασία σύνδεσης με το gateway, αλλά από την άλλη είναι πιο πολύπλοκο από την άποψη ότι πρέπει να γίνει η οργάνωση μεταξύ των end-nodes.

Spread Spectrum technique [44]

Είναι η μέθοδος την οποία χρησιμοποιεί το LoRa για μεταφορά σήματος πληροφορίας. Σκοπός αυτής της τεχνικής είναι η μείωση των παρεμβολών και να εξασφαλιστεί η ιδιωτικότητα.

Σύμφωνα με τη μέθοδο αυτή, έχουμε το σήμα πληροφορίας, το οποίο έχει σχετικά στενό bandwidth, το οποίο κάνουμε modulate με carriers, τα οποία είναι σε μεγαλύτερες συχνότητες, με αποτέλεσμα να έχουμε τελικό σήμα προς αποστολή με μεγαλύτερο bandwidth. Carrier είναι η κυματομορφή την οποία κάνουμε modulate βάζοντας πάνω της το σήμα πληροφορίας, με σκοπό την αποστολή του.

Συχνά, χρησιμοποιείται το **Frequency Hopping Spread Spectrum (FHSS)** , κατά το οποίο αλλάζουμε συχνά carrier, επιλέγοντας από πολλά κανάλια συχνότητας. Τα carriers δημιουργούνται βασισμένα σε κάποιο ψευδοτυχαίο αλγόριθμο, τον οποίο γνωρίζουν πομπός και δέκτης. Πλεονέκτημα που έχει η τεχνική αυτή είναι ότι, αν υπάρξει παρεμβολή σε κάποια συγκεκριμένη συχνότητα, τότε θα επηρεάσει το σήμα μόνο στο μικρό διάστημα που έχει το carrier τη συχνότητα αυτή.

Άλλη τεχνική, λιγότερο χρησιμοποιούμενη, είναι η **Direct Sequence Spread Spectrum (DSSS)**, όπου το σήμα πληροφορίας εφαρμόζεται σε ψευδοτυχαίο θόρυβο, ο οποίος αποτελείται από πολύ μικρούς παλμούς, και έτσι έχει μεγάλο bandwidth (έχει φάσμα παρόμοιο με τυχαία ακολουθία, αλλά είναι παραγόμενος ντετερμινιστικά). Έτσι, το τελικό σήμα αποκτά αντοχή σε παρεμβολές.

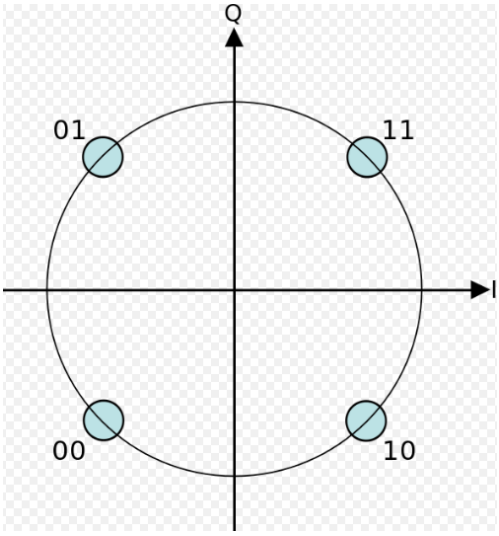
Η τεχνική που χρησιμοποιεί το LoRa είναι το **Chirp Spread Spectrum (CSS)** [45], όπου το carrier (εδώ ονομάζεται chirp) είναι ένα ημιτονοειδές σήμα του οποίου η συχνότητα ανεβαίνει (upchirp) και κατεβαίνει (downchirp) ανάλογα με το χρόνο, σύμφωνα με κάποια συνάρτηση (ανάλογα με τη συνάρτηση με την οποία αλλάζει η συχνότητα προκύπτουν αντίστοιχα τα ονόματα linear chirp, exponential chirp κλπ.).

Αρχιτεκτονική για τα XBEE modules

Το modulation που χρησιμοποιούν τα XBEEs είναι ένα από τα δύο:

-QPSK (Quadrature phase-shift keying) [46]: Το PSK είναι μέθοδος μετάδοσης πληροφορίας κατά την οποία έχουμε το carrier σήμα, στο οποίο ενσωματώνουμε το κάθε κομμάτι δεδομένων, αλλάζοντας, ανάλογα με αυτό, τη φάση του carrier.

Για το QPSK, πιο συγκεκριμένα, έχουμε 4 φάσεις που αντιπροσωπεύουν πληροφορία. Έτσι, αφού η πληροφορία είναι σε δυαδική μορφή, μπορούμε να στείλουμε 2 bits ανά σύμβολο. Για παράδειγμα, όπως φαίνεται στο επόμενο σχήμα, αν θέλουμε να στείλουμε “0111” τότε πρώτα θα σταλθεί ημίτονο με διαφορά φάσης 315 μοίρες και μετά ημίτονο με διαφορά φάσης 45 μοίρες.



-FSK (Frequency-shift keying) [47]: Αποτελεί frequency modulation (FM) όπου έχουμε το carrier σήμα, το οποίο αλλάζει συχνότητα ανάλογα με τη πληροφορία του σήματος, με τη λογική που είδαμε προηγουμένως, δηλαδή τίθενται διαφορετικές συχνότητες για τα 0 και 1 ή και για ζευγάρια αυτών και έτσι προκύπτει το τελικό σήμα για αποστολή. Το πιο απλό FSK είναι το δυαδικό (Binary FSK – BFSK), κατά το οποίο έχουμε μόνο ένα ζευγάρι από συχνότητες, όπου η μία αντιπροσωπεύει το δυαδικό bit 0 και η άλλη το 1.

Όσον αφορά τη σύνδεση του XBEE module με συσκευή βασισμένη σε μικροελεγτή, αν αυτή έχει UART interface, μπορεί να συνδεθεί μεσω αυτού. Κάθε data byte θα έχει την εξής μορφή:

- Idle κατάσταση θα είναι στο high.
- Όταν θέλουμε να στείλουμε κάτι ξεκινάμε με ένα απλό start bit, το οποίο είναι low.
- Στη συνέχεια ακολουθούν 8bits data, με το LSB πρώτο.
- Τελικά στέλνεται ένα απλό stop bit, το οποίο είναι high, και ξαναβρισκόμαστε στην idle κατάσταση.

Από προεπιλογή, τα XBEEs λειτουργούν σε transparent mode, δηλαδή είναι σαν να κάνουμε αντικατάσταση του serial line που θα είχαν δύο πλακέτες για να επικοινωνήσουν μεταξύ τους. Θεωρητικά, όταν είναι σε αυτό το mode, τα XBEEs είναι αόρατα στο δίκτυο, και είναι σαν να υπάρχουν απλά συνδεδεμένα τα boards μεταξύ τους.

Εκτός από το transparent mode, μπορούνε να λειτουργήσουν και σε API operation (application programming interface). Αυτή η λειτουργία έχει την εξής μορφή:

Όλα τα data περιέχονται στα εξής frames:

Transmit data frames: Σε αυτήν την κατηγορία ανήκουν τα RF transmission data frames και τα Command frames (σαν AT (attention) commands).

Receive data frames: Σε αυτήν την κατηγορία ανήκουν τα RF received data frames, τα command responses και τα event notifications(reset, associate κλπ.).

Έτσι, δε χρειάζεται να μπαίνει η συσκευή XBEE σε διαφορετικά modes αν πρόκειται να σταλθεί πιο πολύπλοκη πληροφορία, όπως κάποια εντολή. Αυτός ο τρόπος λειτουργίας του XBEE είναι χρήσιμος για να στέλνονται εντολές χωρίς να χρειάζεται η συσκευή να μπει σε command mode, όπως επίσης και για να λαμβάνει success/failure για κάθε πακέτο που έστειλε.

Οι τοπολογίες δικτύου που υποστηρίζει το XBEE είναι οι εξής:

- Point-to-point: Σύμφωνα με αυτήν τη δομή, η κάθε σύνδεση γίνεται μεταξύ δύο κόμβων και το μήνυμα που στέλνεται δεν μπορεί να ακουστεί από άλλους δέκτες. Δεν υπάρχουν master και slave κόμβοι. Ο συγχρονισμός των κόμβων γίνεται μεταξύ τους, χωρίς την ύπαρξη κάποιου κόμβου, ο οποίος θα είναι υπεύθυνος για αυτή τη διαδικασία.
- NonBeacon (with coordinator): Σύμφωνα με αυτήν τη δομή, μία συσκευή είναι coordinator, η οποία θα συγχρονίζει τα υπόλοιπα nodes μέσω Polling. Polling system είναι ένα σύστημα όπου ο coordinator επισκέπτεται ένα σύνολο από ουρές με κάποια συγκεκριμένη σειρά και επιτρέπει τους κόμβους να επικοινωνήσουν σύμφωνα με τις επισκέψεις αυτές. Σε κάθε ουρά μπαίνουν και περιμένουν οι υπόλοιποι κόμβοι ώστε να έρθει η σειρά τους να επικοινωνήσουν.
- Point-to-multipoint: Σύμφωνα με αυτή τη δομή, υπάρχει μονοπάτι από ένα σημείο σε πολλά. Όσον αφορά τις υπόλοιπες διαδικασίες, όπως επικοινωνία και συγχρονισμός, αυτές ακολουθούν την ίδια λογική με τη δομή point-to-point.
- Mesh: Είναι τοπολογία όπου κάθε κόμβος αναμεταδίδει δεδομένα για το δίκτυο, αν δε γίνεται να επιτευχθεί άμεσα η επικοινωνία μεταξύ των επιθυμητών κόμβων.

Περισσότερες πληροφορίες μπορούν να βρεθούν στο [48].

Αρχιτεκτονική για το NRF24L01

Η σχετική συσκευή πομποδέκτη χρησιμοποιεί τεχνική GFSK (Gaussian frequency-shift keying) [47]. Σύμφωνα με αυτήν, αντί να αλλάζουν ακαριαία οι συχνότητες όταν αλλάζει η πληροφορία, όπως γίνεται στο απλό BFSK, το οποίο περιγράφηκε στα προηγούμενα, οι παλμοί φιλτράρονται με Γκαουσιανό φίλτρο, και έτσι αυτοί και το τελικό σήμα γίνονται πιο ομαλά.

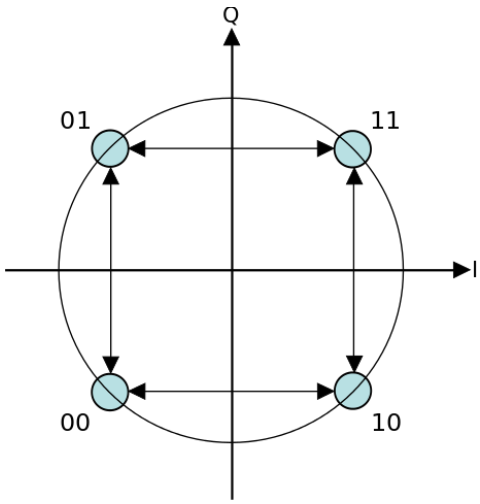
RFM12B

Η συσκευή αυτή χρησιμοποιεί FSK modulation, που περιγράφηκε προηγουμένως.

RFM69HW

Η συσκευή πομποδέκτη αυτή χρησιμοποιεί ένα από τα ακόλουθα:

- FSK, GFSK (περιγράφηκαν προηγουμένως)
- MSK, GMSK (minimum-shift keying) [49]: Πρέπει πρώτα να αναφερθεί το OQPSK (offset Quadrature phase-shift keying). Το OQPSK αποτελεί επέκταση του QPSK, μόνο που επιτρέπονται αλλαγές στις φάσεις μόνο κατά 90 μοίρες, με αποτέλεσμα να έχουμε πιο ομαλές αλλαγές στο σήμα. Στο σχήμα φαίνονται τα βέλη μέσω των οποίων μπορούμε να αλλάξουμε καταστάσεις.



Το MSK έχει την ίδια λογική, απλά αντί για παλμούς που χρησιμοποιεί το OQPSK, εδώ κωδικοποιείται κάθε σύμβολο με μισό ημίτονο. Το GMSK είναι σαν το MSK, όμως το σήμα πρώτα φιλτράρεται με Gaussian φίλτρο, οπότε είναι πιο ομαλό.

- OOK [50]: Αποτελεί την απλούστερη μορφή ASK (amplitude-shift keying), κατά την οποία το carrier σήμα αλλάζει πλάτος ανάλογα με την πληροφορία. Κατά το OOK οι συντελεστές για το πλάτος είναι 0 και 1, δηλαδή αν υπάρχει σήμα για συγκεκριμένο χρονικό διάστημα, τότε έχουμε 1, ενώ αν δεν υπάρχει, τότε έχουμε 0.

2.1.4 Μέθοδος υπολογισμού της εμβέλειας των πομποδεκτών με βάση τα φύλλα δεδομένων

Μία βασική παράμετρος για την αξιολόγηση ενός module για ασύρματη επικοινωνία είναι η εμβέλεια που μπορεί να υπάρχει μεταξύ πομπού και δέκτη ώστε να μπορεί να λαμβάνεται η πληροφορία που στέλνεται. Παρουσιάζεται στη συνέχεια μία μέθοδος για τον υπολογισμό της εμβέλειας συσκευής πομποδέκτη, χρησιμοποιώντας στοιχεία από τα σχετικά φύλλα δεδομένων. Περισσότερες πληροφορίες μπορούν να βρεθούν στο [51].

Στα φύλλα δεδομένων, συνήθως, είτε δεν αναγράφεται η πληροφορία της εμβέλειας, είτε υπάρχει εμβέλεια που έχει υπολογιστεί με πειραματικούς τρόπους, οι οποίοι δεν καταγράφονται, οπότε δεν είναι πάντα αντιπροσωπευτική για την εφαρμογή μας.

Ο βασικός τύπος για υπολογισμό της σχέσης μεταξύ της ισχύος που δέχεται ο δέκτης (Receiver Power, P_r) και της ισχύος που στέλνει ο πομπός (Transmitter Power, P_t) είναι [52]:

$$\frac{P_r}{P_t} = G_t G_r \left(\frac{\lambda}{4\pi R}\right)^2$$

όπου Gt, Gr είναι τα κέρδη κεραιών (Antenna Gains), λ το μήκος κύματος του σήματος και R η εμβέλεια.

Ένας πιο πρακτικός τύπος που περιέχει τιμές που βρίσκονται στα data sheets, είναι το Maximum Path Loss, που είναι υπολογισμένο σε λογαριθμική κλίμακα:

MPL = TXpower – RXsensitivity + gains – losses

Στα data sheets αναφέρεται το Transmitter/TX Power είτε σε μονάδες mW είτε σε dBm, μονάδα που είναι decibel με αναφορά το 1mw [53], δηλαδή χρησιμοποιείται ο τύπος:

$$P(dBm) = 10\log_{10}(\frac{P(mW)}{1\,mW})$$

(Γενικά τα dB υπολογίζονται σε αναφορά ως προς κάποια τιμή [54].) Γενικά, η σχέση μεταξύ απόστασης και ισχύος είναι αντίστροφη τετραγωνική, οπότε, διπλασιάζοντας την απόσταση, χρειάζεται τετραπλάσια ισχύς. Πηγαίνοντας στη λογαριθμική κλίμακα, έχουμε ότι, για κάθε 3dBm διπλασιάζεται η ισχύς, άρα θα χρειαστούν 6dBm παραπάνω για τετραπλασιασμό της ισχύος.

Άλλη μία παράμετρος των πομποδεκτών είναι το Receiver Sensitivity [55]. Είναι η ελάχιστη ισχύς σήματος που πρέπει να φτάσει στο δέκτη ώστε να μπορέσει να αντιληφθεί το σήμα και να λάβει την πληροφορία.

Ο όρος gains, του τύπου MPL, εξαρτάται από τον προσανατολισμό της μετάδοσης και από τις κεραίες για πομπό και δέκτη [56]. Τα κέρδη κεραίας είναι σε dBi, δηλαδή με αναφορά μία ισοτροπική κεραία [57], δηλαδή θεωρητική σημειακή πηγή που εκπέμπει σήμα ίδιας έντασης προς όλες τις κατευθύνσεις (σε σφαιρικό σχήμα). Τα κέρδη της κεραίας εξαρτώνται από την κατευθυνσιμότητα και από την ηλεκτρική αποδοτικότητα αυτής.

Πιο συγκεκριμένα, για τον πομπό, είναι το πόσο καλά η κεραία μετατρέπει την ισχύ εισόδου σε κατευθυνόμενα κύματα σήματος, ενώ για το δέκτη είναι το πόσο καλά η κεραία μετατρέπει τα ραδιοκύματα σε ηλεκτρική ισχύ. Το κέρδος κεραίας είναι ανάλογο με το πόσο προσανατολισμένη είναι. Για παράδειγμα, στην τηλεόραση χρησιμοποιείται κεραία δέκτη προσανατολισμένη προς τους πομπούς, που προσφέρει μεγάλο κέρδος, περίπου 14-15dBi. Στην εφαρμογή μας, επειδή δε γνωρίζουμε πάντα την τοποθεσία των κόμβων, δεν μπορούμε να χρησιμοποιήσουμε προσανατολισμένες κεραίες, οπότε περιορίζουμε το κέρδος τους περίπου στα 2-6dBi.

Ο όρος losses, του τύπου MPL, εξαρτάται από απώλειες του πομπού και του δέκτη (λόγω συνδέσεων).

Επειδή δε γίνεται να υπολογιστεί με ακρίβεια πόσο τα εμπόδια που υπάρχουν μειώνουν το max path loss, μπαίνει τεχνητά ένας ακόμα όρος, που λέγεται fade margin. Τυπικές τιμές είναι 12-30dBm. Άρα, ο τύπος γίνεται:

Max path Loss = Txpower – Rxsensitivy + gains – losses – fade margin

Τελικά, ο τύπος υπολογισμού τις απόστασης είναι:

$$Distance(km) = 10^{(max\,path\,loss - 32.44 - 20\log(f))/20}$$

Το Link Budget [58] είναι ένας άλλος τύπος για υπολογισμό της ισχύος που λαμβάνει ο δέκτης, που, όπως το λέει και η ονομασία του, είναι budget που μπορούμε να "ξοδέψουμε" μέρος του για να πάρουμε απόσταση ή για να βάλουμε κάποιο εμπόδιο ανάμεσα (υγρασία , φυσικά εμπόδια, λαμαρίνες, σανίδες κλπ.). Πρέπει τελικά να παραμένει η τιμή του πάνω από το 0 για να φτάσει το σήμα στο δέκτη. Ο τύπος του είναι:

Pr = TXpower – RXsensitivity + gains – losses – Lfs – fade marin

Το Lfs είναι η παράμετρος για να συνυπολογιστεί η απόσταση στο budget, και έχει τύπο:

$$L_{FS}(dB) = 32.44\,dB + 20\log(frequency(MHz)) + 20\log(distance(km))$$

Επομένως, στους καταγεγραμμένους πομποδέκτες της διπλωματικής εργασίας, η εμβέλεια που αναφέρεται είναι είτε υπολογισμένη με χρήση των προηγούμενων τύπων, είτε παίρνεται από τα φύλλα δεδομένων, αν εξηγείται επαρκώς υπό ποιες συνθήκες έχει υπολογιστεί αυτή.

2.2 Δυνατότητα φόρτωσης προγράμματος στην ολοκληρωμένη συσκευή σε πραγματικό χρόνο από απομακρυσμένο σταθμό βάσης

Στην προσπάθεια αύξησης της ευφυΐας των ολοκληρωμένων συσκευών με μικροεπεξεργαστές, ώστε αυτοί να δέχονται προγράμματα από εξωτερικό σταθμό βάσης, τα οποία πρέπει να εκτελούν με διέγερση είτε από συγκεκριμένα εξωτερικά γεγονότα (events) είτε από ένα πραγματικού χρόνου ρολόι, εξετάστηκε η περίπτωση ιδιοκατασκευών, κατάλληλων για τις ειδικές ανάγκες συγκεκριμένων κατηγοριών πρακτικών εφαρμογών. Στο πλαίσιο αυτό εξετάστηκε η δυνατότητα χρήσης dual-port RAMs ελεγχόμενων από ειδικό κύκλωμα, το οποίο θα επέτρεπε στο σταθμό βάσης να ανανεώσει τις υπορουτίνες και τα προγράμματα του μικροεπεξεργαστή ώστε αυτά να εξυπηρετούν τις διαφορετικές ανάγκες που παρουσιάζονται σε διαφορετικές εφαρμογές. Αυτό θα επέτρεπε να παραμείνει το αναγκαίο υλικό των ολοκληρωμένων συσκευών με τους μικροεπεξεργαστές χαμηλού κόστους, ενώ ταυτόχρονα θα αύξανε σημαντικά τις δυνατότητες της τοπικής ολοκληρωμένης συσκευής.

Σημαντικό είναι για κάθε συσκευή του δικτύου να γίνονται οι απαραίτητες ανανεώσεις (updates) του λογισμικού του, έτσι ώστε να υπάρχει η εγγύηση ότι ο κάθε σταθμός θα εκτελεί την τελευταία έκδοση του σχετικού προγράμματος. Αυτό μπορεί να επιτευχθεί με δύο τρόπους:

1) Να επιλεγεί συσκευή που έχει τη δυνατότητα ενημέρωσης του προγράμματός της μέσω τηλεπικοινωνιακών καναλιών (πιθανή σύνδεση GPRS ή GSM, πιθανή δορυφορική ζεύξη κλπ.).

2) Να σχεδιαστεί ιδιοκατασκευή που να επιτρέπει την εγγραφή από εξωτερικό port σε ειδική μνήμη του προγράμματος προς εκτέλεση από το μικροεπεξεργαστή.

Η διαδικασία για να εξασφαλίζεται η ύπαρξη της τελευταίας έκδοσης είναι ότι το πρόγραμμα θα γράφεται από εξωτερική πηγή και μετά θα εκτελείται από το μικροεπεξεργαστή. Με το σκοπό αυτό, εξετάστηκε αρχικά, ως πιθανή λύση, η χρήση Dual-Port RAMs. Σε αυτό το σημείο της εργασίας παρατίθεται σύντομη πληροφορία σχετικά με αυτές. Λόγω του περιορισμένου όγκου εργασίας της διπλωματικής, στην παρούσα εργασία δεν προχωρήσαμε στη σχεδίαση και υλοποίηση ιδιοκατασκευής.

2.2.1 Πληροφορίες για Dual-Port RAM

Dual-Port RAM είναι μνήμη τυχαίας προσπέλασης, στην οποία υπάρχουν 2 ports για ταυτόχρονη πρόσβαση από 2 χρήστες. Στο επίπεδο ολοκληρωμένων κυκλωμάτων, αυτό σημαίνει ότι κάθε κελί μνήμης μπορεί να γίνει accessed ταυτόχρονα από 2 ανεξάρτητα σύνολα από διευθύνσεις, δεδομένα, και γραμμές ελέγχου.

Ο λόγος που αναζητήθηκαν πληροφορίες για αυτές τις μνήμες είναι ότι θα μπορούσαν να συνδεθούν σε κάθε συσκευή του δικτύου, και να μπορούν να εκτελούν κώδικα που τους έρχεται μέσω ασύρματης επικοινωνίας. Δηλαδή, να διαβάζουν και να εκτελούν κώδικα από τη μνήμη, ενώ ταυτόχρονα μπορεί να γράφεται στη μνήμη και άλλος κώδικας ο οποίος ήρθε από το ασύρματο. Αναφέρονται παρακάτω μερικές βασικές πληροφορίες για την

πληρέστερη κατανόηση της λειτουργίας τους. Περισσότερες πληροφορίες υπάρχουν στο [59].

Με τη χρήση τους, η αποδοτικότητα των προσβάσεων στη μνήμη πρακτικά διπλασιάζεται. Αφού δεν υπάρχουν περιορισμοί του πότε γίνονται οι προσπελάσεις, 2 επεξεργαστές που πχ. έχουν συνδεθεί στο dual-port RAM μπορούν να έχουν μη συγχρονισμένα ρολόγια.

Για να εξασφαλιστεί η εγκυρότητα των δεδομένων στη μνήμη, όταν λειτουργεί με 2 συνδεδεμένους επεξεργαστές, χρειάζεται όταν ο ένας χρησιμοποιεί κάποιο μέρος της μνήμης, τότε αν ο άλλος θέλει να χρησιμοποιήσει αυτό το μέρος, να περιμένει να τελειώσει ο πρώτος.

Αυτό γίνεται με κάποια λογική κλειδώματος ή ατομικών εντολών. Ο πρώτος επεξεργαστής θέτει μία μεταβλητή LOCK σε τιμή 1, και χρησιμοποιεί το μέρος της μνήμης. Ο δεύτερος, όταν πάει να κλειδώσει το ίδιο μέρος μνήμης, θα δει ότι είναι ήδη LOCK=1, άρα καταλαβαίνει ότι υπάρχει κάποιος άλλος σε αυτό το μέρος, οπότε και περιμένει να ξεκλειδώσει.

Μία οριακή περίπτωση είναι αν και οι 2 διευθύνσεις που θέλουν οι επεξεργαστές να προσπελάσουν είναι ίδιες, και η προσπέλαση τυχαίνει να γίνει πρακτικά την ίδια χρονική στιγμή. Τότε υπάρχει η πιθανότητα να “χάσουν” και οι 2 στη διεκδίκηση του κλειδώματος, και να περιμένουν ο ένας τον άλλον, και να μη γίνεται τίποτα. Και τα 2 ports είναι σε κατάσταση busy, και το σύστημα δεν προχωράει. Η λύση για τις dual-port RAMs είναι να μην τίθενται ποτέ 2 masters, αλλά 1 master και 1 slave.

Το interface που χρησιμοποιείται από τις dual-port RAMs έχει 2 σύνολα των εξής σημάτων (ένα σύνολο για το αριστερό port και ένα για το δεξί): address, data, control.

Αν και τα 2 ports αναφερθούν στην ίδια διεύθυνση την ίδια χρονική στιγμή, ο master κάνει διαιτησία και αποφασίζει ποιος κερδίζει. Στην περίπτωση που θέλουν να διαβάσουν ταυτόχρονα το περιεχόμενο της διεύθυνσης, τότε δεν υπάρχει πρόβλημα και διαβάζουν κανονικά. Στην περίπτωση που το ένα θέλει να γράψει και το άλλο να διαβάσει, ο master αποφασίζει ποιος νικάει, και προκύπτουν οι εξής περιπτώσεις:

Αν χάσει η εγγραφή, τότε αυτή αναστέλλεται, και μέσω μιας BUSY flag ο χαμένος ενημερώνεται ότι δεν υπήρχε επιτυχία.

Αν χάσει η ανάγνωση, τότε μπορεί να διαβάσει κάποιο συνδυασμό νέων και παλιών δεδομένων. Το BUSY flag έχει μία τιμή για το χρονικό διάστημα που υπάρχουν όλα τα παλιά δεδομένα, οπότε, με χρήση αυτού του flag, γνωρίζει μέχρι πότε θα υπάρχουν τα παλιά, αν θέλει να διαβάσει αυτά. Αν θέλουν να διαβαστούν τα νέα δεδομένα, τότε μπορεί να χρησιμοποιηθεί πάλι το BUSY flag για να διαβαστούν αφού αλλάξει αυτό τιμή, που σημαίνει ότι πλέον υπάρχουν νέα δεδομένα.

Υπάρχουν σύγχρονες και ασύγχρονες dual-port RAMs. Οι σύγχρονες υποχρεώνουν να γίνονται ταυτόχρονα οι προσπελάσεις, ενώ οι ασύγχρονες επιτρέπουν η κάθε μία από τις δύο θύρες να χρησιμοποιεί κελιά της μνήμης όποτε επιθυμεί, με τους περιορισμούς που αναφέρθηκαν προηγουμένως. Παρατίθεται ένας σύντομος πίνακας για τα πλεονεκτήματα και μειονεκτήματα των σύγχρονων και ασύγχρονων dual-port RAMs.

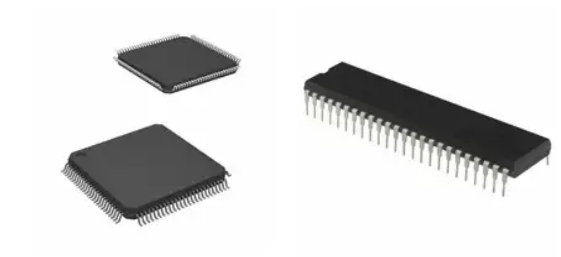
	Σύγχρονη	**Άσυγχρονη**
Πλεονεκτήματα	Πιο απλό interface. Παλιό/Πολυχρησιμοποιημένο design.	Μεγαλύτερες δυνατότητες bandwidth.

Μειονεκτήματα	Μικρότερες δυνατότητες bandwidth.	Πιο πολύπλοκο interface.

2.2.2 Μερικές προτεινόμενες Dual-Port RAMs

Αφού εξηγήθηκε η λειτουργία τους, επακόλουθο είναι να αναφερθούν μερικές επιλεγμένες μνήμες. Τα βασικά χαρακτηριστικά τέτοιων μνημών είναι τα εξής:

- Μέγεθος μνήμης (Memory density)
- Τρόπος λειτουργίας (Operating Mode): Asynchronous, Synchronous, ή και τα δύο.
- Memory organization: Ο τρόπος που η μνήμη χωρίζεται σε μικρότερα υποτμήματα. Για παράδειγμα, μία μνήμη 1MB μπορεί να είναι 64KBx16, 32KBx32 κλπ.
- Access Time-Max: Ο χρόνος που ένα πρόγραμμα ή συσκευή χρειάζεται για να εντοπίσει μία μονάδα πληροφορίας και να την κάνει διαθέσιμη για επεξεργασία.
- Operating Voltage
- Mounting Type: Η δομή των pins με τα οποία γίνεται η σύνδεση της dual-port RAM στο κύκλωμα. Πιο σημαντικές είναι οι Surface mount και Through hole, όπου φαίνονται παρακάτω.



Οι εταιρείες από τις οποίες βρέθηκαν οι παρακάτω Dual-Port RAMs είναι οι:

- IDT / Integrated Device Technology [60], με κύριο διανομέα την DigiKey [61]

- Cypress [62]

Οι παρακάτω μνήμες επιλέχθηκαν περισσότερο συμβολικά, διότι δεν υπάρχει κάποιο χαρακτηριστικό που να καθιστά κάποια μνήμη ως καλύτερη επιλογή. Γενικά, για τους σκοπούς της εργασίας μας δεν έχουμε πολλές απαιτήσεις από επίδοση της μνήμης αυτής, οπότε θα μπορούσε να θεωρηθεί βασικό κριτήριο η τιμή.

- IDT7028L high speed 64Kx16 dual-port static ram 68.5 ευρώ

Περαιτέρω πληροφορίες σχετικά με αυτή τη συσκευή μπορούν να βρεθούν στο [63].

Memory Size: 1MB (64K x 16)
Technology: Asynchronous
Access Time: 15ns

Operating Voltage: 4.5V
Mounting Type: Surface Mount

- IDT7130LA35PDG high speed 1Kx8 dual-port static sram 9 ευρώ

Περαιτέρω πληροφορίες σχετικά με αυτή τη συσκευή μπορούν να βρεθούν στο [64].

Memory Size: 8KB (1K x 8)
Technology: Asynchronous
Access Time: 35ns
Operating Voltage: 4.5V
Mounting Type: Through Hole

- IDT7130LA100PDG 6.9 ευρώ

Περαιτέρω πληροφορίες σχετικά με αυτή τη συσκευή μπορούν να βρεθούν στο [65].

Memory Size: 8KB (1K x 8)
Technology: Asynchronous
Access Time: 100ns
Operating Voltage: 4.5V
Mounting Type: Through Hole

- IDT7132LA100PDG 8.5 ευρώ

Περαιτέρω πληροφορίες σχετικά με αυτή τη συσκευή μπορούν να βρεθούν στο [66].

Memory Size: 16KB (2K x 8)
Technology: Asynchronous
Access Time: 100ns
Operating Voltage: 4.5V
Mounting Type: Through Hole

Παρατηρούμε, λοιπόν, ότι υπάρχουν αρκετές διαφορετικές επιλογές, και καταγράφηκε ένα πολύ μικρό ποσοστό τους, ώστε να μπορούμε να βρούμε την επιθυμητή λύση στην ανάγκη μας. Στην υλοποίηση που γίνεται στη συνέχεια ενός δικτύου αυτοοργάνωσης δε χρησιμοποιήθηκε dual-port RAM, διότι είναι μία σχετικά απλή υλοποίηση όπου δεν υπάρχουν μεγάλες ανάγκες ανταλλαγής κώδικα και δεδομένων.

2.3 Παρουσίαση επιλεγμένων μεθοδολογιών και πρωτοκόλλων για την αυτοοργάνωση των αυτόνομων, ευφυών, ολοκληρωμένων συσκευών σε τοπικά δίκτυα

Έχει διαπιστωθεί από την ομάδα που δουλεύει ερευνητικά και αναπτυξιακά στον τομέα της εξυπηρέτησης αναγκών και της παροχής υπηρεσιών που σχετίζονται με τα δίκτυα μεταφορών (τόσο αγαθών, όσο και επιβατών), ότι μία μελλοντική εξέλιξη που θα εξυπηρετούσε σημαντικά τις προαναφερθείσες ανάγκες και θα ανέβαζε το επίπεδο παροχής υπηρεσιών στις μεταφορές είναι η δυνατότητα αυτοοργάνωσης συσκευών σε τοπικές δικτυακές υποομάδες. Για να γίνει αυτό κατανοητό, δίνονται δύο παραδείγματα:

1) Βαγόνια σε τρένα: Οι ανάγκες σε αυτήν την κατηγορία αλλάζουν συνεχώς λόγω της φύσης των υπηρεσιών και των αγαθών που μεταφέρονται. Πιο συγκεκριμένα, υπάρχουν αγαθά τα οποία χρειάζονται συγκεκριμένη μεταχείριση προκειμένου να εξασφαλιστεί η ασφαλής και απαιτούμενη μετακίνησή τους. Για παράδειγμα, αγαθά που χρειάζονται συγκεκριμένη θερμοκρασία πρέπει να βρίσκονται σε βαγόνι-ψυγείο, το οποίο θα πρέπει να βρίσκεται έτοιμο να παραλάβει τα προϊόντα τη στιγμή που φτάνουν στο σταθμό του τρένου. Χώρος σε τέτοιο βαγόνι μπορεί να υπάρχει σε τρένο το οποίο πλησιάζει τη στάση, οπότε μπορεί απλά όταν φτάσει, να τοποθετηθούν τα προϊόντα και να υπάρχουν οι επιθυμητές συνθήκες. Αν, όμως, δεν υπάρχει αρκετός χώρος ή αν το τρένο πρόκειται να μην φτάσει στην προκαθορισμένη ώρα, τότε θα πρέπει να βρεθεί εναλλακτική λύση, δηλαδή να ενεργοποιηθεί νέο βαγόνι-ψυγείο, να τοποθετηθούν εκεί τα προϊόντα και μόλις φτάσει το τρένο να συνδεθεί το βαγόνι σε αυτό και να συνεχιστεί η πορεία του. Όλη αυτή η διαδικασία, η οποία μπορεί να φαίνεται απλή για εύρεση λύσης, όταν προστεθεί με άλλες πολυάριθμες ανάλογες διαδικασίες, δημιουργούν ένα πολύπλοκο σύνολο προβλημάτων το οποίο έχει πολλές λύσεις. Η εύρεση της βέλτιστης λύσης, η οποία θα ικανοποιήσει κάθε ανάγκη με το ελάχιστο δυνατό κόστος, αποτελεί πολύπλοκο πρόβλημα απόφασης για τον υπεύθυνο οργάνωσης του τρένου. Σχεδιάζοντας μία δομή η οποία θα αυτοοργανώνει τα βαγόνια του τρένου, ώστε να μπορούν να επικοινωνούν μεταξύ τους χωρίς την επίβλεψη υπεύθυνου, και τελικά να προσφέρουν μία απλοποιημένη πληροφορία για τις δυνατές λύσεις βοηθάει τόσο τον υπεύθυνο τελικά να πάρει τη σωστή λύση, αλλά και τους πελάτες που επιθυμούν άμεση και βέλτιστη εξυπηρέτηση.

Επίσης, σε περιπτώσεις μη αναμενόμενων γεγονότων, όπως βλάβης κάποιου βαγονιού ή κλοπής του κλπ., μπορεί να γίνει άμεσα η ενημέρωση της κατάστασης των βαγονιών μέσα από την αυτοοργάνωσή τους, προσθέτοντας λειτουργία συχνού ελέγχου μεταξύ των βαγονιών.

Αυτές και άλλες είναι οι περιπτώσεις όπου η αυτοοργάνωση τοπικών δικτύων βαγονιών θα προσέφερε ευκολία στη χρήση και τον έλεγχό τους.

2) Σειρά φορτηγών μεταφορών, το ένα πίσω από το άλλο: Σε αυτήν την περίπτωση μπορούν να δημιουργηθούν προβλήματα διαφορετικής φύσης, μερικά από τα οποία είναι τα εξής.

Πρώτον, τα φορτηγά μεταξύ τους χρειάζεται να έχουν μία προκαθορισμένη απόσταση, ώστε να γνωρίζουν τις συνθήκες που βρίσκεται το καθένα. Αν αλλάξουν οι αποστάσεις, θα μπορεί εύκολα με την ύπαρξη αυτοοργανούμενου δικτύου να γίνει ενημέρωση της αλλαγής και να γίνει προσαρμογή.

Δεύτερον, μπορεί να υπάρξει επιθυμία ή και ανάγκη αλλαγής δρομολογίου, για διάφορους λόγους, από αλλαγή παραλήπτη μέχρι και ατυχήματα στο δρόμο. Προκειμένου να μπορεί να παρθεί γρήγορα η απόφαση αλλαγής και να ενημερωθούν άμεσα όλα τα φορτηγά, είναι

χρήσιμο να υπάρχει άμεση ενημέρωση, ώστε μόλις παρθεί η απόφαση είτε από κάποιο φορτηγό είτε από τον κεντρικό σταθμό οργάνωσης, να μπορέσουν όλα τα φορτηγά να προσαρμόσουν την πορεία τους.

Τρίτον, όπως και με τα βαγόνια του τρένου, αν υπάρξουν μη αναμενόμενα γεγονότα στα φορτηγά, όπως βλάβη κάποιου ψυγείου ή βλάβη της μηχανής του φορτηγού με αποτέλεσμα να μην είναι δυνατή η παροχή ηλεκτρικής ισχύος στο εμπορευματοκιβώτιό του, είναι σημαντικό να βρεθεί άμεση λύση ώστε να μην υπάρξουν απώλειες ή τουλάχιστο να ελαχιστοποιηθούν.

Παρατηρείται, επομένως, η ανάγκη και η χρησιμότητα που θα είχε η προσπάθεια σχεδιασμού αυτοοργάνωσης τέτοιων τοπικών δικτύων. Για αυτό, είδαμε ορισμένες εργασίες σε αυτοοργάνωση. Επελέγησαν τρεις εργασίες, μία κύρια στον τομέα η οποία είναι για peer-to-peer δίκτυα (επελέγη ο αγγλικός όρος λόγω της ευρείας χρήσης του) και δύο άλλες λόγω του ενδιαφέροντός τους.

2.3.1 1ο σύστημα: Το σύστημα Chord

Στο σημείο αυτό γίνεται μία σχετικά λεπτομερής παρουσίαση του paper με ονομασία:

Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications

Το paper αυτό θεωρήθηκε ως βασική δομή για εφαρμογές που μας ενδιαφέρουν και θα βοηθήσουν στην δημιουργία πιο ευφυών συστημάτων αυτοοργάνωσης για μεταφορές [67].

Έχει γραφεί και μία παρουσίαση στο [68].

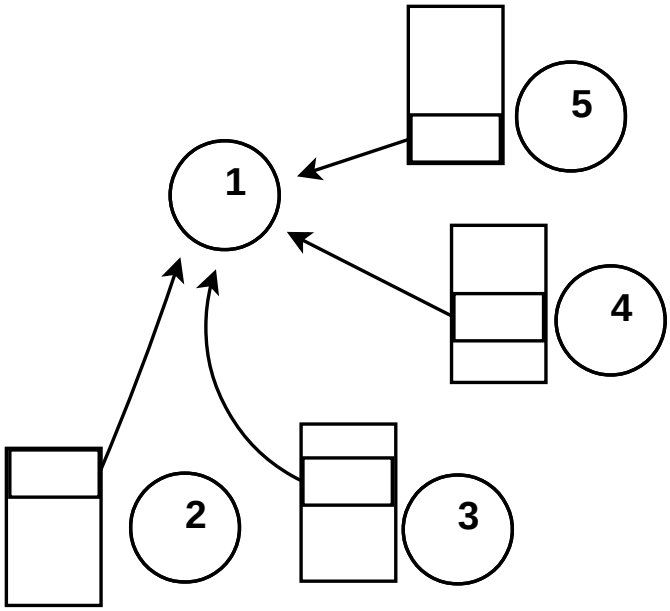
Η βασική λειτουργία που προσφέρει το chord είναι: δεδομένου ενός κλειδιού, το αντιστοιχεί σε έναν κόμβο.

Ο τρόπος που επεκτείνεται αυτό σε δεδομένα είναι να αντιστοιχούνται τα δεδομένα με τα κλειδιά, οπότε όταν προσπαθεί κανείς να βρει κάτι, γνωρίζει το κλειδί για τα δεδομένα που θέλει, και έτσι βρίσκει αυτό που θέλει.

Αυτή η λειτουργία γίνεται σε O(logN) βήματα (πάντα σε προγραμματιστικές εφαρμογές, όταν υπάρχει log χωρίς αναφορά βάσης, είναι λογάριθμος με βάση το 2), όπου N είναι το πλήθος των κόμβων στο δίκτυο. Κατά τη λειτουργία του δικτύου, αν κάποιος κόμβος χρειαστεί κάποια πληροφορία, θα την πάρει από μερικούς άλλους κόμβους του δικτύου, όχι μόνο από έναν, με αποτέλεσμα να γίνει πιο αποδοτική η μεταφορά δεδομένων, αφού περισσότεροι χρήστες θα στέλνουν δεδομένα. Πιο συγκεκριμένα, δείχνεται ότι χρειάζεται να επικοινωνήσει με O(logN) κόμβους προκειμένου να πάρει όλη την πληροφορία. Για να βρει κάθε κόμβο χρειάζεται επίσης O(logN) βήματα, οπότε για τη λήψη του συνολικού πακέτου δεδομένων που χρειάζεται ο κόμβος θα γίνουν $O(\log^2 N)$.

Το peer-to-peer (p2p) σύστημα δικτύου περιέχει ισάξιους κόμβους, δηλαδή είναι decentralized (δεν υπάρχει κάποιος κεντρικός). Είναι distributed αρχιτεκτονική, κατά την οποία οι εργασίες που πρόκειται να εκτελεστούν τμηματοποιούνται ανάμεσα στους κόμβους του δικτύου (peers) με σκοπό την αύξηση της αποδοτικότητας. Στο επόμενο σχήμα φαίνεται ένα απλό παράδειγμα, όπου ο κόμβος 1 θέλει να λάβει έναν πίνακα που γνωρίζει ότι υπάρχει στους κόμβους 2 έως και 5. Οι υπόλοιποι κόμβοι θα οργανωθούν και θα

στείλουν διαφορετικό τμήμα του πίνακα αυτού, με αποτέλεσμα τελικά να στείλει ο καθένας μόνο το ένα τέταρτο του πίνακα και η αποστολή να ολοκληρωθεί συντομότερα.



Το chord είναι κλιμακωτό (scalable) πρωτόκολλο για αναζήτηση σε δυναμικό peer-to-peer σύστημα, οπού συχνά γίνονται εντάξεις και αναχωρήσεις κόμβων στο δίκτυο.

Κάθε κόμβος χρειάζεται να γνωρίζει μόνο για μερικούς ακόμα κόμβους του δικτύου. Προκειμένου να βρει ο,τι χρειάζεται, στέλνει μηνύματα στους κόμβους των οποίων γνωρίζει την ύπαρξη, και αυτοί με τη σειρά τους στέλνουν σε αυτούς, των οποίων γνωρίζουν την ύπαρξη στο δίκτυο, με αποτέλεσμα τελικά να υπάρχει επικοινωνία μεταξύ όλων των κόμβων.

Στη συνέχεια παρουσιάζονται συνοπτικά μερικές σχετικές εργασίες με το Chord:

Όπως είπαμε, το chord αντιστοιχεί το κλειδί σε κόμβο. Για να έχουμε αντιστοίχηση από κλειδί σε κάποια τιμή, χρήσιμο δεδομένο ή γενικότερα πληροφορία, απλά έχουμε στον κόμβο τη σχέση κλειδιού-τιμής.

- Το DNS (domain name system) είναι decentralized σύστημα ονομάτων για υπολογιστές, υπηρεσίες και άλλους πόρους που είναι συνδεδεμένοι στο διαδίκτυο. Η πιο βασική λειτουργία του είναι η αντιστοίχηση ενός ευανάγνωστου και εύκολου στην απομνημόνευση host name στο IP address που χρειάζεται για να εντοπιστεί ο επιθυμητός υπολογιστικός πόρος. Αποτελεί βασική λειτουργία του διαδικτύου.

  Το chord μπορεί να κάνει την ίδια λειτουργία με το να θεωρήσει τα host names ως κλειδιά και να κάνει έτσι την αντιστοίχηση, θεωρώντας τη διεύθυνση του κάθε κόμβου ίση με την IP address που πρέπει να συνδεθεί όταν κάποιος πληκτρολογεί το host name. Ως πλεονεκτήματα, το chord δε χρειάζεται κάποιο server και δεν έχει κάποια προαπαίτηση για τη δομή των ονομάτων, αφού με ευκολία απλά αντιστοιχίζεται το κάθε όνομα με τιμές που προκύπτουν από τη διαδικασία του συστήματος Chord.

- Το Freenet είναι μία peer-to-peer πλατφόρμα για επικοινωνία με αντίσταση στη λογοκρισία [69]. Χρησιμοποιεί decentralized distributed data store, το οποίο είναι δίκτυο υπολογιστών όπου η πληροφορία αποθηκεύεται σε παραπάνω από έναν κόμβο. Έτσι, δεν υπάρχει συγκεκριμένη ευθύνη από κάθε server, με αποτέλεσμα να μπορεί να προσαρμόζεται σε συνδέσεις και αποχωρήσεις κόμβων του δικτύου. Αυτό

προσφέρει ανωνυμία, όμως έχει ως μειονέκτημα ότι δεν εγγυάται η παραλαβή της πληροφορίας, όπως επίσης δεν υπάρχει κάποιος περιορισμός στο χρόνο που θα χρειαστεί να παραληφθεί.

Το chord δεν προσφέρει ανωνυμία, αλλά είναι πιο γρήγορο και πιο αξιόπιστο στη μεταφορά πληροφορίας.

- Το distributed data location protocol από Plaxton προηγήθηκε του Chord. Εγγυάται ότι θα φτάσουμε σε O(logN) βήματα στον επιθυμητό κόμβο χωρίς να τον έχουμε ξεπεράσει προηγουμένως, εξοικονομώντας βήματα. Όμως, αυτή η μέθοδος τον κάνει πιο πολύπλοκο από το Chord.
- Το CAN (scalable content addressable network) έχει ένα d-dimensional cartesian coordinate space για να υλοποιήσει distributed hash table προκειμένου να αντιστοιχεί τα κλειδιά σε τιμές. Έτσι, η αναζήτηση γίνεται σε $O(dN^{1/d})$. Θέλει συντήρηση στις αντιστοιχήσεις μεταξύ του identifier space και κόμβων.

Μοντέλο συστήματος

Το chord προσπαθεί να βελτιώσει τα peer-to-peer δίκτυα ως προς τις εξής παραμέτρους:

- Load balance: Να υπάρχει ισορροπία στο πλήθος των κλειδιών στους κόμβους.
- Decentralization: είναι distributed, όλοι οι κόμβοι είναι ισάξιοι και δεν υπάρχει κάποιος κεντρικός κόμβος, ο οποίος είναι υπεύθυνος για αποφάσεις και οργάνωση του δικτύου.
- Scalability: Το κόστος για αναζήτηση ενός κόμβου από έναν άλλον είναι O(logN), το οποίο είναι πολύ αποδοτικό ακόμα και για μεγάλα σε πλήθος κόμβων δίκτυα.
- Availability: Εξασφαλίζεται ότι κάθε κόμβος θα μπορέσει να βρει αυτό που χρειάζεται. Με αυτόματες διαδικασίες, οι ευθύνες του δικτύου μοιράζονται ανάμεσα στους κόμβους.
- Flexible naming: Δεν υπάρχουν περιορισμοί για τη δομή του κλειδιού. Οι εφαρμογές μπορούν να αντιστοιχούν τα κλειδιά σε άλλου είδους δεδομένα με ελαστικότητα.

Το λογισμικό του chord αποτελεί μία βιβλιοθήκη που συνδέεται με την εφαρμογή. Οι διαδικασίες που προσφέρει είναι οι εξής:

- lookup(key): Επιστρέφει το IP address του κόμβου που είναι υπεύθυνο για το κλειδί key.
- Κάθε κόμβος ενημερώνει την εφαρμογή για κάθε αλλαγή στα κλειδιά για τα οποία είναι υπεύθυνος.

  Το chord χρησιμοποιεί flat key space, δηλαδή ο χώρος των δυνατών λειτουργικών κλειδιών περιέχει ισοδύναμα κλειδιά ως προς την εγκυρότητα και ασφάλειά τους.

Η εφαρμογή που χρησιμοποιεί το chord είναι υπεύθυνη για πιστοποιήσεις, αποθήκευση δεδομένων, αναπαραγωγή τους και χρήση ονομάτων φιλικών προς το χρήστη.

2.3.1.1 Το πρωτόκολλο

Σύμφωνα με το πρωτόκολλο του Chord, τίθενται οι τρόποι για τις εξής διαδικασίες:

- Εύρεση τοποθεσίας κλειδιού.
- Πώς συνδέονται νέοι κόμβοι.
- Πώς ανακτάται η κανονική λειτουργία μετά από βλάβη ή προγραμματισμένη αποχώρηση κάποιου κόμβου.

Στην αρχή περιγράφεται μία απλή εκδοχή του αλγορίθμου, ο οποίος δε χειρίζεται διαδοχικές συνδέσεις και αποχωρήσεις.

Πρακτικά υπολογίζει μία συνάρτηση κατατεμαχισμού (hash function) για να κάνει αντιστοίχηση (map) των κλειδιών σε κόμβους.

Γίνεται χρήση του consistent hashing: Το πλεονέκτημά του έναντι του απλού hashing είναι ότι όταν αλλάζει το μέγεθος του hash table (δηλαδή στην περίπτωσή μας το πλήθος των κόμβων, που γενικά θα συμβαίνει σχετικά συχνά) τότε δε χρειάζεται να αλλάξουν οι αντιστοιχήσεις όλων των κλειδιών, αλλά μόνο των K/N (κατά μέσο όρο), όπου K το πλήθος όλων των κλειδιών και N το πλήθος των κόμβων.

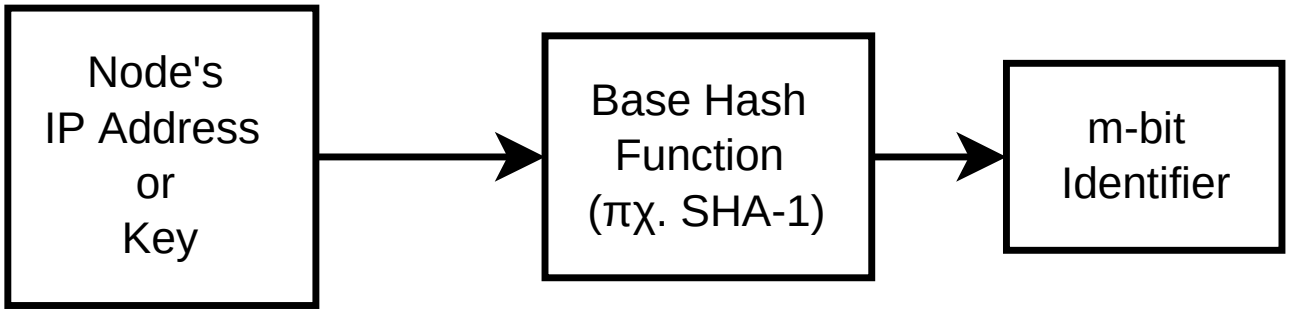
2.3.1.2 Συνεπής κατατεμαχισμός (Consistent hashing)

Πώς λειτουργεί το consistent hashing (με περιγραφικό τρόπο, υπάρχει αναλυτικότερη επεξήγηση στη συνέχεια): Έχουμε έναν κύκλο, στον οποίο έχουμε τοποθετήσει “κουβάδες” σε περίπου ισαπέχουσες γωνίες. Για ένα κλειδί βρίσκουμε μία γωνία του κύκλου στην οποία αντιστοιχεί σύμφωνα με κάποια συνάρτηση για αυτό το σκοπό. Το κλειδί θα “πέσει” στον κουβά που είναι ακριβώς μετά από το κλειδί αυτό όσον αφορά τη γωνία του. Έτσι, κάθε κουβάς περιέχει όλα τα κλειδιά που οι αντίστοιχες γωνίες τους βρίσκονται ανάμεσα σε αυτόν και στον προηγούμενο κουβά.

Όταν κάποιος κουβάς φύγει, τότε τα κλειδιά αυτόματα “πέφτουν” στον επόμενο κουβά. Όμοια, όταν ένας νέος κουβάς μπαίνει στον κύκλο, όλα τα κλειδιά που βρίσκονται ανάμεσα σε αυτόν και στον προηγούμενό του θα ανήκουν τώρα σε αυτόν. Δηλαδή παίρνει την ευθύνη μερικών κλειδιών για τα οποία είχε ευθύνη ο επόμενος του.

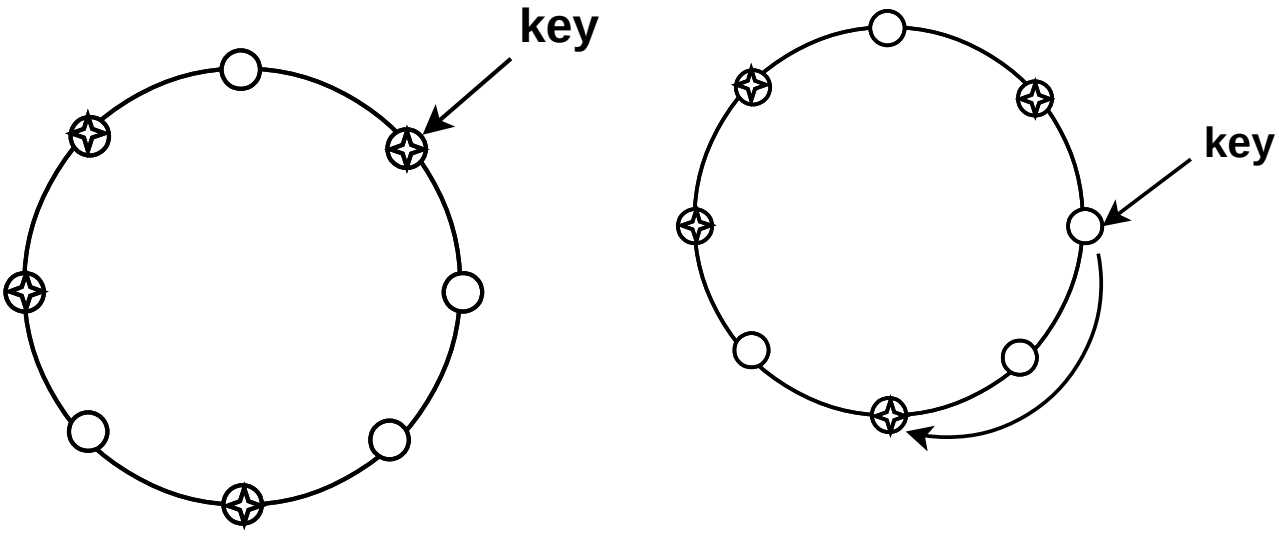
Το chord βελτιώνει το consistent hashing με την απαίτηση ότι κάθε κόμβος θα γνωρίζει την ύπαρξη μερικών άλλων κόμβων του δικτύου, για συντομότερες συνδέσεις. Αυτό γίνεται με χρήση των λεγόμενων finger tables, τα οποία εξηγούνται καλύτερα σε επόμενη παράγραφο.

Πιο συγκεκριμένα, εφαρμόζοντας την προηγούμενη λογική, εδώ θα έχουμε m-bit identifiers. Για τους κόμβους, το identifier τους θα προκύπτει εφαρμόζοντας μία hash συνάρτηση που θα χρησιμοποιούμε στο IP address το κόμβου. (δεν έχει οριστεί ποια θα χρησιμοποιείται, αλλά είναι εκτός συζήτησης εδώ). Για τα κλειδιά χρησιμοποιούμε την hash με είσοδο απλά την τιμή του κλειδιού. Σημαντικό είναι να προκύπτουν πάντα identifiers με m-bits.



Οπότε, τελικά περνάμε μέσα από το σύστημα base hash function όλα τα IP addresses των κόμβων και όλα τα κλειδιά, και έχουμε μόνο m-bit identifiers. Με αυτόν τον τρόπο, γίνεται δυνατή η άμεση αντιστοίχηση (mapping) των κλειδιών σε κόμβους, αφού μπορούμε άμεσα να συγκρίνουμε δύο αριθμούς που έχουν τα ίδια bits και αναλόγως να πάρουμε αποφάσεις. Στη συνέχεια δείχνεται πως ακριβώς γίνεται αυτό.

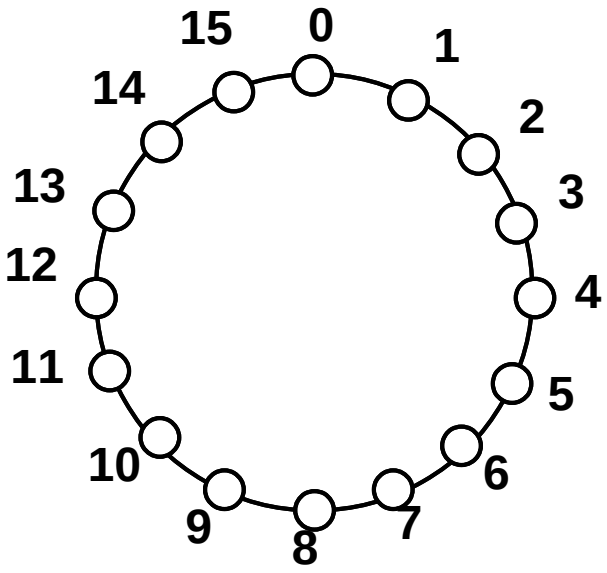
Προκειμένου να έχουμε αντιστοιχία μεταξύ "κουβάδων" και κόμβων, δημιουργούμε αρχικά έναν identifier circle, με 2^m ισαπέχοντα σημεία (φαίνεται στο επόμενο σχήμα, όπου έχουμε m=3, οπότε προκύπτουν 8 δυνατές τιμές για identifiers, που στο σχήμα φαίνονται ως μικροί κύκλοι). Οι κόμβοι παίρνουν θέση σε κάποια από τα σημεία, στις θέσεις που έχουν προκύψει κάνοντας hash τα IP addresses τους (δεν υπάρχει απαραίτητα κόμβος σε κάθε σημείο του κύκλου). Τα κλειδιά θα μπαίνουν στο σημείο του κύκλου που τους αντιστοιχεί και ο κάθε κόμβος είναι υπεύθυνος για όλα τα κλειδιά που βρίσκονται μεταξύ αυτού και του προηγούμενού του. Αν το κλειδί πέφτει ακριβώς πάνω σε κόμβο, τότε αυτός θα είναι υπεύθυνος για αυτό. Αν το κλειδί πέφτει σε σημείο όπου δεν υπάρχει κόμβος, τότε ο πρώτος κόμβος που θα βρει το κλειδί κινούμενο με τη φορά του ρολογιού θα είναι υπεύθυνος για αυτό. Στο επόμενο σχήμα δείχνονται οι δύο αυτές περιπτώσεις, όπου οι αστερίσκοι είναι υπάρχοντες κόμβοι στο δίκτυο.



Ο κόμβος για τον οποίον ισχύει η προαναφερθείσα ιδιότητα για κάποιο κλειδί k, θα λέμε ότι είναι successor(k).

Παράδειγμα:

Έστω m=4. Τότε θα έχουμε $2^4 = 16$ δυνατούς identifiers. Σχηματικά, έχουμε το εξής.



Έστω, επίσης, ότι έχουμε 4 κόμβους, από τους οποίους όταν περάσουμε τις IP addresses τους από μία βασική hash συνάρτηση προκύπτουν οι identifiers 0, 4, 10 και 13 αντίστοιχα. Τα κλειδιά που υπάρχουν στο δίκτυο, αφού περαστούν από την ίδια hash συνάρτηση, έχουν identifiers 0, 3, 4, 5 και 11. Σημαντικό εδώ είναι να τονίσουμε ότι γενικά τα κλειδιά και οι κόμβοι δεν έχουν κάποια συσχέτιση ως προς το σκοπό τους. Δηλαδή, αν ένα κλειδί έχει ίδια τιμή με κόμβο, στις περισσότερες περιπτώσεις, θα είναι τυχαίο. Εφαρμόζοντας την successor(i) συνάρτηση σε κάθε κλειδί, ώστε να βρούμε σε ποιον κόμβο αντιστοιχεί το καθένα, προκύπτει:

successor(0) = 0

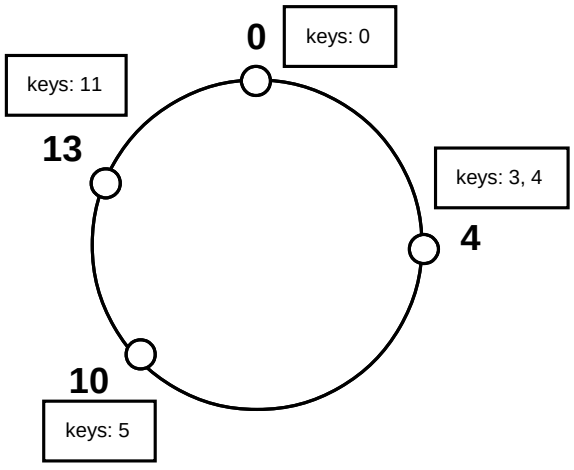
successor(3) = 4

successor(4) = 4

successor(5) = 10

successor(11) = 13

Τοποθετώντας τα δεδομένα αυτά στον identifier circle, έχουμε το δίκτυο που ακολουθεί (όπου, τώρα, βάζουμε τους κόμβους απλά με μικρούς κύκλους για να είναι λιγότερο πυκνό και δυσνόητο το σχήμα).

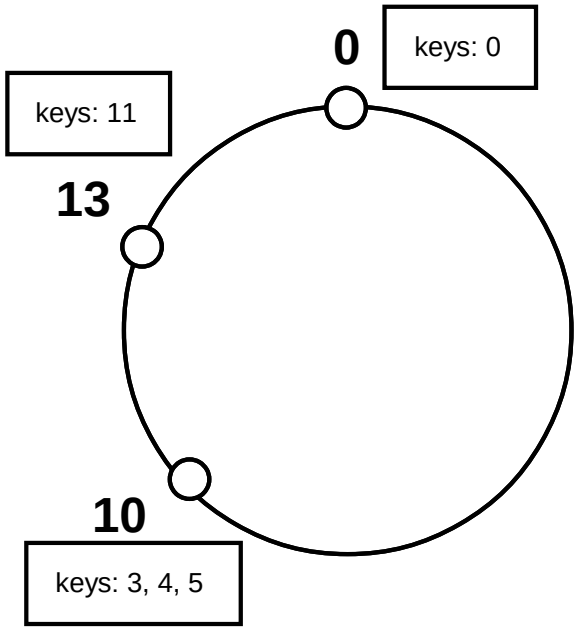


Όπως είπαμε, σε τέτοια δίκτυα θα υπάρχουν αναχωρήσεις και αφίξεις κόμβων. Πρέπει να δούμε πώς θα γίνει η προσαρμογή του υπόλοιπου δικτύου όταν συμβαίνουν τέτοιες διαδικασίες.

2.3.1.3 Χειρισμός αλλαγών κόμβων στο δίκτυο

Αναχώρηση Κόμβου από το δίκτυο (Node Leave)

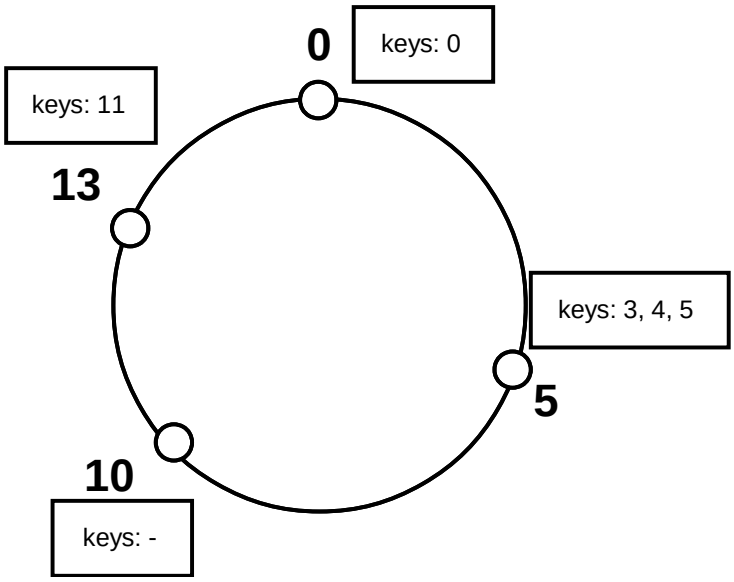
Όταν ένας κόμβος n (n είναι ο identifier που προέκυψε από το hashing) φύγει από το δίκτυο, τότε τα κλειδιά που είχαν ανατεθεί σε αυτόν θα ανατεθούν στον successor(n). Αυτή αποτελεί λογική λειτουργία, αφού θέλουμε να διατηρήσουμε τη δομή του δικτύου, όπου κάθε κλειδί πάει στον πρώτο κόμβο που βρίσκει κινούμενο με τη φορά του ρολογιού. Στο δίκτυο του προηγούμενου σχήματος, αν φύγει ο κόμβο 4, τότε τα κλειδιά του 3 και 4 θα πάνε στον κόμβο successor(4) = 10. Οπότε, θα προκύψει το επόμενο δίκτυο του σχήματος.



Άφιξη Κόμβου στο δίκτυο (node join):

Όταν ένας κόμβος n συνδεθεί με το υπάρχον δίκτυο, τότε κάποια κλειδιά τα οποία έχει ο κόμβος successor(n) θα ανατεθούν τώρα στον n. Όπως είπαμε, θέλουμε να ισχύει ότι κάθε κλειδί πάει στον πρώτο κόμβο που βρίσκει. Επομένως, προκειμένου να διατηρείται αυτό, ο νέος κόμβος n θα αναφερθεί στον κόμβο successor(n) και θα πάρει από αυτό όλα τα κλειδιά k για τα οποία ισχύει k<=n.

Στο δίκτυο του προηγούμενου σχήματος, έστω ότι πρόκειται να συνδεθεί ο κόμβος με identifier 5. Θα αναφερθεί στον κόμβο successor(5) = 10 και θα του πάρει κάποια κλειδιά. Παρατηρούμε ότι τα κλειδιά που έχει ο κόμβος 10 είναι με identifiers 3, 4 και 5 που όλοι είναι μικρότεροι ή ίσοι του 5, οπότε όλα τα κλειδιά θα πάνε στον κόμβο 5 και ο κόμβος 10 δε θα έχει κανένα κλειδί. Σχηματικά, έχουμε το εξής.

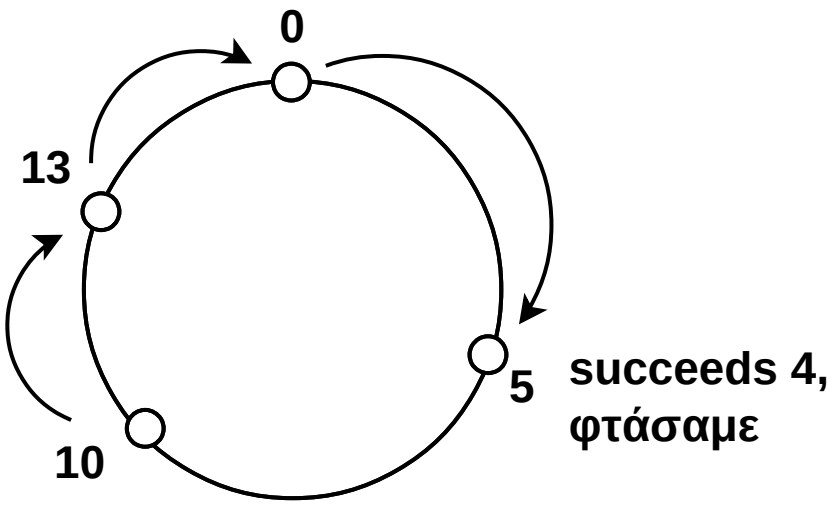


Σύμφωνα με την υλοποίηση αυτή, προκύπτει ότι:

- Κάθε κόμβος είναι υπεύθυνος για (κατά μέσο όρο) K/N κλειδιά.
- Όταν φύγει ή έρθει κάποιος κόμβος, τότε αλλάζει η ευθύνη για O(K/N) κλειδιά, και αυτές οι αλλαγές γίνονται μόνο από ή προς τον κόμβο που έφυγε ή ήρθε.

Η ελάχιστη απαιτούμενη γνώση κάθε κόμβου για το υπόλοιπο δίκτυο, ώστε να λειτουργούν σωστά οι διαδικασίες του Chord, είναι να γνωρίζει μόνο τον επόμενό του κόμβο. Όταν ικανοποιείται αυτή η απαίτηση, τότε είναι σίγουρο ότι μπορεί να γίνει επικοινωνία μεταξύ όλων των στοιχείων του δικτύου.

Για παράδειγμα, στο τελευταίο σχήμα, αν ο κόμβος 10 θέλει να βρει ποιος κόμβος είναι ο successor(4), τότε κινούμενος σειριακά θα ρωτήσει τον 13, με τη σειρά του ο 13 θα ρωτήσει τον 0, και ο 0 θα δει ότι ο successor(0) = 5 είναι και successor(4), οπότε θα επιστρέψει τη ζητούμενη πληροφορία.



Όμως, αν υπάρχει μόνο αυτή η σύνδεση, τότε η μεταφορά μηνυμάτων και εύρεση κόμβων γίνεται πολύ πιο αργά από αυτήν που υπόσχεται το Chord. Πιο συγκεκριμένα, η πολυπλοκότητα της διαδικασίας εύρεσης κόμβου θα είναι O(N), δηλαδή γραμμική, αφού στη χειρότερη περίπτωση θα χρειαστεί να γίνει προσπέλαση όλων των κόμβων. Στο Chord,

προκειμένου να επιταχυνθεί η διαδικασία, πάντα θα υπάρχει σύνδεση με περισσότερους κόμβους, όπως εξηγείται στα επόμενα.

2.3.1.4 Επέκταση διαδικασιών του δικτύου με χρήση finger tables

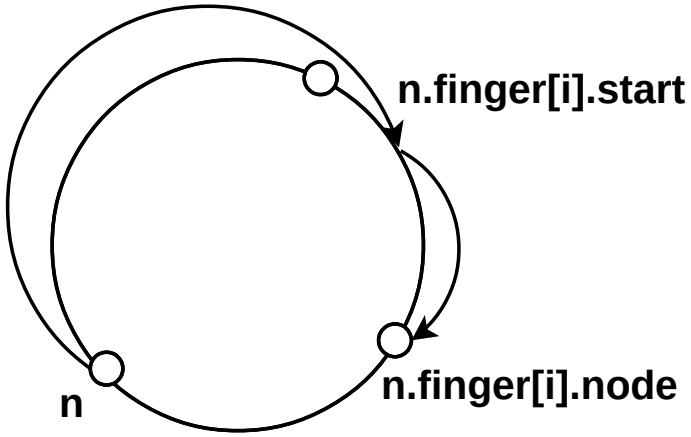
Κάθε κόμβος θα έχει έναν πίνακα δρομολόγησης (routing table) με το πολύ m καταχωρήσεις (όπου m είναι τα μέγεθος σε bits των identifiers). Η i-οστή καταχώρηση έχει το αναγνωριστικό του κόμβου s = successor($n + 2^{i-1}$), $0 < i < m+1$ και την IP address του. Ο κόμβος s, περιγραφικά, είναι ο πρώτος κόμβος που έχει απόσταση τουλάχιστον 2^{i-1} από τον n.

Το s θα ονομάζεται i-οστό finger του κόμβου n, και θα έχει σύμβολο s = n.finger[i].node. Το 1ο finger του κόμβου είναι ο successor του. Μερικοί άλλοι συμβολισμοί:

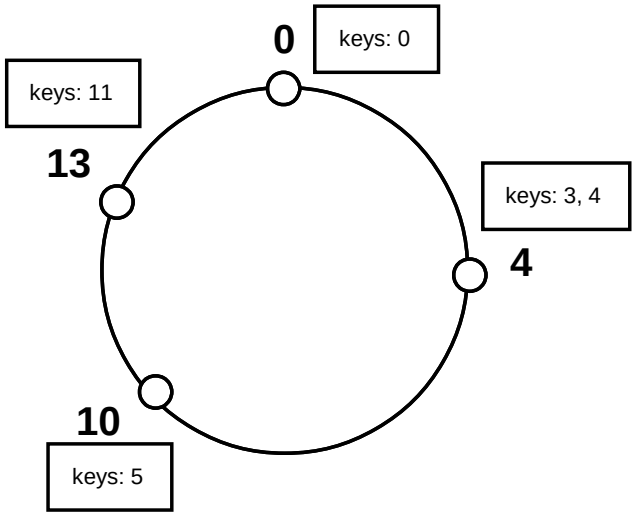
- s = n.finger[i].node
- n.finger[i].start = $n + 2^{i-1}$ (δείχνει την απόσταση αυτή)
- n.finger[i].interval = [n.finger[i].start , n.finger[i+1].start)
- n.successor(i) = n.finger[1].node

Εδώ σημειώνουμε ότι όλες οι πράξεις είναι με λογική modulo 16, ώστε τα αριθμητικά αποτελέσματα να βρίσκονται πάντα στο διάστημα [0, 16), δηλαδή μέσα στους δυνατούς identifiers.

Σχηματικά, για το i-οστό finger του n, έχουμε:



Παρατηρούμε από τον τύπο του πίνακα δρομολόγησης ότι κάθε κόμβος γνωρίζει περισσότερα για τους πιο κοντινούς του, επειδή υπάρχει το εκθετικό 2^{i-1} και άρα διπλασιάζεται η απόσταση του επόμενου κόμβου ανά βήμα με τον οποίον θα είναι συνδεδεμένος. Έτσι, αν χρειαστεί να επικοινωνήσει με έναν κόμβο k τον οποίον δεν έχει στο finger table του, τότε θα ρωτήσει τον κόμβο από τον πίνακά του ο οποίος είναι ο πιο κοντινός προηγούμενος του k, επειδή θα υπάρχει μεγαλύτερη πιθανότητα να γνωρίζει αυτός (λόγω του εκθετικού 2^{i-1}). Στη συνέχεια παρατίθεται ένα παράδειγμα, με το πρώτο δίκτυο που αναλύσαμε.



Τα finger tables για τον κάθε κόμβο δίνονται στους επόμενους πίνακες.

node 0: start	interval	successor
1	[1,2)	4
2	[2,4)	4
4	[4,8)	4
8	[8,0)	10

keys: 0

node 4: start	interval	successor
5	[5,6)	10
6	[6,8)	10
8	[8,12)	10
12	[12,4)	13

keys: 3, 4

node 10: start	interval	successor
11	[11,12)	13
12	[12,14)	13
14	[14,2)	0
2	[2,10)	4

keys: 5

node 13: start	interval	successor
14	[14,15)	0
15	[15,1)	0
1	[1,5)	4
5	[5,13)	10

keys: 11

Πώς γίνεται η επικοινωνία με χρήση των finger tables:

Στο προηγούμενο παράδειγμα, έστω ότι ο κόμβος 10 θέλει να βρει ποιος κόμβος είναι ο successor(1). Πάει στο finger table του και πρώτα κοιτάει την πρώτη στήλη με τα starts. Παρατηρεί ότι δεν υπάρχει άμεσα η πληροφορία (δεν υπάρχει το 1 στη στήλη με τα starts, ώστε να βρει άμεσα το successor(1)), οπότε πάει στα intervals. Έχουμε ότι το 1 ανήκει στο διάστημα 10.finger[3].interval=[14,2), οπότε ο 10 θα ρωτήσει τον κόμβο 10.finger[3].node=0. Αν το 10.finger[3].node ήταν μεγαλύτερο του 1, τότε ο 10 θα ρωτούσε το προηγούμενο finger του. Ο κόμβος 0 βλέπει άμεσα από το finger table του ότι successor(1) = 0.finger[1].node = 5. Οπότε το επιστρέφει στον 10 ως πληροφορία.

Αποδεικνύεται ότι αυτή η διαδικασία αναζήτησης που περιγράφηκε έχει πολυπλοκότητα O(logN). Διαισθητικά, παρατηρούμε ότι στο finger table, η κάθε διαδοχική καταχώρηση έχει start με διπλάσια απόσταση από την προηγούμενη. Οπότε, ανά βήμα αναζήτησης κόμβου, η απόσταση θα μειώνεται, κατά μέσο όρο, στο μισό. Έτσι προκύπτει και το O(logN). Αν το δούμε από πλευρά χειρότερης περίπτωσης, το τελευταίο finger έχει start ίσο με το μισό του πλήθους των identifiers. Στην περίπτωσή μας, με 16 identifiers, το start της τελευταίας καταχώρησης είναι n + 8. Οπότε, στη χειρότερη περίπτωση, στο πρώτο βήμα θα καλύψουμε το μισό κύκλο. Στο δεύτερο βήμα, στη χειρότερη περίπτωση, θα καλύψουμε το μισό του μισού κύκλου, κλπ.

Κατά τη σύνδεση ή αποχώρηση νέου κόμβου στο δίκτυο, πρέπει να διατηρείται το ελάχιστο απαιτούμενο, δηλαδή να υπάρχει το σωστό node successor και για κάθε κλειδί k ο κόμβος successor(k) να είναι υπεύθυνος για αυτό. Για να γίνονται οι διαδικασίες πιο αποδοτικά, αλλά χωρίς αυτό να αποτελεί ανάγκη για τη σωστή λειτουργία, είναι οι κόμβοι να έχουν σωστά finger tables.

Αποδεικνύεται ότι όταν ένα node κάνει join / leave, χρειάζονται να σταλθούν $O(\log^2 N)$ μηνύματα.

Ως επιπλέον επέκταση, για απλοποίηση των διαδικασιών, κάθε κόμβος έχει και έναν predecessor pointer, που δείχνει στον αμέσως προηγούμενο κόμβο του.

Σύνδεση νέου κόμβου με χρήση finger tables

Οι διαδικασίες που γίνονται όταν ένας νέος κόμβος πρόκειται να συνδεθεί στο υπάρχον δίκτυο είναι:

α. Αρχικοποίηση των στοιχείων του κόμβου n

β. Ανανέωση των υπόλοιπων κόμβων του δικτύου

γ. Ενημέρωση του λογισμικού υψηλότερου επιπέδου

Στα επόμενα εξηγούνται οι τρεις αυτές διαδικασίες, με παραδείγματα.

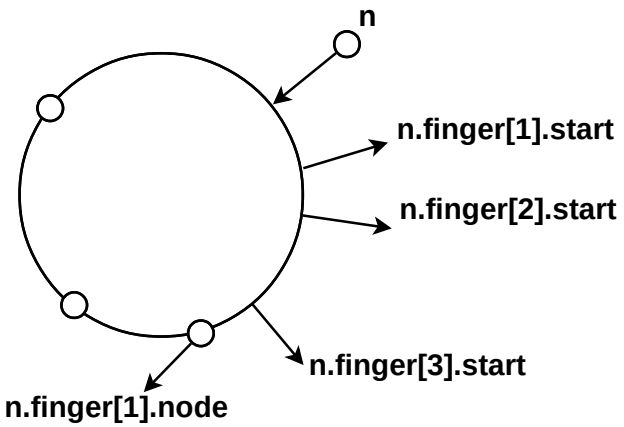
α. Αρχικοποίηση των στοιχείων του κόμβου n:

Θεωρούμε ότι ο κόμβος n γνωρίζει έναν άλλον κόμβο n' που υπάρχει ήδη στο δίκτυο, μέσω κάποιου εξωτερικού μηχανισμού. Ο κόμβος n θα ζητήσει από τον n' να βρει τα στοιχεία του finger table του n. Αυτό μπορεί να γίνει με δύο τρόπους. Ο πρώτος τρόπος είναι ο n' να καλέσει μία συνάρτηση find_successor(i) για κάθε i του finger table του n (η συνάρτηση find_successor(i) βρίσκει τον κόμβο successor(i)). Αυτή είναι απλή διαδικασία, αλλά αργή. Πιο αποδοτικό είναι να καλεί την find_successor για το πρώτο finger και για το επόμενο finger εκτελεί το εξής:

Αν n.finger[i].node >= n.finger[i+1].start,

τότε n.finger[i].node = n.finger[i+1].node.

Φαίνεται στο ακόλουθο σχήμα πότε ισχύει αυτή η ανίσωση.



Έστω ότι ο κόμβος n θέλει να συνδεθεί στο δίκτυο. Καλείται η find_successor(1) και επιστρέφει το n.finger[1].node. Για i=2 παρατηρούμε ότι ισχύει η ανίσωση n.finger[1].node=n.finger[2].start, οπότε θα έχουμε n.finger[2].node=n.finger[1].node. Όμοια για το 3ο finger, επειδή ισχύει η ανίσωση και για αυτό, θα έχουμε n.finger[3].node=n.finger[2].node.

β. Ανανέωση των υπόλοιπων κόμβων του δικτύου:

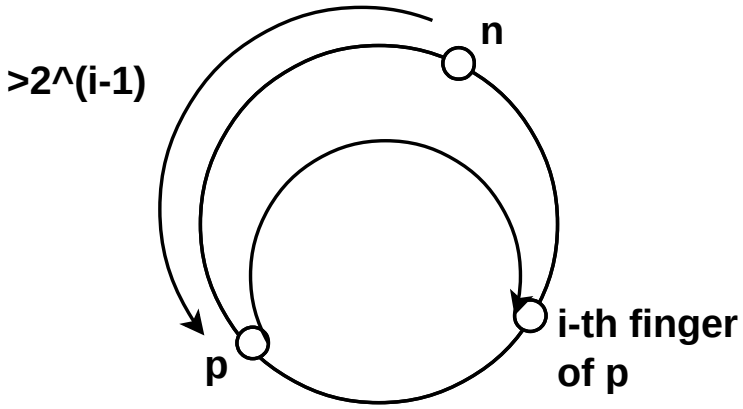
Η σχέση που δείχνει σε ποιους κόμβους θα γίνει finger ο νέος κόμβος n είναι:

Ο νέος κόμβος n θα είναι το i-οστό finger του ήδη υπάρχοντος στο δίκτυο κόμβου p αν και μόνο αν:

- Ο p βρίσκεται πριν το n τουλάχιστον 2^{i-1}.

- Το i-οστό finger του p ακολουθεί αμέσως το n, δηλαδή successor(n) = p.finger[i].node.

Σχηματικά, φαίνεται η περίπτωση που ισχύουν και οι δύο αυτές προϋποθέσεις.



Αυτές οι υποθέσεις μπορούν να ελέγχονται αυτόματα και αποδοτικά με τον ακόλουθο αλγόριθμο σε ψευδοκώδικα, ο οποίος αποτελείται από δύο συναρτήσεις:

n.update_others(){

for i=1 to m

p = find_predecessor($n - 2^{i-1}$)

p.update_finger_table(n,i)}

p.update_finger_table(n,i){

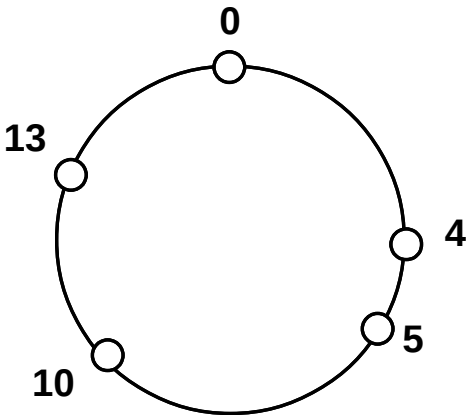
if (n belongs [p , finger[i].node))

{finger[i].node = n

p = predecessor

p.update_finger_table(n,i)}}

Στο επόμενο παράδειγμα φαίνεται καλύτερα η λειτουργία του αλγόριθμου αυτού. Έστω ότι έχουμε δίκτυο με κόμβους 0, 4, 10 και 13 και μπαίνει νέος κόμβος 5.



Για i=1:

p = find_predecessor($5 - 2^{1-1}$) => p = 4

Το 5 ανήκει στο διάστημα [4 , 4.finger[1].node) = [4 , 10) άρα θέτουμε 4.finger[1].node = 5 και πάμε πιο πίσω στον κύκλο p = predecessor(4) = 0.

Είμαστε i=1 και p=0:

Το 5 δεν ανήκει στο διάσημα [0 , 0.finger[1].node) = [0 , 4) άρα σταματάμε εδώ για i=1.

Για i=2 και i=3 προκύπτει ότι δεν γίνεται κάποια αλλαγή.

Για i=4:

p = find_predecessor($5 – 2^{4-1}$) => p = 13 (Όπως είπαμε, οι πράξεις γίνονται modulo 16, οπότε για το 5-8 = -3 θα κινηθούμε πάνω στον κύκλο -3 θέσεις από το 0 και θα φτάσουμε στο 13.)

Το 5 ανήκει στο διάστημα [13 , 13.finger[4].node) = [13 , 10) άρα θέτουμε 13.finger[4].node = 5 και πάμε πιο πίσω στον κύκλο p = predecessor(13) = 10.

Είμαστε i=4 και p=10:

Το 5 δεν ανήκει στο διάστημα [10, 10.finger[4].node) = [10 , 4) άρα σταματάμε εδώ για i=4 και συνολικά.

γ. Ενημέρωση του λογισμικού υψηλότερου επιπέδου

Εκτός από την ανανέωση των κόμβων, πρέπει να γίνει και ανανέωση των κλειδιών που υπάρχουν στο δίκτυο, μεταφέροντας αυτά ώστε τελικά κάθε κλειδί να βρίσκεται στον κόμβο που βρίσκει πρώτο κινούμενο στον identifier circle. Η διαδικασία αυτή υλοποιείται από το λογισμικό υψηλότερου επιπέδου, δηλαδή από τους κόμβους του δικτύου. Τα δυνατά προς μετακίνηση κλειδιά βρίσκονται μόνο στον successor του νέου κόμβου, οπότε χρειάζεται να γίνει αναφορά μόνο σε αυτόν.

Ταυτόχρονες λειτουργίες

Όπως ήδη αναφέρθηκε, στο δίκτυο το οποίο χρησιμοποιεί το Chord, γίνονται αναχωρήσεις και αφίξεις σε τακτές και απρόβλεπτες χρονικές στιγμές. Προβληματική περίπτωση, η οποία χρειάζεται ειδικό χειρισμό, είναι όταν γίνονται ταυτόχρονες αναχωρήσεις ή αφίξεις. Σε αυτή την περίπτωση, ακολουθείται ένα πρωτόκολλο σταθεροποίησης, που περιγράφεται στα επόμενα

Το βασικό πρωτόκολλο σταθεροποίησης με διαδοχικές διεργασίες είναι να διατηρούνται σωστοί οι δείκτες των successor κόμβων (το ελάχιστο για ορθότητα του δικτύου).

Σε γενικές γραμμές, υπάρχουν τρεις περιπτώσεις όταν κάνει κάποιος κόμβος αναζήτηση:

- Οι συνδέσεις, δείκτες και finger tables είναι όλα σχετικά σωστά, οπότε βρίσκει αυτό που θέλει σε O(logN).
- Είναι σωστά μόνο τα successor pointers, οπότε το βρίσκει αλλά πιο αργά (περίπου O(N)).
- Υπάρχουν κάποια λάθος successor pointers, το οποίο μπορεί να οδηγήσει σε αποτυχία αναζήτησης.

Γενικά, στο δίκτυο μπορεί να γίνουν διαδοχικές αφίξεις και αναχωρήσεις, με αποτέλεσμα να μην έχει προλάβει να ανανεωθεί πλήρως το δίκτυο πριν την άφιξη νέου κόμβου, και να υπάρχουν λάθος δείκτες. Για παράδειγμα, στο δίκτυο με τους κόμβους 0, 4, 10 και 13, αν

μπουν πρακτικά ταυτόχρονα νέοι κόμβοι 5 και 6, θα θεωρούν και οι δύο ως successor τους τον κόμβο 10, που είναι σωστό μόνο για τον κόμβο 6.

Το βασικό πρωτόκολλο σταθεροποίησης, ώστε να διατηρούνται σωστοί οι successor δείκτες είναι το εξής:

- Θέλει να συνδεθεί ο κόμβος n.
- Ζητάει από τον ήδη υπάρχον στο δίκτυο κόμβο n' για τα στοιχεία του n.
- Το δίκτυο δεν έχει ενημερωθεί ακόμα για τον n.
- Οι κόμβοι εκτελούν μία συνάρτηση *stabilize* ανά τακτά χρονικά διαστήματα, ώστε να ενημερωθούν σωστά.

Στη συνέχεια έχουμε ένα απλό παράδειγμα που δείχνει τη λειτουργία σύνδεσης κόμβου με χρήση της συνάρτησης *stabilize*.

- Έστω δίκτυο με κόμβους 0, 4, 10 και 13.
- Έστω ένας νέος κόμβος 5 θέλει να συνδεθεί.
- Ζητάει από κάποιον κόμβο n', οποίος υπάρχει ήδη στο δίκτυο, για τα στοιχεία του 5, οπότε ο 5 θεωρεί τον 10 ως successor.
- Ο 5 ενημερώνει τον 10 για την ύπαρξή του, οπότε ο 10 θέτει ως predecessor του τον 5.
- Ο 4 τρέχει σε κάποια χρονική στιγμή τη συνάρτηση stabilize, και ζητάει τον predecessor του successor του. Ο 4 νομίζει ότι ο successor του είναι ο 10 (που είναι λάθος), οπότε τον ρωτάει για τον predecessor του.
- Ο 10 απαντάει λέγοντας ότι ο predecessor του είναι ο 5.
- Ο 4 ενημερώνεται και θέτει ως successor του τον 5.
- Ο 4 ενημερώνει τον 5, ώστε ο 5 να θέσει ότι ο predecessor του είναι ο 4.
- Μία συνάρτηση fix fingers, την οποία επίσης τρέχουν όλοι οι κόμβοι του δικτύου ανά τακτά χρονικά διαστήματα, διορθώνει τα fingers των κόμβων.

Θεώρημα: Μόλις ένας κόμβος μπορεί να απαντήσει σε μία συγκεκριμένη ερώτηση, θα μπορεί την απαντήσει και στο μέλλον.

Θεώρημα: Κάποια στιγμή μετά την τελευταία σύνδεση, όλα τα successor pointers θα είναι σωστά.

Node Failures

Όταν ένας κόμβος αποτυγχάνει, τότε πρέπει να διατηρηθούν τουλάχιστον οι σωστοί successor pointers. Προκειμένου να ισχύει αυτό, κάθε κόμβος έχει μία λίστα με r successors. Όταν κάποιο successor αποτύχει, τότε δοκιμάζει το επόμενο στη λίστα.

Θεώρημα: Αν r = O(logN) σε αρχικά σταθερό δίκτυο, και στη συνέχεια κάθε κόμβος αποτυγχάνει με πιθανότητα 1/2, τότε με μεγάλη πιθανότητα η συνάρτηση find_successor θα βρίσκει το κοντινότερο “ζωντανό” successor.

Με την πάροδο του χρόνου, οι συναρτήσεις *stabilize* και *fix fingers θα* διορθώσει τα successor pointers και τα finger tables.

2.3.2 2ο σύστημα: Self-Organizing Distributed Sensor Networks

Το σύστημα αυτό περιγράφει έναν τρόπο αυτοοργάνωσης πολλών συστημάτων [70]. Ως προς τη δομή του δικτύου του συγκεκριμένου συστήματος, θεωρείται ότι η εμβέλεια του κάθε πομποδέκτη θα είναι πολύ μικρότερη από όλο το δίκτυο, οπότε θα έχουμε τοπολογία multi-hop (η πληροφορία που πρόκειται να σταλθεί μεταφέρεται από ενδιάμεσους κόμβους για να φτάσει το στόχο). Η κύρια μέτρηση της επίδοσης του συστήματος σχετίζεται με την αποδοτικότητα ισχύος σήματος. Θεωρείται ότι τα sensor nodes είναι γενικά ακίνητα, οπότε αξίζει (από άποψη ενέργειας) να γίνει προσπάθεια μία φορά μόνο για να δημιουργηθεί η δομή του δικτύου.

Το δίκτυο του συστήματος είναι βασισμένο σε δομή TDMA (Time Division Multiple Access), όπου χρησιμοποιείται μία συχνότητα, η οποία χωρίζεται σε χρονικά τμήματα, στα οποία θέτονται διάφορες λειτουργίες.

Τέτοιες λειτουργίες μπορεί να είναι:

1. Είσοδος/Πρόσβαση στο δίκτυο

2. Κόμβος να κάνει broadcast σε γειτονικούς κόμβους

3. Broadcast των γειτονικών κόμβων

4. point-to-point επικοινωνία

5. Ανάγκες για προσωρινό υψηλό bandwidth

6. Για άλλους σκοπούς που δεν κάνει το node αλλά χρησιμοποιείται από το node

7. Το υπόλοιπο bandwidth, που είναι κενό και είναι ελεύθερο

Στη συνέχεια περιγράφονται οι ρουτίνες αρχικοποίησης ενός τυχαίου κόμβου με σκοπό την αυτοοργάνωση του δικτύου:

- Αρχικοποίηση του node κατά το άναμα: Ο κόμβος εκτελεί ορισμένες ρουτίνες, όπως δοκιμή ορθής λειτουργίας του, προσδιορισμός καταστάσεών του (μπαταρία κλπ), και built-in calibration. Επίσης ξεκινά να εκτελεί διεργασίες που είναι προγραμματισμένο, όπως να παίρνει δείγματα από αισθητήρα.
- Ανακάλυψη δικτύου: Ο κόμβος ψάχνει για ήδη υπάρχον δίκτυο, ακούγοντας για invitations ώστε να συνδεθεί σε αυτό. Αν δεν ακούσει κάποιο invitation για κάποιο προκαθορισμένο χρονικό διάστημα, τότε θεωρεί ότι είναι το πρώτο node που ενεργοποιείται, οπότε αρχίζει να στέλνει invitations για άλλα nodes για να ενωθούν.
- Network entry access: Μόλις ακούσει κάποιο invitation, στέλνει response.
- Node-type announcement: Το node που έστειλε την πρόκληση θέτει ένα τμήμα στο TDMA για περαιτέρω επικοινωνία με το νέο node. Ανάλογα με τον τύπο που είναι ο κόμβος, θέτονται και οι παράμετροι επικοινωνίας. Πιο συγκεκριμένα, αν είναι κόμβος χρήστη, τότε θα δοθεί προτεραιότητα στην επικοινωνία με όλο το δίκτυο. Αν είναι κόμβος-αισθητήρας, τότε θα χρησιμοποιηθεί μία πιο αργή άλλα οικονομική διαδικασία.

- Program exchange: Αν το νέο node έχει κάποιο νέο πρόγραμμα για το δίκτυο, τότε το κατεβάζουν όλα τα άλλα στο δίκτυο. Αν, αντίθετα, το δίκτυο έχει κάτι να δώσει στο νέο node, τότε το ανεβάζει σε αυτό.
- Topology learning and position location: Συχνή τεχνική είναι να έχουμε έναν πίνακα μεγέθους Nx1, όπου N το πλήθος των nodes και να θέτουμε 1 block σε κάθε node. Το node χρησιμοποιεί αυτή τη θέση για να ενημερώσει ποιων άλλων nodes αντιλαμβάνεται την ύπαρξη άμεσα, δηλαδή είναι μέσα στην εμβέλειά του, οπότε τελικά με αυτόν τον πίνακα έχουμε μία κατανόηση της τοπολογίας του δικτύου. Καλός αλγόριθμος για δίκτυα με πολλά κινητά nodes, και για την περίπτωση που όλο το δίκτυο ανάβει ταυτόχρονα.
- Neighbor TDMA scheduling: Αφού γίνει γνωστή η τοπολογία, τότε πρέπει να οργανωθούν για το πότε θα στέλνουν το καθένα σήματα. Άρα φτιάχνεται το πρόγραμμα TDMA.
- Subnetwork merging: Υπάρχει η περίπτωση ένα νέο node να ενώνει δύο υποδίκτυα με την τοποθεσία του. Είναι αποδοτικό να προσπαθήσει να συνδεθεί και με κάποιο τρόπο να ενημερώσει το νέο αυτό μονοπάτι που μπορούν να ακολουθήσουν Θα συνδεθεί με το ένα υποδίκτυο πρώτα, και μετά θα συνεχίσει να ακούει για invitations. Αν ακούσει, τότε προσπαθεί να συνδεθεί και με το άλλο υποδίκτυο. Στη συνέχεια, θα είναι αυτό υπεύθυνο να οργανώσει και να συγχρονίσει τα δύο υποδίκτυα.
- Traffic determination and routing: Το νέο node πρέπει να ενημερώσει το δίκτυο για τις ανάγκες του για μετακίνηση πληροφορίας.
- Network TDMA scheduling: Αφού τεθούν οι ανάγκες πληροφορίας στις συνδέσεις, χρησιμοποιούνται για να φτιαχτούν TDMA προγράμματα που ικανοποιούν όλες τις ανάγκες.
- Normal operation: Αφού γίνουν όλα, τότε το node είναι ενσωματωμένο στο δίκτυο. Πρέπει όμως να εκτελεί και διαδικασίες συντήρησης (πχ. να στέλνει invitations για άλλα nodes).

2.3.3 3ο σύστημα: Protocols for Self-Organization of a Wireless Sensor Network

Το βασικό σενάριο λειτουργίας περιγράφεται στα επόμενα [71]. Το δίκτυο αισθητήρων πρέπει να λειτουργεί σε δυναμικές συνθήκες, δηλαδή τα πρωτόκολλα πρέπει να θέτουν τις διαδικασίες κατά το start-up, steady state, και failure. Αφού ανάψουν τα nodes και δημιουργηθεί το δίκτυο, είμαστε σε σταθερή κατάσταση. Τα περισσότερα nodes θα ανήκουν σε multi-hop δίκτυο, και θέτουν τους δρόμους από τους οποίους θα περνάει πληροφορία σε ένα ή περισσότερα sink nodes, που μπορεί να είναι κάποιος σταθμός μεγάλης εμβέλειας.

Υπάρχουν οι διαδρομές sensor-to-sink και sink-to-sensor, αλλά κυρίως γίνεται η πρώτη. Έτσι, όμως, τα nodes που βρίσκονται κοντά στο sink θα εκτελούν πολύ περισσότερες λειτουργίες, και είναι πιθανό να πέσει η μπαταρία. Πρέπει τα πρωτόκολλα routing και MAC (Medium Access Control, προσφέρει τις συνδέσεις για τη μεταφορά δεδομένων μεταξύ των nodes) να μπορούν να προσαρμόζονται και να αλλάζουν τις διαδρομές αποφεύγοντας αυτά τα πολυχρησιμοποιημένα nodes.

Σε αυτό το σημείο αναφέρονται, συνοπτικά, κάποια από τα πιο συνηθισμένα μοντέλα wireless network:

- Mobile Ad-hoc network (MANET): Ad-hoc σημαίνει ότι δεν βασίζεται σε προϋπάρχουσα υποδομή [72].

  1.Είναι point-to-point δίκτυο.

  2.Οι κόμβοι του δικτύου είναι όλοι κινητοί.

  3.Multi-hop δίκτυο.

  Για να έχουμε καλό QoS (quality of service) απέναντι στην κινητικότητα, πρέπει:

  1.Να οργανώνεται το δίκτυο ώστε να μπορούν όλες οι συσκευές να έχουν πρόσβαση σε κάθε μέσο επικοινωνίας.

  2.Να γίνεται αποδοτική δρομολόγηση των δεδομένων.

  3.Να διατηρεί το δίκτυο την οργάνωση παρά την κινητικότητα.

- Cellular network: Σε αυτό το δίκτυο υπάρχουν κάποια ακίνητα nodes (base stations), είναι συνδεδεμένα ενσύρματα, και δημιουργούν μία σταθερή υποδομή. Τα κινητά nodes είναι πολύ περισσότερα, και θα είναι μόνο ένα hop μακριά από κάποιο base station. Τα base stations έχουν άπειρη παροχή ενέργειας, ενώ τα κινητά είναι με μπαταρία.
- Bluetooth: Το δίκτυο αυτό είναι μικρής εμβέλειας, με κύριο σκοπό να αντικαταστήσει την ενσύρματη σύνδεση μεταξύ συσκευών. Η τοπολογία του δικτύου είναι η λεγόμενη star topology, όπου υπάρχει ένα master node, το οποίο μπορεί να έχει συνδεδεμένα μέχρι και 7 slave nodes. Η συνολική δομή με το ένα master node και τα μέχρι 7 slave nodes ονομάζεται piconet.

Στο δίκτυο του συστήματος του συστήματος που παρουσιάζουμε εδώ θεωρείται ότι υπάρχουν εκατοντάδες ακίνητα nodes, και μερικά κινητά. Για δίκτυο αισθητήρων, ενδιαφέρον έχει να επεκτείνουμε τη διάρκεια ζωής του κάθε node από πλευράς μπαταρίας. Άρα είναι πιθανό να μειώσουμε την επίδοση των συσκευών που βρίσκονται στο δίκτυο ώστε να μειώσουμε και την κατανάλωσή τους. Η κατανάλωση των συσκευών του δικτύου χωρίζεται σε 3 κατηγορίες: sensing, data processing, και communications. Η περισσότερη κατανάλωση, κατά μεγάλο βαθμό, γίνεται από την επικοινωνία. Για μείωσή της, γίνονται περισσότεροι υπολογισμοί στο κάθε τοπικό node, ώστε να μην χρειάζεται να σταλθεί τόση πολλή και περίπλοκη πληροφορία.

Τα nodes μπορούν να ανάψουν και να σβήσουν τους πομποδέκτες τους με εντολή. Μπορούν να θέσουν το carrier που χρησιμοποιείται για την ασύρματη μεταφορά των δεδομένων σε διαφορετικές συχνότητες. Υπάρχουν σχετικά πολλές τέτοιες συχνότητες.

Ένα κανάλι για επικοινωνία μεταξύ δύο κόμβων του δικτύου τίθεται ως ένα χρονικό διάστημα, ορισμένο από ένα ζευγάρι χρονικών στιγμών.

Η διαδικασία δημιουργίας του δικτύου για το οποίο γίνεται προσπάθεια αυτοοργάνωσης ξεκινάει ως εξής: τοποθετούνται τα nodes τυχαία, και μετά ανάβουν σε τυχαίους χρόνους.

Στη συνέχεια περιγράφονται τα πρωτόκολλα που έχουν συγγραφεί από τους δημιουργούς του συστήματος αυτού με σκοπό την αντιμετώπιση τέτοιων δικτύων.

- **Πρωτόκολλο SMACS** (Self-organizing Medium Access Control for Sensor networks) είναι υποδομή που δημιουργεί επίπεδη τοπολογία. Είναι distributed protocol. Τα nodes ανακαλύπτουν και οργανώνονται με τους γείτονές τους αυτόνομα, χωρίς κάποιο master node.

Έχουν συνδυαστεί οι φάσεις ανακάλυψης γειτόνων και ανάθεσης καναλιών, με σκοπό την εξοικονόμηση χρόνου και ενέργειας. Δηλαδή τίθεται κανάλι αμέσως μετά την ύπαρξη κάποιας σύνδεσης.

Τελικά, πρέπει να υπάρχει τουλάχιστον ένα multi-hop μονοπάτι που να ενώνει οποιαδήποτε δύο nodes.

Αφού φτιαχτεί κάποια σύνδεση, το node γνωρίζει πότε χρειάζεται να ανάψει και να σβήσει το ράδιο του. Έτσι, το κρατάει σβηστό στο υπόλοιπο χρονικό διάστημα για εξοικονόμηση ενέργειας.

Τα nodes ξυπνούν σε τυχαίους χρόνους. Περιμένουν μήνυμα πρόσκλησης. Αν, μετά από συγκεκριμένο χρόνο, δεν ακούσουν τίποτα, στέλνουν αυτά μήνυμα πρόσκλησης (TYPE1 message). Για παράδειγμα, έστω ότι αυτό κάνει το node A. Όσα λάβουν το μήνυμα, απαντούν με TYPE2 message, έστω ότι συμβαίνει αυτό σε δύο nodes, τα B και C. Αν οι απαντήσεις τους δεν συγκρουστούν, τότε το A ακούει και τις δύο απαντήσεις. Επιλέγει ποιο από τα δύο θα συνδεθεί πρώτα, σύμφωνα με κάποιο αλγόριθμο. Έστω ότι επιλέγει το B. Το A τότε στέλνει TYPE3 message για να ενημερώσει ποιο επέλεξε. Το C, που δεν επιλέχθηκε, κοιμάται για ένα χρονικό διάστημα, και μετά ξαναπροσπαθεί. Αν το A ήταν ήδη συνδεδεμένο στο δίκτυο, τότε στέλνει με πληροφορίες για το πρόγραμμά του. Το B διαβάζει το πρόγραμμα και θέτει ένα χρονικό διάστημα που το A είναι ελεύθερο για επικοινωνία μεταξύ τους.

Αλγόριθμος EAR (eavesdrop and register):

Υπάρχει ένας σταθερός κόμβος στο δίκτυο, ο οποίος στέλνει invitation message. Ο κινητός κόμβος κάνει eavesdrop το MAC (Medium Access Control) protocol (βλέπει SNR κλπ. ώστε να επιλέξει τον κόμβο με τον οποίο θα έχει την καλύτερη σύνδεση).

Τα σήματα που στέλνονται είναι:

Broadcast invite (BI): Το stationary στέλνει μήνυμα πρόσκλησης για να συνδεθούν άλλα νέοι κόμβοι.

Mobile Invite (MI): Ο κινητός κόμβος αποκρίνεται στην πρόσκληση, και στέλνει μήνυμα αίτησης για σύνδεση στο δίκτυο.

Mobile Response (MR): Ο σταθερός κόμβος δέχεται την αίτηση του MI.

Mobile Disconnect (MD): Ο κινητός κόμβος ενημερώνει για κάποια αποσύνδεση. Δε χρειάζεται απάντηση από τον σταθερό κόμβο.

Γενικά, ο κινητός κόμβος θα δεχθεί πολλά BIs. Αντί να απαντήσει, κρατάει τις πληροφορίες που δέχεται (για αυτό ονομάζεται και eavesdrop – κρυφακούει) για να βγάλει συμπεράσματα για την ποιότητα σύνδεσης με το κάθε stationary node. Έτσι, μπορεί να ζητήσει σύνδεση και αποσύνδεση με διαφορετικά nodes ενώ κινείται.

Routing:

Routing είναι η διαδικασία κατά την οποία επιλέγεται μονοπάτι για μετακίνηση πληροφορίας. Η αποδοτικότητα του routing μπορεί να βελτιωθεί από τις εξής πλευρές:

Route Setup

Route maintenance

Service

Για να φτιαχτούν πολλά μονοπάτια προς το sink node, δημιουργούνται πολλά δέντρα, τα οποία έχουν ρίζα τα nodes που είναι 1 hop μακριά από το sink. Κάθε δέντρο απλώνεται σε κάθε δυνατό γείτονα, αλλά αποφεύγει τις συνδέσεις που έχουν χαμηλό QoS. Αυτό επιτρέπει κάθε κόμβο να έχει επιλογή για το μονοπάτι που θα ακολουθήσει η πληροφορία προς αποστολή.

Αφού φτιαχτούν όλα τα μονοπάτια, το κάθε node χρησιμοποιεί αλγόριθμο Sequential Assignment Routing (SAR), ο οποίος έχει ως εξής: Λαμβάνει υπ' όψιν την ενέργεια και το QoS κάθε μονοπατιού, όπως και την προτεραιότητα του πακέτου. Για κάθε πακέτο υπολογίζεται weighted QoS metric, ως το γινόμενο του QoS metric και της προτεραιότητας του πακέτου. Πρακτικά υπολογίζεται η σχέση μεταξύ ποιότητας και προτεραιότητας. Το SAR προσπαθεί να ελαχιστοποιήσει τη μέση τιμή του weighted QoS metric στη διάρκεια ζωής του δικτύου.

2.4 Τεχνολογίες οργάνωσης πληροφορίας για την ολοκλήρωση πολύ σημαντικού αριθμού επιμέρους τοπικών μικροελεγκτών σε ένα σύστημα-ομπρέλα

2.4.1 Γενικά

Όπως ήδη αναφέρθηκε, μπορεί να υπάρξει σημαντική βελτίωση του είδους και της ποιότητας που οι γεωγραφικά διεσπαρμένοι κινητοί υπολογιστικοί σταθμοί παρέχουν, αν συνδεθούν σε έναν (ή περισσότερους) "Σταθμούς Βάσης", δηλαδή σε κεντρικά, ισχυρά υπολογιστικά συστήματα, ο σκοπός των οποίων είναι να αποτελέσουν αποτελεσματική πλατφόρμα για τα ακόλουθα:

1. Προγράμματα διαχείρισης των καθημερινών ή περιοδικών λειτουργιών των κινητών σταθμών, ή των υποδικτύων των κινητών σταθμών.
2. Προγράμματα για την υποστήριξη της λήψης αποφάσεων των σχεδιαστών μεταφορών (transport planners) ή των χειριστών (operators) των επί μέρους μεταφορικών επιχειρήσεων κάθε μεγέθους.
3. Προγράμματα για τον χειρισμό έκτακτων επιχειρήσεων ή έκτακτων συμβάντων

Επί πλέον, σχετικές αναλύσεις απαιτήσεων σε τυπικές εφαρμογές της πράξης έχουν δείξει ότι επιλεγμένα δεδομένα του Σταθμού Βάσης πρέπει να είναι προσβάσιμα σε πολλούς από τους παράγοντες που εμπλέκονται σε επί μέρους επιχειρησιακές διαδικασίες. Ένα παράδειγμα θα βοηθήσει στην κατανόηση της προαναφερθείσας θέσης:

Ας υποθέσουμε ότι μεταξύ των γεωγραφικά διεσπαρμένων κινητών μονάδων είναι εμπορευματοκιβώτια που είναι φορτωμένα σε ένα πλοίο που πλησιάζει σε ένα μεγάλο λιμάνι. Ας υποθέσουμε, επίσης, ότι ένας σημαντικός αριθμός από τα εμπορευματοκιβώτια αυτά πρόκειται να εκφορτωθούν στο λιμάνι αυτό. Ένας αριθμός από χρήστες (πρόσωπα ή φορείς), θα χρειαστεί την εξής πληροφορία από τον Σταθμό Βάσης που είναι υπεύθυνος για τα συγκεκριμένα εμπορευματοκιβώτια, ανάμεσα σε αυτούς οι ακόλουθοι:

i. Οι αρχές του λιμανιού, δεδομένου ότι πρέπει να γνωρίζουν εκ των προτέρων τον τρόπο χειρισμού των προς εκφόρτωση εμπορευματοκιβωτίων: Πόσα και ποια θα χρειαστεί να αποθηκευτούν στον χώρο του λιμανιού μέχρι να γίνει χειρισμός τους, ποια θα χρειαστεί να μεταφορτωθούν σε τραίνα ή σε φορτηγά, ανάγκες ειδικού χειρισμού συγκεκριμένων φορτίων κλπ.

ii. Οι τελωνειακές αρχές, ώστε να προγραμματίσουν την λειτουργία τους κατάλληλα.

iii. Οι χειριστές μεταφορών (transport operators) για να προγραμματίσουν την συνέχεια των επιχειρήσεών τους.

iv. Οι ιδιοκτήτες των μεταφερομένων φορτίων, ώστε να παρακολουθούν της κίνηση και την κατάσταση των φορτίων τους.

v. Οι ασφαλιστικοί φορείς, για όποια φορτία είναι ασφαλισμένα.

Ο Σταθμός Βάσης μπορεί να επικοινωνεί με τους γεωγραφικά διεσπαρμένους κινητούς υπολογιστικούς σταθμούς που ευρίσκονται στις ανάλογες μεταφορικές μονάδες με διάφορους τρόπους, μερικοί των οποίων είναι οι ακόλουθοι:

- ✓ Σύνδεση σε wireless hotspot, όταν και για όποιο χρονικό διάστημα αυτό είναι δυνατό (στις μεταφορές σε σταθμούς, λιμάνια, αεροδρόμια κλπ,

- ✓ GSM/GPRS, στις περιοχές που υπάρχει κάλυψη δικτύου κινητής τηλεφωνίας,
- ✓ Δορυφορική ζεύξη, σε μέρη χωρίς κάλυψη άλλου τηλεπικοινωνιακού δικτύου,

ή άλλον δυνατό τρόπο (αν υπάρχει).

Από την πρώτη μακροσκοπική εξέταση έγινε σαφές ότι ένας Σταθμός Βάσης του τύπου αυτού έχει ανάγκη από ένα ισχυρό υπολογιστικό σύστημα, του οποίου τα δεδομένα θα καταλήξουν να είναι *μεγάλου όγκου*, *πολύπλοκα* και *πολυσχιδή* και, το χειρότερο, *πολύμορφα*. Επιπλέον δε οι τεχνολογικές εξελίξεις και οι μεταβολές που γίνονται σε πραγματικό χρόνο στους φορείς που ασχολούνται με τις περιοχές εφαρμογών που μας ενδιαφέρουν (πχ. μεταφορές), επιβάλουν την χρήση στο σύστημα αυτό αρχιτεκτονικής που επιδέχεται με σχετικά εύκολο τρόπο σημαντικές αλλαγές, θεματικές μεταβολές ή και ολικές στροφές προς άλλες θεματικές κατευθύνσεις.

Επομένως, προτείνεται για το σύστημα του Σταθμού Βάσης μία αρχιτεκτονική προγραμμάτων και δεδομένων που έχει ονομαστεί "Τελεολογική Δομή" από την ομάδα εργασίας του εργαστηρίου "Συστημάτων Πολυθεματικής και Γεωσυσχετισμένης Πληροφορίας" της ΣΗΜΜΥ. Στα επόμενα περιγράφεται ο τρόπος αυτός οργάνωσης της πληροφορίας (περισσότερες λεπτομέρειες στο [1]). Προκειμένου να καλυφθεί η γενικότερη δυνατή περίπτωση, εξετάζεται ο τρόπος οργάνωσης μιας πλατφόρμας πολλαπλών Συστημάτων Υποβοήθησης της Λήψης Αποφάσεων (Decision Support Systems, DSS). Η Τελεολογική Δομή περιγράφεται στα αμέσως επόμενα.

Τονίζεται ότι, λόγω της σημασίας της οργάνωσης του Σταθμού Βάσης με την χρήση της Τελεολογικής Δομής και της ανάγκης ικανοποιητικής πληρότητας της παρούσας εργασίας, η παρουσίαση της Τελεολογικής Δομής που ακολουθεί είναι σχετικά εκτεταμένη. Τα σχετικά παραδείγματα δε είναι από τον τομέα των Μεταφορών Αγαθών και Επιβατών. Δεδομένου, επίσης, ότι θέλουμε ο Σταθμός Βάσης να αποτελέσει αποτελεσματική πλατφόρμα για (πιθανά πολλαπλά) Συστήματα Υποβοήθησης Λήψης Αποφάσεων, πρέπει να τονίσουμε την διαφορετική χρήση των όρων "Δεδομένα" και "Πληροφορία" που εμείς κάνουμε.

2.4.2 Διαφοροποίηση Δεδομένων και Πληροφορίας – Σχετικοί Ορισμοί

(Περισσότερες πληροφορίες στο [73])

Από εδώ και στο εξής, θα κάνουμε την ακόλουθη διάκριση:

- Ως *πληροφορία* θα χαρακτηρίζουμε αυτά τα ειδικά δεδομένα, τα οποία είναι στην κατάλληλη μορφή, ώστε να βοηθήσει το ανθρώπινο μυαλό να εξάγει συγκεκριμένη πληροφορία από τα δεδομένα αυτά (*να πάρει την αντίστοιχη πληροφορία*, όπως λέγεται χαρακτηριστικά). Ας δώσουμε ένα παράδειγμα: Μια βάση γεωσυσχετισμένων δεδομένων, στην οποία χρησιμοποιείται ένα μοντέλο για να εκτιμήσει την κυκλοφοριακή συμφόρηση στους δρόμους μια περιοχής, και η οποία παράγει αντίστοιχη απεικόνιση των δρόμων αυτών, χρωματισμένων με μία διαβάθμιση χρώματος από το πράσινο προς το κόκκινο, όπου το χρώμα υποδηλώνει τον βαθμό κυκλοφοριακής συμφόρησης στον κάθε δρόμο (αμιγές πράσινο: μη συμφορημένος δρόμος, ενδιάμεσο χρώμα: κατά ένα μέρος συμφορημένος δρόμος, αμιγές κόκκινο: ολικά συμφορημένος δρόμος), λέμε ότι *παρέχει πληροφορία για το θέμα της συμφόρησης*.
- Ως *δεδομένα*, πλέον, θα χαρακτηρίζουμε αυτά τα δεδομένα, τα οποία χρειάζεται να υποστούν κατάλληλη επεξεργασία (με χρήση μαθηματικών υπολογισμών ή/και αλγοριθμικών διαδικασιών), ώστε να εξαχθεί από αυτά επιθυμητή πληροφορία. Στο

προηγούμενο παράδειγμα, τα δεδομένα της βάσης, τα οποία χρησιμοποιεί το υπολογιστικό μοντέλο για να παράξει την απεικόνιση των δρόμων με τον χρωματικό κώδικα συμφόρησης, λέμε ότι είναι απλά δεδομένα.

Για να αποφευχθεί πιθανή σύγχυση, θα χρησιμοποιηθεί η ακόλουθη γενική ορολογία σε σχέση με την αρχιτεκτονική δομή που θα περιγραφεί στα επόμενα:

- Ως *Υπολογισμένα Δεδομένα* (computed data) ή *Υπολογισμένη Πληροφορία* ή *Ενδείκτες* (indicators) θα χαρακτηρισθούν αυτά τα δεδομένα που είναι αναγκαία ή υποβοηθούν την λήψη αποφάσεων από τους ειδικούς στον εκάστοτε τομέα εφαρμογών. Τα δεδομένα αυτά υπολογίζονται με μαθηματικό ή αλγοριθμικό τρόπο.
- Ως *Δεδομένα Εισόδου* θα χαρακτηρισθούν αυτά τα δεδομένα που είναι αναγκαία για να υπολογιστούν (με μαθηματικό ή αλγοριθμικό τρόπο) οι *ενδείκτες* (ή *υπολογισμένα δεδομένα*).
- Ως *Δεδομένα Πηγών* θα χαρακτηρισθούν τα δεδομένα που εξωτερικές πηγές παρέχουν στον Σταθμό Βάσης και τα οποία, μετά από διαδικασία μετατροπής, ελέγχου ποιότητας και εναρμόνισης, τελικά ολοκληρώνονται στα *δεδομένα εισόδου* του Σταθμού Βάσης.

Ακόμη, θα θεωρηθεί ότι, *τόσο τα δεδομένα εισόδου, όσο και τα υπολογισμένα δεδομένα* (οι ενδείκτες) *μπορούν να αποτελούν μέρη του πυρήνα δεδομένων* του συστήματος. *Είναι σώφρον, επίσης, τα αρχικά σύνολα δεδομένων που λαμβάνονται κάθε φορά από τις πηγές να αποθηκεύονται και αυτά στον πυρήνα δεδομένων* για τους εξής λόγους:

- *Καταλογισμού ευθυνών,* σε περίπτωση λαθών, προβλημάτων, χαμηλής ποιότητας δεδομένων κλπ. και
- Για να επαναληφθεί μέρος ή το σύνολο των *διαδικασιών* μετατροπής, ελέγχου ποιότητας, εναρμόνισης και ολοκλήρωσης στον πυρήνα δεδομένων, αν αυτό καταστεί αναγκαίο (κάτι που όντως συμβαίνει ενίοτε στην πράξη).

2.4.3 Το νόημα και η χρήση του *Χάρτη Περιεχομένων* ή *Τελεολογικής Δομής* του Συστήματος

Στο προηγούμενο κεφάλαιο εδείχθη ότι, προκειμένου να υποστηριχθούν διαδικασίες του Σταθμού Βάσης και οι σχετικές προγραμματισμένες ή έκτακτες επιχειρησιακές διαδικασίες, πρέπει να αναπτυχθούν, ει δυνατόν εκ των προτέρων, κατάλληλοι πυρήνες (κυρίως) γεωσυσχετισμένης πληροφορίας, οι οποίοι, όμως, *αναμένεται* να εμφανίσουν (όλα ή κάποια από) τα εξής χαρακτηριστικά:

- *Πολυθεματικότητα*,
- πολύ μεγάλο *όγκο*,
- ιδιαίτερα αυξημένη *εσωτερική πολυπλοκότητα* και
- *πολυμορφία*.

Επί πλέον δε, είναι κρίσιμο τα δεδομένα και η πληροφορία αυτών των πυρήνων δεδομένων να καταστούν

- *εσωτερικά συμβατά* και
- *εσωτερικά διαλειτουργικά*.

Η δε πληροφορία που τελικά συμπεριλαμβάνεται σε ένα Πληροφοριακό Σύστημα (ΠΣ) του τύπου αυτού (πχ. σαν αυτό του Σταθμού Βάσης) μπορεί να προέρχεται (στο σύνολό της ή

κατά ένα μέρος) *από ένα σημαντικό αριθμό από εξωτερικές, αυτόνομες και ετερογενείς πηγές.* Στις περιπτώσεις αυτές συνήθως χρειάζονται ειδικές διαδικασίες

- *μετατροπής* της πληροφορίας στην επιθυμητή μορφή,
- *ελέγχου ποιότητας και πληρότητας* της πληροφορίας,
- *εναρμόνισης* της πληροφορίας και
- *ολοκλήρωσης* της πληροφορίας στο υπό ανάπτυξη σύστημα.

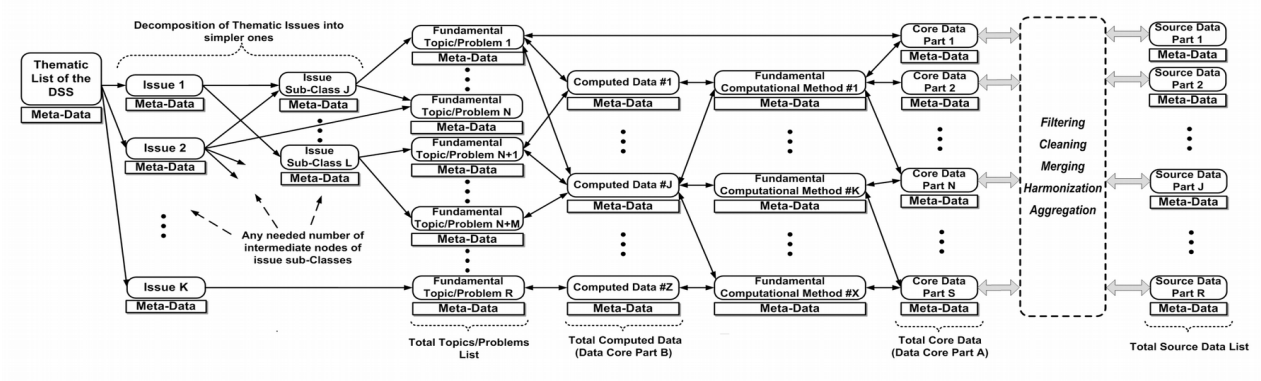
Τα χαρακτηριστικά αυτά της πληροφορίας, καθώς και οι προαναφερθείσες διαδικασίες, μπορεί να δημιουργήσουν πραγματικό δομικό χάος και να οδηγήσουν σε κατάρρευση τα αντίστοιχα Πληροφοριακά Συστήματα. Η προσέγγιση που επιλέγεται από τα μέλη του Εργαστηρίου Συστημάτων Πολυθεματικής και Γεωσυσχετισμένης Πληροφορίας είναι η ακόλουθη:

> Δεδομένου ότι αναμένεται να δημιουργηθεί χάος αν δεν ληφθεί ειδική μέριμνα για την αντιμετώπιση των προαναφερθέντων προβλημάτων και των συνεπειών τους, *προτιμάται η εκ των προτέρων πρόβλεψη και η κατάλληλη οργάνωση της πληροφορίας και του λογισμικού των Πληροφοριακών Συστημάτων, ώστε αυτά να παραμείνουν οργανωμένα και ελέγξιμα.*

Αυτό επιτυγχάνεται με την χρήση μιας καινοτομικής οργάνωσης των Πληροφοριακών Συστημάτων, η οποία στηρίζεται στην εκτεταμένη χρήση μιας δομής μεταδεδομένων και την οποία τα μέλη του Εργαστηρίου καλούν *Χάρτη Περιεχομένων* ενός Συστήματος, ή, εναλλακτικά, *Τελεολογική Δομή* του Συστήματος αυτού. Η δομή αυτή θα περιγραφεί στα επόμενα.

2.4.4 Μορφή του Χάρτη Περιεχομένων (ή Τελεολογικής Δομής) του συστήματος

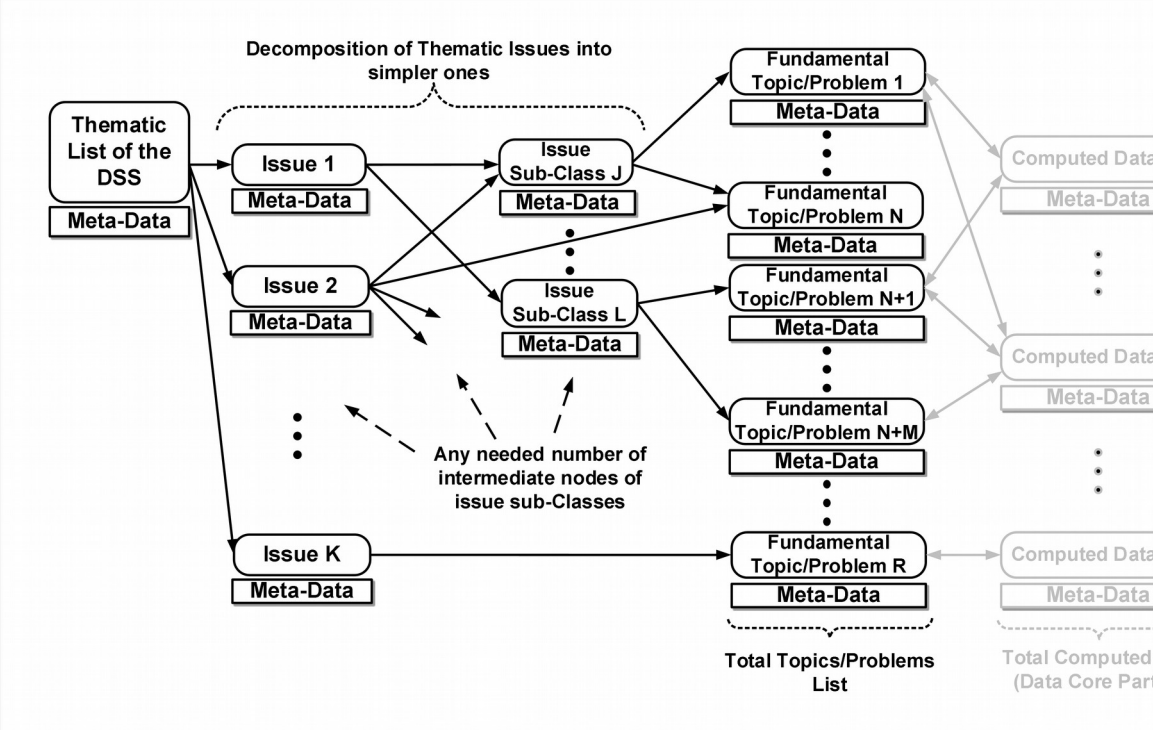
Ο *Χάρτης Περιεχομένων* (όπου η χρήση του όρου *χάρτης* είναι *μεταφορική* και όχι κυριολεκτική) ή *Τελεολογική Δομή* (από την αρχαϊκή σημασία της λέξης *τέλος* που σημαίνει *σκοπός*) του συστήματος είναι μια ειδική δομή μεταδεδομένων, που περιγράφεται στην υποενότητα αυτή και απεικονίζεται *συνολικά* (αλλά χονδρικά) στο επόμενο Σχήμα:



Σχήμα: Ο Χάρτης Περιεχομένων ή Τελεολογική Δομή του συστήματος

Η δομή του Σχήματος «κόπηκε» σε τρία τμήματα, προκειμένου να καταστεί ευκολότερα αναγνώσιμη από τον αναγνώστη. Ακολουθεί η παρουσίαση κάθε ενός των τμημάτων αυτών.

Το πρώτο τμήμα του Χάρτη Περιεχομένων ή Τελεολογικής Δομής είναι αυτό που αφορά την θεματική αποσύνθεση του συστήματος. Το τμήμα αυτό φαίνεται στο επόμενο Σχήμα.



Σχήμα: Θεματική αποσύνθεση του συστήματος

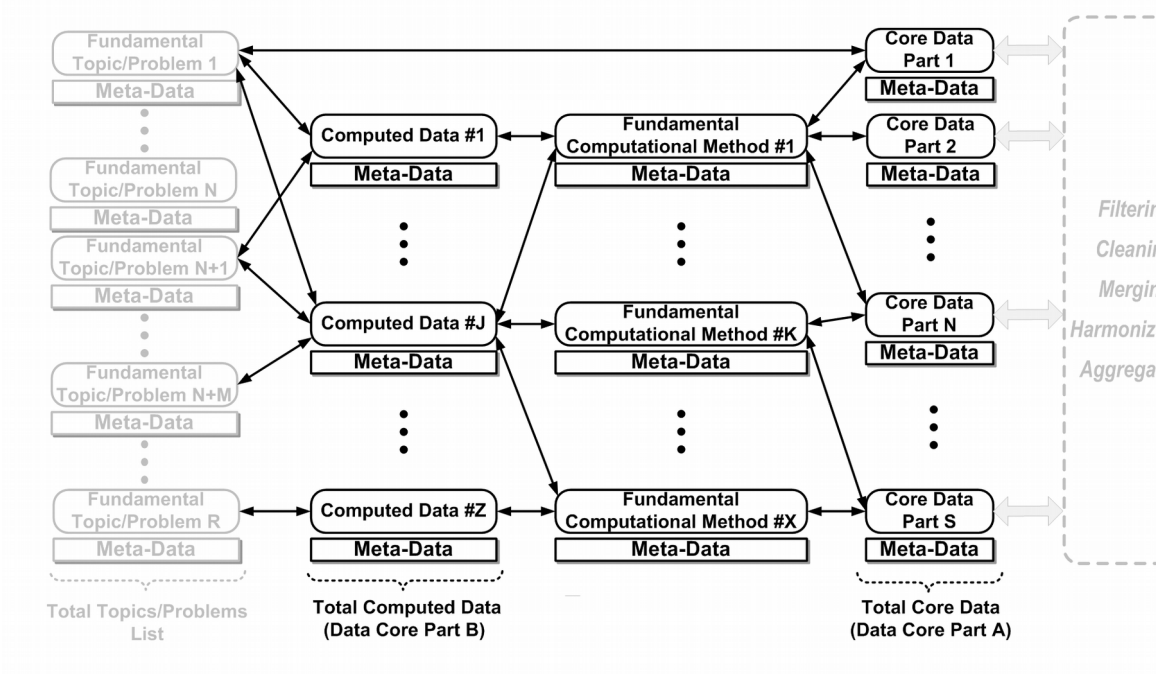
Με τον όρο θεματική αποσύνθεση εννοείται η σταδιακή θεματική ανάλυση όλων των θεματικών περιοχών ή γνωστικών αντικειμένων ή περιοχών ενδιαφέροντος, για τις οποίες το σύστημα πρέπει να προσφέρει πληροφορία ή να παρέχει λύσεις, κατευθύνσεις κλπ. Αρχή γίνεται από τον αρχικό Θεματικό Κατάλογο του Συστήματος στα αριστερά. Η θεματική αποσύνθεση προχωρεί από τα αριστερά προς τα δεξιά, με ανάλυση κάθε θέματος σε υποθέματα, και αυτών σε περαιτέρω υποθέματα, κλπ., μέσω μιας δομής που μοιάζει με δένδρο (αλλά στην πραγματικότητα είναι ακυκλικός γράφος), έως ότου φθάσει κανείς σε θεμελιώδη (αλλά όχι απαραίτητα απλά) υποθέματα ή προβλήματα. *Θεμελιώδη υποθέματα/προβλήματα είναι αυτά τα οποία δεν αποσυντίθενται περαιτέρω θεματικά*, αλλά κάθε ένα από αυτά αποτελεί ένα καλώς καθορισμένο υποθέμα ή πρόβλημα, *όπου είναι γνωστό το τι ζητείται ακριβώς*, δηλαδή το ακριβές είδος της πληροφορίας (υπολογισμένων δεδομένων ή ενδεικτών) που το σύστημα πρέπει να παράσχει στους χρήστες του, ή με χρήση της οποίας το σύστημα μπορεί να προβεί σε περαιτέρω, πλήρως καθορισμένες ενέργειες. Ας δανειστούμε ένα παράδειγμα από την περιοχή των μεταφορών και συγκεκριμένα από την προσπάθεια για την κατασκευή του ETIS (*E*uropean *T*ransport Policy *I*nformation *S*ystem), ενός Συστήματος Υποβοήθησης Λήψης Αποφάσεων και χάραξης πολιτικής για τις Μεταφορές και τις Συγκοινωνίες σε Ευρωπαϊκό επίπεδο, στο οποίο αντιμετωπίζονται αντίστοιχα προβλήματα, καθώς το ETIS είναι ιδιαίτερα πολυθεματικό, με πολύ μεγάλο, γεωσυσχετισμένο πυρήνα δεδομένων και πληροφορίας και πολλές εκατοντάδες εξωτερικών πηγών δεδομένων. Ένα μονοπάτι του ETIS από την ρίζα της δενδροειδούς δομής (τον θεματικό κατάλογο του συστήματος), μέχρι ένα καταληκτικό φύλλο της δομής αυτής (θεμελιώδες θέμα/πρόβλημα) μπορεί να είναι αυτό που περιγράφεται στη συνέχεια. Στην περίπτωση του παραδείγματος, προχωρούμε λίγο ακόμη στην Τελεολογική Δομή, δείχνοντας τους ενδείκτες ή την υπολογισμένη πληροφορία που είναι αναγκαία για να ληφθούν αποφάσεις σχετικές με το θέμα του παραδείγματος:

Θέματα Πολιτικής Μεταφορών σε Ευρωπαϊκό επίπεδο (επί μέρους θέμα του θεματικού καταλόγου) -> Επιπτώσεις των μεταφορών στο Περιβάλλον (μεταξύ άλλων υποθεμάτων) -> Θόρυβος (μεταξύ άλλων επιπτώσεων, όπως οι ρύποι κλπ.) -> Θόρυβος στα Κυκλοφοριακά Δίκτυα (μεταξύ άλλων υποθεμάτων, όπως ο θόρυβος σε περιοχές που έχουν αεροδρόμια, κλπ.) -> (θεμελιώδες πρόβλημα ή υποθέμα:) Εκτίμηση του θορύβου που οφείλεται στην κυκλοφορία αυτοκινήτων στα Ευρωπαϊκά δίκτυα αυτοκινητοδρόμων για μια καθορισμένη γεωγραφική περιοχή (ανάμεσα σε άλλα θεμελιώδη προβλήματα όπως είναι ο θόρυβος λόγω αυτοκινήτων στις πόλεις) -> (Ενδείκτης ή υπολογισμένη πληροφορία, πλήρως καθορισμένη:) Εκτιμήσεις θορύβου σε όλους τους οδικούς συνδέσμους και τους κόμβους των Ευρωπαϊκών αυτοκινητοδρόμων μιας καθορισμένης γεωγραφικής περιοχής.

Από το παράδειγμα φαίνεται ότι, όταν φθάσουμε σε ένα θεμελιώδες πρόβλημα ή θέμα, το τι ζητείται είναι καθορισμένο ακριβώς.

Αξίζει να αναφερθεί ότι ο ενδείκτης *«Μετρήσεις ή εκτιμήσεις θορύβου σε όλους τους οδικούς συνδέσμους και τους κόμβους των Ευρωπαϊκών αυτοκινητοδρόμων μιας καθορισμένης γεωγραφικής περιοχής»* δεν είναι καθόλου απλός. Στην πραγματικότητα χρειάζεται κανείς μια υπό-βάση γεωσυσχετισμένων δεδομένων (μικρή ή μεγαλύτερη ανάλογα με την έκταση της περιοχής ενδιαφέροντος) για να τηρήσει τα υπολογισμένα αυτά δεδομένα και ένα GIS για τη σχετική οπτικοποίηση (visualization) της πληροφορίας. Εάν αυξήσουμε την απαιτούμενη διακριτική ικανότητα (level of detail, πχ. αν συμπεριλάβουμε και δρόμους χαμηλότερης κατηγορίας, όπως τοπικούς δρόμους), ή την έκταση της περιοχής ενδιαφέροντος, μπορεί να αυξήσουμε δραματικά το μέγεθος της πληροφορίας ενδιαφέροντος και της παραγόμενης βάσης δεδομένων των αποτελεσμάτων.

Στη συνέχεια της Τελεολογικής Δομής ή Χάρτη Περιεχομένων του συστήματος, είναι περαιτέρω δομές που φαίνονται στο επόμενο σχήμα:



Σχήμα: Μεταδεδομένα για τον υπολογισμό των *Ενδεικτών* ή *Υπολογισμένων Δεδομένων* από τα *Δεδομένα Εισόδου* με χρήση των *Θεμελιωδών Υπολογιστικών Μεθόδων*

Αυτό που περιγράφεται στην υποδομή αυτή (το δεύτερο τμήμα της συνολικής δομής) είναι, στην πραγματικότητα, οι αναγκαίοι *ενδείκτες* ή *υπολογισμένη πληροφορία,* όπως αυτή προκύπτει από τα *θεμελιώδη προβλήματα ή θέματα*, ο τόπος υπολογισμού τους (με την χρήση των *θεμελιωδών υπολογιστικών μεθόδων*) και τα *δεδομένα εισόδου* που χρειάζονται για τον υπολογισμό αυτόν.

Τα κουτιά του Σχήματος που ονομάζονται *Θεμελιώδεις Υπολογιστικές Μέθοδοι* υποδηλώνουν οποιουδήποτε τύπου μαθηματικές ή αλγοριθμικές μεθόδους υπολογισμού (ή και μαθηματικά ή αλγοριθμικά μοντέλα, δηλαδή *εκτιμήσεις συμπεριφοράς* επί μέρους πραγματικών ή ιδεατών συστημάτων, στην περίπτωση που η λειτουργία των συστημάτων αυτών είναι υπερβολικά πολύπλοκη ή άγνωστη). Οι θεμελιώδεις μέθοδοι υπολογισμού μπορεί να είναι

- απλές ή πολύπλοκες,
- να περιλαμβάνουν εναλλακτικούς τρόπους υπολογισμού ποσοτήτων,
- να αναλύονται σε περισσότερες ενδιάμεσες υπολογιστικές μεθόδους, διασυνδεδεμένες με οποιονδήποτε πιθανό τρόπο,
- ή, τέλος, να είναι αναδρομικές.

Τόσο *τα δεδομένα εισόδου*, όσο και *τα υπολογισμένα δεδομένα ή ενδείκτες αποτελούν* (συνήθως) *μέρη του πυρήνα δεδομένων και πληροφορίας του συστήματος.* Πολλές φορές δε, απλά δεδομένα εισόδου είναι ταυτόχρονα και ενδείκτες. Στο προηγούμενο παράδειγμα για τον θόρυβο στους Ευρωπαϊκούς αυτοκινητοδρόμους, εάν έχουμε διαθέσιμες μετρήσεις θορύβου στους αυτοκινητοδρόμους για κάποιες παρελθούσες χρονιές (πχ. το 2012), οι μετρήσεις αυτές είναι δεδομένα εισόδου, τα οποία είναι ταυτόχρονα και ενδείκτες. Εάν όμως χρειάζεται να κάνουμε εκτιμήσεις για τον θόρυβο στους ίδιους αυτοκινητοδρόμους σε μια μελλοντική ημερομηνία (πχ. στο 2030, οπότε η διαδικασία ονομάζεται *πρόβλεψη* ή forecasting), τότε χρειαζόμαστε τελείως διαφορετικού τύπου δεδομένα εισόδου, πολύ πιο πολύπλοκα και εκτεταμένα, καθώς και ιδιαίτερα σύνθετη θεμελιώδη υπολογιστική μέθοδο που θα μας δώσει *εκτιμήσεις* (και όχι μετρήσεις) της κίνησης στους αντίστοιχους δρόμους το 2030. Όμως, το τι ακριβώς χρειαζόμαστε κάθε φορά είναι σαφώς καθορισμένο όταν η θεματική αποσύνθεση έχει φθάσει στα θεμελιώδη προβλήματα ή θέματα, καθώς το θεμελιώδες θέμα «Μετρήσεις θορύβου στους Ευρωπαϊκούς αυτοκινητοδρόμους το 2012» και το θεμελιώδες πρόβλημα «Υπολογισμός εκτιμήσεων θορύβου στους Ευρωπαϊκούς αυτοκινητοδρόμους το 2030» είναι τελείως διαφορετικά και ως τέτοια πρέπει να τα χειριστεί το σύστημα.

Στην πράξη, *προκειμένου να πετύχουμε αποτελεσματική αντιμετώπιση της εσωτερικής πολυπλοκότητας και πολυμορφίας των δεδομένων*, πρέπει οπωσδήποτε να φροντίσουμε, *ώστε η θεματική αποσύνθεση του συστήματος να φθάνει σε ενδείκτες ή υπολογισμένη πληροφορία που είναι αναμφίβολα και πλήρως ορισμένοι*. Ακόμη και ελαφρά διαφέροντες ενδείκτες πρέπει να αντιμετωπίζονται ως διαφορετικές οντότητες. Αυτό σημαίνει ότι, για κάθε υποσύνολο επιθυμητών ενδεικτών, η αποσύνθεση πρέπει να καταλήξει σε ένα αντίστοιχο υποσύνολο υπολογιστικών μεθόδων του τύπου

$$\underline{\boldsymbol{i}} = f(\underline{\boldsymbol{d}})$$

όπου $\underline{\boldsymbol{i}}$ είναι το διάνυσμα των συγκεκριμένων ενδεικτών, f είναι η μέθοδος για τον υπολογισμό των ενδεικτών και $\underline{\boldsymbol{d}}$ είναι το διάνυσμα των δεδομένων εισόδου, που είναι αναγκαία για τον υπολογισμό των συγκεκριμένων ενδεικτών. *Από το σημείο αυτό και πέρα*, λόγω του πλήρους και ακριβούς προσδιορισμού των ενδεικτών, της σχετικής υπολογιστικής

μεθόδου και των σχετικών δεδομένων, *κάθε ίχνος πολυμορφίας ή εσωτερικής διαφοροποίησης των δεδομένων έχει αρθεί*.

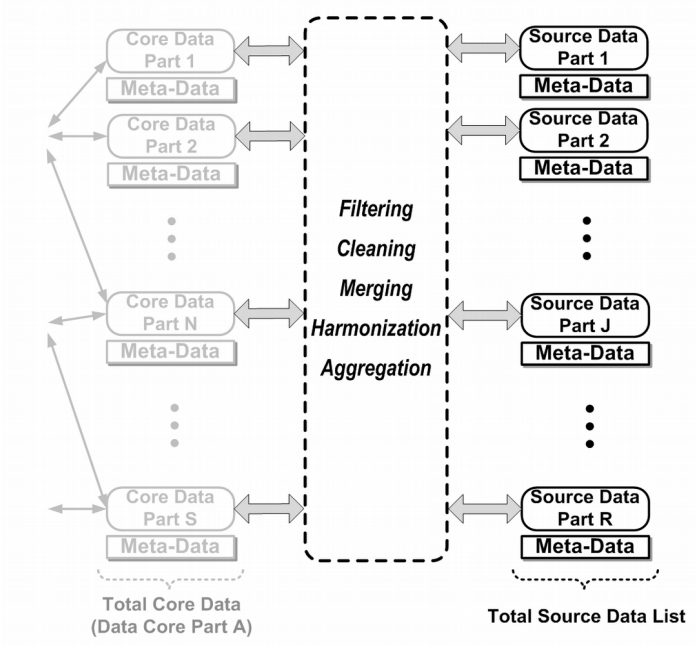
Το τμήμα αυτό του Χάρτη Περιεχομένων ή Τελεολογικής Δομής του συστήματος δίνει, επομένως, ανά πάσα στιγμή:

1. Τον συνολικό κατάλογο των *ενδεικτών* ή *υπολογισμένης πληροφορίας* που περιλαμβάνεται στο σύστημα.
2. Τον συνολικό κατάλογο των υπολογιστικών μεθόδων για του υπολογισμό των ποσοτήτων του σημείου (1) και
3. Τον συνολικό κατάλογο των δεδομένων εισόδου που είναι αναγκαία για του υπολογισμό των ποσοτήτων του σημείου (1).

Στους καταλόγους αυτούς δε, προστίθενται οποιεσδήποτε επί πλέον πληροφορίες είναι αναγκαίες για τον ορθό χειρισμό και την οπτικοποίηση της πληροφορίας ή/και των δεδομένων των σημείων (1), (2) και (3).

Η τήρηση του αυτού του *συνολικού σύνθετου καταλόγου* καθιστά το σύστημα νομοτελειακά *επικαιροποιήσιμο* (updatable) *και αναβαθμίσιμο* (upgradable) και, γενικά, *εξελίξιμο* και, επομένως, *βιώσιμο* (sustainable).

Το τελευταίο τμήμα του Χάρτη Περιεχομένων ή Τελεολογικής Δομής του συστήματος, το οποίο φαίνεται στο επόμενο Σχήμα, συσχετίζει κάθε υποομάδα δεδομένων εισόδου του προηγούμενου σημείου (3) με τις πηγές από τις οποίες αποκτήθηκε (και, ενδεχομένως, από τις οποίες επικαιροποιείται).



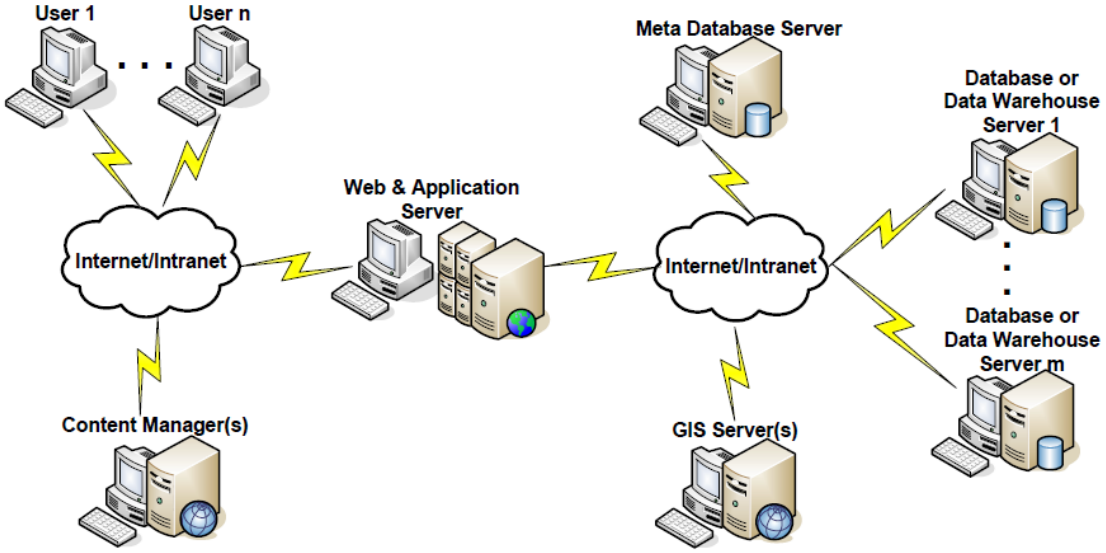
Σχήμα: Μεταδεδομένα για του τρόπο απόκτησης των δεδομένων εισόδου από τις εξωτερικές πηγές

Σε όλους τους κόμβους της προηγουμένως περιγραφείσας δομής μπορεί και πρέπει να συμπεριληφθούν κατάλληλα επί πλέον μεταδεδομένα [73].

Περαιτέρω περιγραφή των επί μέρους υποδομών του Χάρτη Μεταδεδομένων ή Τελεολογικής Δομής του συστήματος είναι πέραν του σκοπού της παρούσας εργασίας και για τον λόγο αυτόν παραλείπεται.

2.4.5 Το συνολικό Πληροφοριακό Σύστημα του Σταθμού Βάσης

Στο σχήμα που ακολουθεί φαίνεται μία πιθανή υλοποίηση ενός Σταθμού Βάσης. Η συνιστώσες αυτού του Σταθμού Βάσης περιγράφονται σύντομα στα επόμενα.



Το Πληροφοριακό Σύστημα του Σταθμού Βάσης πρέπει να περιλαμβάνει:

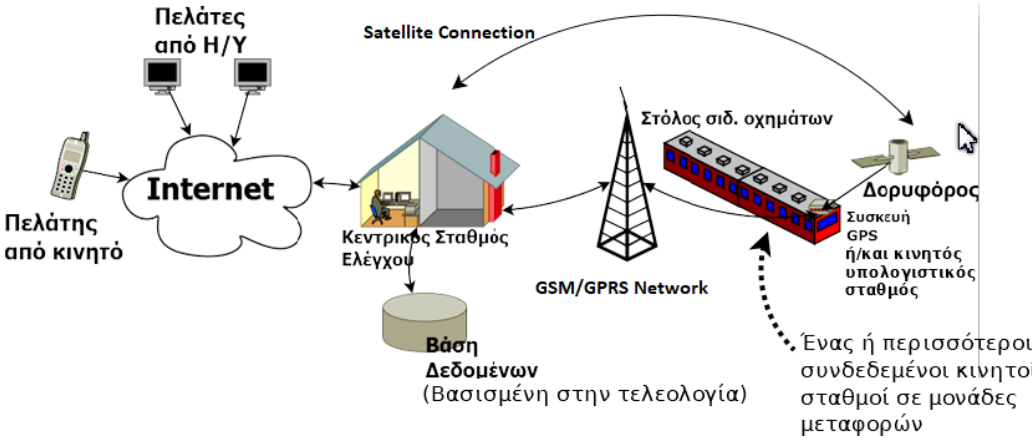
- Έναν εξυπηρετητή εφαρμογών (Application Server), στον οποίο θα εκτελούνται τα αναγκαία προγράμματα του Σταθμού Βάσης. Ακόμη, έναν εξυπηρετητή χρηστών μέσω του διαδικτύου (Web Server), ο οποίος επιτρέπει διαδικτυακή πρόσβαση στους αδειοδοτημένους χρήστες του συστήματος. Στο προηγούμενο σχήμα, ο Application Server και ο Web Server φαίνεται να είναι μέρη ενός Πληροφοριακού Συστήματος, κάτι που δεν είναι απαραίτητο, αλλά θα προσδιοριστεί από τις ανάγκες του Σταθμού Βάσης.
- Έναν Meta-Database Server, όπως έχει ονομαστεί ο εξυπηρετητής στον οποίον βρίσκεται η βάση μεταδεδομένων, μέσω της οποίας υλοποιείται ο Χάρτης Περιεχομένων του Συστήματος ή Τελεολογική Δομή.
- Μία ή περισσότερες Βάσεις Δεδομένων [ή Αποθήκες Δεδομένων (Data Warehouses) ή Αποθετήρια Δεδομένων και Πληροφορίας (Data and Information Observatories)], στις οποίες αποθηκεύονται όλοι οι Ενδείκτες και τα σχετικά δεδομένα του συστήματος.
- Ένα Σύστημα Χειρισμού Γεωγραφικής Πληροφορίας (GIS), το οποίο μπορεί να χρησιμοποιείται για την οργάνωση των δεδομένων και της πληροφορίας σε χάρτες (δεδομένου ότι συντριπτικό ποσοστό των δεδομένων και της πληροφορίας, όπως η πράξη έχει καταδείξει, είναι γεωσυσχετισμένο).
- Ένα ή περισσότερα συστήματα διαχειριστών περιεχομένου (Content Managers), τα οποία θα χειρίζονται ή ειδικοί ή ομάδες ειδικών (experts ή groups of experts) που θα διαχειρίζονται το περιεχόμενο του Πληροφοριακού Συστήματος του Σταθμού Βάσης. Δεδομένης της πολυπλοκότητας και όγκου προγραμμάτων και δεδομένων

του Σταθμού Βάσης, και δεδομένης της χρήσης της Τελεολογική Δομής για την οργάνωση του Σταθμού Βάσης (όπως προτείνεται από την ομάδα του Εργαστηρίου Συστημάτων Πολυθεματικής και Γεωσυσχετισμένης Πληροφορίας), υπάρχει σαφής ανάγκη διαχείρισης του περιεχομένου προγραμμάτων και δεδομένων του συνολικού Πληροφοριακού Συστήματος από ειδικούς στα θέματα και τα προβλήματα που σχετίζονται με κάθε θεματική περιοχή που το σύστημα καλύπτει.

Ανάλογα με το φορτίο του Σταθμού Βάσης, Πληροφοριακά Συστήματα, τα οποία φαίνονται στο σχήμα ως διακριτά, μπορούν να ενοποιηθούν σε ένα σύστημα εάν μία τέτοια λύση είναι επαρκής. Όμως, εάν ο φόρτος του Σταθμού Βάσης είναι μεγάλος, μπορεί να γίνει και το ακριβώς αντίθετο, δηλαδή διακριτά Πληροφοριακά Συστήματα του σχήματος να αντικατασταθούν από σχετικές ομάδες (clusters) Πληροφοριακών Συστημάτων.

2.4.6 Πιθανή δομή του συνολικού συστήματος με τους Σταθμούς Βάσης και κινητούς υπολογιστικούς σταθμούς σε μονάδες μεταφοράς

Στο σχήμα που ακολουθεί [74] φαίνεται μία πιθανή οργάνωση του συνολικού συστήματος που προτείνεται από την ομάδα του Εργαστηρίου Συστημάτων Πολυθεματικής και Γεωσυσχετισμένης Πληροφορίας. Το σύστημα αυτό περιλαμβάνει τον Σταθμό Βάσης καθώς και κινητούς υπολογιστικούς σταθμούς που είναι τοποθετημένοι σε μονάδες μεταφορών. Παρ' όλον ότι στο σχήμα εμφανίζονται ως τρόποι επικοινωνίας του Σταθμού Βάσης με τους κινητούς σταθμούς, οι δορυφορικές ζεύξεις ή οι ζεύξεις μέσω του δικτύου GSM/GPRS, μπορεί να φανταστεί κανείς οποιοδήποτε άλλο δυνατό τρόπο επικοινωνίας, όπως πχ. WiFi hotspots σε αεροδρόμια, λιμάνια κλπ. (που είναι κατάλληλα για προσωρινές μόνο συνδέσεις) κλπ.



Σχήμα: Πιθανή δομή του συνολικού συστήματος με τους Σταθμούς Βάσης και τους κινητούς σταθμούς

3 Το συνολικό πρόβλημα

Στη συνέχεια εξηγούνται οι περιοχές οι οποίες προτείνονται για βελτίωση στη λειτουργία και δομή.

Οι τρεις τομείς μέσω των οποίων μπορεί να αυξηθεί η ευφυΐα των τοπικών δικτύων ασύρματης επικοινωνίας είναι οι εξής:

- Αυτοοργάνωση τοπικών δικτύων

- Ισχυρότερα μηχανήματα

- Επικοινωνία με Σταθμό Βάσης

Αυτοί εξηγούνται συνοπτικά στα επόμενα.

- Στα τοπικά δίκτυα μεταφορών (πχ. τα containers σε ένα τραίνο) έχει παρατηρηθεί πολύ μικρή ανάπτυξη ως προς την ευφυΐα τους, ειδικά σε σχέση με τα υπόλοιπα συστήματα δικτύων (πχ. υπολογιστών). Ταυτόχρονα, έχουν αυξηθεί οι απαιτήσεις από αυτά και η πολυπλοκότητά τους, αφού γίνονται συνεχώς περισσότερες μεταφορές παγκοσμίως και αυξάνονται οι ανάγκες βελτίωσης των παρεχομένων υπηρεσιών. Σε αυτές τις κατηγορίες δικτύων υπάρχει μικρή ευφυΐα και χρειάζεται να παρεμβαίνει ο άνθρωπος σε πολλά σημεία για να γίνει η επιθυμητή δομή. Χρειάζεται να βελτιωθούν αυτές οι διαδικασίες, και όσον αφορά το δίκτυο ως σύνολο, αλλά και κάθε κόμβος ξεχωριστά να μπορεί να εκτελεί περισσότερες διεργασίες μόνος του, επικοινωνώντας, ταυτόχρονα, με το υπόλοιπο δίκτυο. Είναι σημαντικό όχι μόνο να μπορούν να λειτουργούν οι τοπικοί αυτοί κόμβοι και το δίκτυο που δημιουργούν μεταξύ τους σε σταθερή κατάσταση, αλλά και όταν υπάρξουν αναταραχές, όπως για παράδειγμα αν ένας κόμβος του δικτύου κλαπεί ή πάθει βλάβη, ή αν ολόκληρο μέρος του δικτύου φύγει από την προκαθορισμένη θέση του, χωρίς να υπάρξει κάποια ενημέρωση για το γεγονός αυτό. Είναι χρήσιμο, λοιπόν, να γίνει προσπάθεια στη βελτίωση των υπολογιστικών συσκευών που τοποθετούνται στα μέσα μεταφοράς αγαθών (πχ. βαγόνια τρένου, φορτηγά αυτοκίνητα κλπ.) και των δικτύων που δημιουργούνται με τη χρήση των συσκευών αυτών, ώστε να είναι πιο εύκολη η κάθε σχετική διεργασία.

  Δύο τομείς από τους οποίους πάρθηκαν τρόποι αυτοοργάνωσης ήταν τα δίκτυα υπολογιστών και τα ασύρματα δίκτυα αισθητήρων. Αρχικά, τα δίκτυα υπολογιστών έχουν αναπτυχθεί σε μεγάλο βαθμό, λόγω της καθοριστικής χρήσης τους όσον αφορά την ασύρματη επικοινωνία μεταξύ ανθρώπων και στη μεταφορά δεδομένων. Τα ασύρματα δίκτυα αισθητήρων χρησιμοποιούνται σε πολλές εφαρμογές, και ειδικά σε στρατιωτικές. Σε αυτές, υπάρχει απαίτηση για βέλτιστη επίδοση, οπότε κανείς μπορεί να βρει χρήσιμες δομές που μπορούν να εφαρμοστούν στα τοπικά δίκτυα μεταφοράς προϊόντων.

- Άλλη κατηγορία η οποία θα μπορούσε να βοηθήσει στην αύξηση της ευφυΐας είναι η βελτίωση των μηχανημάτων (hardware). Αποτελεί ίσως την πιο γρήγορη και εύκολη λύση, και την πηγή για την ύπαρξη της αυτοοργάνωσης και άλλων λειτουργιών. Θα μπορούσε να θεωρηθεί ως μειονέκτημα αυτής της τακτικής το κόστος, ότι, δηλαδή, θα καθιστούν με τη χρήση τους οι δομές δικτύων πιο ακριβές, γεγονός το οποίο μπορεί να είχε ως αποτέλεσμα να μην έχει νόημα η προσπάθεια που γίνεται για τη βελτίωση της ευφυΐας μέσω ισχυρότερων μηχανημάτων. Όμως, το κόστος για τέτοια μηχανήματα, ειδικά σε σχέση με το κόστος των βαγονιών ή των φορτηγών μέσα στα οποία θα τοποθετούνται, είναι μηδαμινό.

- Τέλος, βελτίωση στην ευφυΐα μπορεί να επιτευχθεί μέσα από την ασύρματη επικοινωνία των κόμβων του αυτοοργανούμενου τοπικού δικτύου με Σταθμό Βάσης. Αυτή η επικοινωνία θα προσφέρει λύση σε πολλά προβλήματα λήψης αποφάσεων, τα οποία δημιουργούνται κατά τη διαδικασία μεταφοράς, όπως βλάβες, κλοπές κλπ, αφού θα είναι δυνατή η ανταλλαγή χρήσιμης πληροφορίας για την επίλυση των σχετικών προβλημάτων που εμφανίζονται.

Έγινε αρχικά προσπάθεια αναζήτησης μηχανημάτων ικανών να προσφέρουν λύσεις στις νέες αυτές ανάγκες. Στη συνέχεια αναφέρθηκαν σημαντικές εργασίες στους σχετικούς τομείς αυτοοργανούμενων δικτύων με ασύρματη επικοινωνία μεταξύ των κόμβων των δικτύων αυτών. Τέλος, προσφέρθηκε ολοκληρωμένη λύση, σύμφωνα με την Τελεολογική Δομή, ενός Πληροφοριακού Συστήματος, το οποίο πρόκειται να χρησιμοποιεί ο Σταθμός Βάσης για να υλοποιεί το Σύστημα Υποστήριξης Λήψης Αποφάσεων. Η δομή αυτή δεν αναλύθηκε περαιτέρω στα πλαίσια της εργασίας, αφού υπερβαίνει τα όριά της. Στα επόμενα παρουσιάζεται μία προσέγγιση του προβλήματος της βελτίωσης της ευφυΐας που αναφέρθηκε στα προηγούμενα, όπου αρχικά θα γίνει επίδειξη επιλογής ευφυών συσκευών βασισμένων σε μικροελεγκτή και συσκευών πομποδέκτη. Στη συνέχεια, θα χρησιμοποιηθούν τέτοιες συσκευές για να υλοποιηθεί ένα απλό τοπικό αυτοοργανούμενο δίκτυο μεταξύ κόμβων που αποτελούνται από αυτές τις συσκευές. Θα παρουσιαστούν ο τρόπος λειτουργίας τους και μερικά αποτελέσματα όσον αφορά την επίδοση της υλοποίησης αυτής.

4 Μία προσέγγιση για τη βελτίωση παροχής υπηρεσιών στις μεταφορές μέσω της αύξησης της ευφυΐας των υποστηρικτικών συστημάτων

Στα επόμενα θα εξηγηθεί μία προτεινόμενη προσέγγιση για τη συνολική διαδικασία που θα μπορούσε να ακολουθηθεί για επιλογή και σχεδίαση αυτοοργάνωσης τοπικών δικτύων μεταφοράς.

4.1 Αρχικά κριτήρια επιλογής ολοκληρωμένων συσκευών για τις σχετικές εφαρμογές της πράξης

Στα επόμενα περιγράφονται τα σημαντικότερα κριτήρια που χρησιμοποιήθηκαν για να γίνει η ταξινόμηση των συσκευών που βρέθηκαν και να γίνει η τελική επιλογή συσκευών για την υλοποίηση μίας αυτοοργάνωσης δικτύου τέτοιων συσκευών που γίνεται σε επόμενο κεφάλαιο.

4.1.1 Ολοκληρωμένες συσκευές με μικροελεγκτές

Με την ανάπτυξη της τεχνολογίας γίνεται προσπάθεια να αναπτύσσονται συνεχώς νέες πλακέτες ανάπτυξης. Σκοπός τους είναι είτε απλώς να έχουμε καλύτερες επιδόσεις ή και να προσφέρουν κάποια λύση σε συγκεκριμένα προβλήματα, με την προσθήκη ενός παραπάνω συστήματος ή μηχανισμού στο σχετικό σύστημα που έχει το πρόβλημα.

Επίσης, με την ανάπτυξη αυτών των συσκευών, γίνεται προσπάθεια για προσαρμογή σε νέες ανάγκες που εμφανίζονται στο χώρο του προγραμματισμού, αλλά και σε γενικότερους τομείς. Για παράδειγμα, μπορεί να θεωρηθεί σημαντικό μία πλακέτα να έχει ενσωματωμένο ένα σύστημα GPS ή WiFi, προκειμένου να είναι εύκολο ο προγραμματιστής να δοκιμάζει τις εφαρμογές του και άλλες, οι οποίες απαιτούν να υπάρχει ασύρματη επικοινωνία μεταξύ των πλακετών και με το δίκτυο γενικότερα, ή εντοπισμός τοποθεσίας της πλακέτας με σκοπό να είναι εύκολο να γίνει ανάκτησή της αν χαθεί χωρίς προειδοποίηση.

Λόγω της ύπαρξης τέτοιων αναγκών, οι οποίες μπορούν να τοποθετηθούν σε ποικίλες κατηγορίες, ως επακόλουθο είναι το γεγονός ότι παράγονται πλακέτες που προσπαθούν να ανταποκριθούν ξεχωριστά όσο μπορούν στις διαφορετικές αυτές κατηγορίες. Επομένως, η τιμή, οι καταναλώσεις ρεύματος, το μέγεθος, η θερμότητα, και άλλες παράμετροι ποικίλουν σε μεγάλο βαθμό προκειμένου να υπάρχει ικανοποίηση από κάθε χρήστη.

Για το σκοπό της εργασίας, είναι συνεπές να γίνει μία καταγραφή μερικών πολυχρησιμοποιημένων συσκευών τέτοιων βασισμένων σε μικροελεγκτή και στη συνέχεια μία επίδειξη του τρόπου επιλογής μίας ή και περισσότερων οι οποίες θα θεωρηθούν οι πιο ιδανικές για υλοποίηση ενός απλού δικτύου αυτοοργάνωσης που γίνεται σε επόμενο κεφάλαιο. Η καταγραφή δεν έγινε εξαντλητικά, αλλά αναζητήθηκαν οι πιο πολυχρησιμοποιημένες σχετικές συσκευές. Προκειμένου να επιτευχθεί η καταγραφή, είναι απαραίτητο αρχικά να οριστούν τα πιο σημαντικά κριτήρια για την κατηγοριοποίηση των πλακετών, όπως επίσης και για τη σωστή επιλογή τους. Χρειάζεται τα κριτήρια αυτά να στοχεύουν στις απαιτήσεις που θέλουμε να έχουμε από τις πλακέτες.

Κριτήριο επιλογής δεν είναι οποιοδήποτε χαρακτηριστικό της συσκευής, χωρίς σημαντικό λόγο ύπαρξης. Δεν είναι κάτι παραπάνω που προσφέρεται, το οποίο μπορεί ή όχι να βοηθάει με κάποιο τρόπο την εργασία. Για παράδειγμα, αν μία πλακέτα προσφέρει κάποια παθητική ψύξη, αυτό σίγουρα είναι κάτι θετικό που ίσως αρχικά φανεί ότι χρησιμεύει για την καλύτερη επίδοση της συσκευής. Όμως, θα παρατηρηθεί ότι γενικά η πλακέτα δε βρίσκεται σε κατάσταση συνεχούς λειτουργίας ή και κορεσμού πρακτικά ποτέ. Συνεπώς, μια ψύξη δεν προσφέρει κάτι για το συγκεκριμένο πρόγραμμα, και παρατηρείται ότι χωρίς αυτήν η πλακέτα βρίσκεται σε κανονικές και αποδεκτές θερμοκρασίες λειτουργίας. Παρατηρούμε λοιπόν ότι δε γίνεται μόνο να καταγράφουμε όλα τα χαρακτηριστικά ελπίζοντας ότι θα βρούμε μέσα από αυτά τη βέλτιστη λύση. Με αφαιρετική διαδικασία, απομονώνουμε τα χαρακτηριστικά, τα οποία πραγματικά συμβάλλουν στην επίλυση και στη συνέχεια μπορούμε να επιλέξουμε.

Επίσης, τα κριτήρια πρέπει να αναφέρονται σε κάποιο χαρακτηριστικό που όλες οι πλακέτες έχουν. Δηλαδή, για παράδειγμα, θα καταγράφονται η κατανάλωση, το μέγεθος, η αυτονομία κ.ά., τιμές οι οποίες υπάρχουν σε όλες τις πλακέτες και γίνεται εύκολα να ταξινομηθούν.

Δύσκολο είναι να ταξινομηθούν μοναδικά χαρακτηριστικά, όπως προηγουμένως αναφέρθηκε, για παράδειγμα, ενσωματωμένο GPS ή WiFi. Αυτά δεν αποτελούν αντικειμενικά και μετρήσιμα κριτήρια, οπότε δεν γίνεται απλά να ταξινομηθούν και να βαθμολογηθούν. Όμως, προκειμένου να γίνει η επιλογή, μπορούν απλά να αναφερθούν σε μία κατηγορία με Μοναδικά Χαρακτηριστικά, ώστε όταν κανείς δει τη λίστα συνολικά και ίσως είναι δύο πλακέτες που πρακτικά έχουν ίση συνολική βαθμολογία, να βοηθήσουν τα μοναδικά αυτά χαρακτηριστικά στην επιλογή.

Πρέπει να υπάρχουν λίγα κριτήρια, αλλά και αρκετά, τα οποία θα αναφέρονται πρακτικά στις απαιτήσεις που έχουμε για το πρόβλημά μας. Είναι σημαντικό να καλύπτουν από όλες τις κατευθύνσεις τις ανάγκες, προκειμένου να είναι σίγουρο ότι μετά την καταγραφή και κατάταξη των πλακετών θα μπορούμε εύκολα και τεκμηριωμένα να αποφασίσουμε ποια είναι η σωστή επιλογή.

Ως πιο σημαντικά κριτήρια θεωρήθηκαν τα εξής:

- Αυτονομία/Κατανάλωση Ρεύματος

- Τιμή

- Υποβοήθηση και δραστηριότητα στο διαδίκτυο

- Επίδοση Πλακέτας

- Βασικά Περιφερειακά

Στη συνέχεια εξηγούνται αυτά με λεπτομέρεια και οι λόγοι που επιλέχθηκαν.

Αυτονομία/Κατανάλωση Ρεύματος

Ένα μηχάνημα που δε λειτουργεί είναι άχρηστο. Έτσι, και στην περίπτωσή μας, πρέπει να είμαστε σίγουροι ότι η πλακέτα μας θα είναι “ζωντανή” για όλη τη διάρκεια του ταξιδιού που θα κάνει (η συσκευή πρόκειται να τοποθετηθεί σε κάποιο μέσο μεταφοράς αγαθών, όπως βαγόνι τρένου ή φορτηγό αυτοκίνητο). Κάθε συσκευή του αυτοοργανούμενου δικτύου τέτοιων υπολογιστικών συσκευών θα έχει διαφορετικές απαιτήσεις σε ρεύμα, διότι κάθε βαγόνι θα έχει διαφορετικά περιεχόμενα που, για παράδειγμα, απαιτούν συνεχή επικοινωνία με κάποιο κεντρικό κόμβο, ή συνεχή καταγραφή της θερμοκρασίας ενός θαλάμου διότι περιέχει ευαίσθητα προϊόντα που θα καθιστούν άχρηστα αν βρεθούν σε υψηλή θερμοκρασία για συγκεκριμένο χρονικό διάστημα.

Επίσης, κατά τη διάρκεια της μεταφοράς και αποθήκευσης, θα υπάρχουν, σε "τυχαίες" χρονικές περιόδους, υψηλές απαιτήσεις από μεγάλο μέρος του δικτύου, διότι θα υπάρχουν συνεχείς επικοινωνίες μεταξύ των κόμβων. Αυτό, σε συνδυασμό με το γεγονός ότι κάθε πλακέτα θα έχει συνδεδεμένα πάνω της διαφορετικά σε πλήθος μηχανήματα (όπως αισθητήρες), καθιστά την κατανάλωση ρεύματος ένα σημαντικό παράγοντα επιλογής. Τέλος, μπορεί να υπάρχουν άλλου είδους αλλαγές κατά τη διάρκεια του ταξιδιού, όπως για παράδειγμα κάποιο βαγόνι να χαθεί, να κλαπεί ή να πάθει βλάβη.

Αναφέρεται εδώ ότι οι πλακέτες, γενικά, δεν έχουν ενσωματωμένη μπαταρία, αλλά συνδέονται εξωτερικά. Έτσι, όταν μιλάμε για αυτονομία, μπορεί να γίνει η σύγκριση μόνο όταν χρησιμοποιούμε μία ίδιας χωρητικότητας μπαταρία ανάμεσα στις πλακέτες. Για αυτό το λόγο, θεωρείται πιο σημαντική και πιο ευλύγιστη η αναφορά της τυπικής κατανάλωσης ρεύματος, διότι είναι πιο εύκολο έτσι να γίνει η σύγκριση μεταξύ των συσκευών αυτών. Όμως, για καλύτερη κατανόηση και για να υπάρχει μία αίσθηση αυτής της κατανάλωσης, χρήσιμο είναι να αναφέρεται και η αυτονομία με χρήση μίας τυπικής χωρητικότητας μπαταρίας που ίσως θα χρησιμοποιόταν σε τέτοια περίπτωση. Προφανώς, η χωρητικότητα μπορεί να αλλάξει ακόμα και τάξη μεγέθους, διότι δεν υπάρχει αυστηρός περιορισμός στο χώρο στον οποίον πρόκειται να τοποθετηθεί η συσκευή με το μικροελεγκτή.

Τιμή

Είναι ίσως το πιο σημαντικό ή τουλάχιστον ένα από τα πιο σημαντικά κριτήρια για την εργασία μας, διότι είναι κριτήριο περιοριστικής φύσης.

Σίγουρα αποτελεί σημαντικό κριτήριο και εκτός της εργασίας, διότι προσφέρει ευλυγισία στην επιλογή. Δηλαδή, μία πλακέτα που είναι αρκετά οικονομική μπορεί να προσφέρει τη δυνατότητα ακόμα και να αντικαθίσταται ολόκληρη όταν τελειώνει η μπαταρία της ή όταν έχει κάποια όχι και πολύ σημαντική βλάβη. Σε άλλη περίπτωση, θα ήταν απαραίτητο να γίνει επισκευή της πλακέτας, διότι θα ήταν επιβλαβές οικονομικά αν γινόταν πλήρης αντικατάστασή της. Επίσης, κατά την επισκευή της, το βαγόνι, στο οποίο είχε τοποθετηθεί η συσκευή αυτή, θα είναι άχρηστο.

Φαίνεται, λοιπόν, πόσο σημαντικό κριτήριο επιλογής είναι η τιμή. Αναφέρθηκε η χρησιμότητα των πλακετών χαμηλής τιμής, όμως πρέπει να καταγραφούν και αυτές με υψηλότερη τιμή. Αυτό χρειάζεται επειδή, για μελλοντικές εργασίες που ίσως βασιστούν σε κάποιο βαθμό στην εργασία αυτή, μπορεί να χρειαστεί επέκταση στον αλγόριθμο που υλοποιήθηκε, άρα και μεγαλύτερη απαίτηση όσον αφορά την επίδοση ή άλλα τεχνικά χαρακτηριστικά. Άλλος πιθανός λόγος για να γίνει καταγραφή συσκευών με μικροελεγκτή με μεγαλύτερης τιμή είναι απλά η δοκιμή το αλγορίθμου σε συσκευές διαφορετικής αρχιτεκτονικής και δομής, ώστε να δειχθεί ότι λειτουργεί πλήρως και σε αυτές, χωρίς να υπάρχουν προβλήματα κατά την εκτέλεσή του. Επομένως, θα είναι πιο εύκολο να υπάρχουν και άλλες επιλογές όσον αφορά τις συσκευές μικροελεγκτών, οι οποίες θα μπορούν να ικανοποιούν μία πληθώρα απαιτήσεων.

Υποστήριξη και δραστηριότητα στο διαδίκτυο

Μαζί με το κριτήριο της τιμής, η υποστήριξη και η δραστηριότητα που υπάρχει στο διαδίκτυο αποτελούν τα πιο σημαντικά κριτήρια επιλογής τέτοιων συσκευών. Αν μία πλακέτα έχει εντυπωσιακά τεχνικά χαρακτηριστικά, αλλά δεν υπάρχει καμία αναφορά ή σχετικό forum στο διαδίκτυο, αυτό την καθιστά κατά μία έννοια άχρηστη. Σίγουρα, κανείς θα μπορούσε να τη χρησιμοποιήσει και να αναπτύξει ο,τι επιθυμεί πάνω της, αλλά βάζει τον εαυτό του σε μειονεκτική θέση, αφού θα χρειαστεί να αναπτύξει βασικά προγράμματα από

την αρχή (ενώ για τις πιο "γνωστές" συσκευές θα υπάρχουν αυτά ήδη έτοιμα και υλοποιημένα στο διαδίκτυο).

Αν επιλέξει μία πλακέτα με ίσως όχι τόσο καλά χαρακτηριστικά αλλά με σχετικές ιστοσελίδες να είναι πολύ ενεργές, αυτό θα βοηθήσει σε μεγάλο βαθμό. Καταρχάς, θα μπορεί να λύνει απορίες που μπορεί να έχει, για τις οποίες μπορεί είτε να βρει έτοιμες απαντήσεις μέσα από την υποστήριξη της εταιρίας παραγωγής της πλακέτας και μέσα από προηγούμενες αναρτήσεις στα σχετικά forums είτε, αν δε βρει απάντηση, μπορεί να αναρτήσει σχετική ερώτηση ο ίδιος στο διαδίκτυο και θα βρει απαντήσεις από την ενεργή κοινότητα. Η δραστηριότητα (activity) που βρίσκεται στο διαδίκτυο, γενικά, εξελίσσεται εκθετικά, οπότε, από τη μία, η επιλογή των γνωστών πλακετών θα προσφέρει όλα τα προαναφερθέντα θετικά, όμως με αυτήν την τακτική ίσως καταργούνται πλακέτες που θα είχαν μεγαλύτερες δυνατότητες ανάπτυξης, τελικά.

Εφόσον υπάρχει αρκετή υποστήριξη και έχει γίνει αρκετά γνωστή η πλακέτα, φυσικό επακόλουθο είναι οι εταιρίες παραγωγής συσκευών προορισμένων για σύνδεση με τέτοιες συσκευές βασισμένες σε μικροελεγκτή να προσπαθούν να προσαρμόζουν το σχεδιασμό των προϊόντων τους ώστε να είναι πιο εύκολη και άμεση η σύνδεση και η χρήση τους. Επομένως, εάν χρησιμοποιηθεί "διάσημη" πλακέτα, θα υπάρχουν περισσότερες επιλογές προϊόντων για αγορά, κάτι που μας ενδιαφέρει, αφού θέλουμε να συνδέσουμε αρκετές συσκευές πάνω τους, ώστε να προσφέρουν πολλές λειτουργίες, όπως, για παράδειγμα, την ασύρματη επικοινωνία μεταξύ τους. Επίσης, εφόσον μία συσκευή με μικροελεγκτή γίνεται "διάσημη" είναι φυσικό να δημιουργούνται, από χρήστες και από τις ίδιες τις εταιρίες, βιβλιοθήκες για την πλήρη και αποτελεσματική χρήση των συσκευών που συνδέονται με την πλακέτα.

Τέλος, οι γνωστές συσκευές θα έχουν έτοιμα ολοκληρωμένα περιβάλλοντα ανάπτυξης (IDE – integrated development environment), όπως για παράδειγμα το arduino, το οποίο έχει, από την επίσημη ιστοσελίδα του, το arduino ide για εύκολη εφαρμογή και compilation του κώδικα που επιθυμείται να εκτελεστεί από τη συσκευή αυτή. Τέλος, όσο πιο "διάσημη" είναι μία πλακέτα, δημιουργούνται περισσότεροι μεταγλωττιστές γλωσσών προγραμματισμού, ώστε κανείς να μπορεί να προγραμματίζει στη γλώσσα που επιθυμεί και είναι πιο εξοικειωμένος και εξειδικευμένος.

Επίδοση Πλακέτας

Σε αυτό το κριτήριο περιλαμβάνονται η ταχύτητα του επεξεργαστή, η χωρητικότητα της μνήμης, αλλά και η αρχιτεκτονική που χρησιμοποιείται από αυτά. Αποτελεί βασική παράμετρος μίας πλακέτας, διότι αφορά κάθε εφαρμογή που μπορεί να γίνει πάνω της. Έχοντας μία πλακέτα με καλή, σε γενικές γραμμές, επίδοση, εξασφαλίζουμε ότι δε θα υπάρχει πρόβλημα όσον αφορά μεγάλο εύρος απαιτήσεων που μπορεί να έχουν οι εφαρμογές, τόσο στη χωρητικότητα αλλά και στην ταχύτητα.

Σημαντικό είναι να αναφέρουμε, όμως, ότι η χρησιμότητα αυτή εξαρτάται από τις απαιτήσεις της εφαρμογής μας. Αν μία εφαρμογή χρειάζεται πολύ μικρή επίδοση προκειμένου να λειτουργεί ακόμα και υπό βαριές (για την εφαρμογή αυτή) συνθήκες, τότε το κριτήριο αυτό θα μειώνεται σε σημαντικότητα. Αν το δούμε και αντίθετα, όμως, αποτελεί και περιοριστικό κριτήριο, διότι είναι υποχρεωτικό, τουλάχιστον, να καλύπτονται οι ανάγκες της εφαρμογής, οι οποίες αφορούν την επίδοση, αλλιώς δε θα μπορεί να πραγματοποιηθεί σωστά και αποδοτικά η εφαρμογή αυτή.

Μέσα σε αυτό το κριτήριο περιέχεται και η αρχιτεκτονική που χρησιμοποιείται, με κύριο χαρακτηριστικό τα bits ανά εντολή. Πιο συγκεκριμένα, τα bits είναι σημαντικά όσον αφορά την ασύρματη επικοινωνία μεταξύ τέτοιων συσκευών (κάτι που μας ενδιαφέρει), διότι

μπορεί να πραγματοποιηθεί αυτή πιο πυκνά ως προς τα δεδομένα όσον αυξάνονται τα bits της αρχιτεκτονικής της συσκευής μικροελεγκτή. Έτσι, πλακέτες με αρχιτεκτονική με περισσότερα bits ανά κομμάτι εντολής θα καθιστούν πιο χρήσιμες για την εφαρμογή μας.

Βασικά Περιφερειακά

Σε αυτό το κριτήριο δεν περιέχονται τα μοναδικά χαρακτηριστικά, τα οποία αναφέρθηκαν προηγουμένως. Αφορά τις συνδέσεις που επιτρέπονται να γίνουν άμεσα με την πλακέτα, τα πρωτόκολλα που μπορούν να εφαρμοστούν με αυτές και άλλα βασικά hardware χαρακτηριστικά.

Πιο συγκεκριμένα, θα αναφέρονται πόσα pins υπάρχουν για GPIO (general purpose input output), αναλογικά και ψηφιακά, τι πρωτόκολλα μπορούν να εφαρμοστούν (πχ. Universal asynchronous receiver transmitter - UART, inter integrated circuit - I2C, serial peripheral interface bus – SPI κλπ.). Επίσης, θα καταγράφονται οι συνδέσεις που μπορούν να γίνουν μεταξύ συσκευής και προσωπικού υπολογιστή, ώστε να υπάρχει σειριακή επικοινωνία με τον υπολογιστή, και οι συνδέσεις για την ηλεκτρική τροφοδοσία της.

Αποτελεί ίσως το πιο ασήμαντο κριτήριο, ειδικά για την περίπτωση της υλοποίησης που παρουσιάζεται σε επόμενο κεφάλαιο. Αυτό επειδή, γενικά, δε θα υπάρχουν πολλές συνδέσεις για να συνδεθούν τα wireless modules και να πραγματοποιηθεί η επικοινωνία. Παρ' όλα αυτά, πρέπει πάντα να κοιτάει κανείς και τις δυνατές μελλοντικές εργασίες που μπορεί να πραγματοποιηθούν. Οπότε, αν πρόκειται να γίνει κάποια επέκταση της εφαρμογής, θα τεθεί το κριτήριο αυτό ως πιο σημαντικό, για να ικανοποιηθούν οι αυξημένες ανάγκες σύνδεσης περισσότερων εξωτερικών περιφερειακών, όπως αισθητήρες, GPS, GSM κλπ.

4.1.2 Σχετικές συσκευές πομποδέκτη

Όσον αφορά τις συσκευές πομποδέκτη (transceiver modules), δεν υπάρχουν τόσες παράμετροι, οι οποίες να μπορούν να επηρεάσουν σημαντικά την επιλογή, σε σχέση με τις πλακέτες. Όμως, θα αναφερθούν οι πιο σημαντικές, ώστε να μπορέσει να γίνει μία ταξινόμηση μεταξύ τους και να βρεθεί η βέλτιστη για την εργασία.

Τα σχετικά κριτήρια επιλογής είναι τα εξής:

- Τιμή

- Εμβέλεια

- Ρυθμός μεταφοράς δεδομένων

- Κατανάλωση Ρεύματος/Αυτονομία

Σημείωση: Θα μπορούσε να αναφερθεί και το online support ως κριτήριο, όμως οι συσκευές που επιλέχθηκαν έχουν όλες επαρκές και περίπου ίδιο σε ποσότητα υλικό στο διαδίκτυο, οπότε δεν περιλαμβάνεται.

Τα κριτήρια της τιμής και της κατανάλωσης ρεύματος περιγράφηκαν στα προηγούμενα και μπορούν να εφαρμοστούν τα επιχειρήματα που διατυπώθηκαν όμοια και για τις συσκευές που αναζητούνται.

Στη συνέχεια έχουμε τις επεξηγήσεις των προαναφερθέντων κριτηρίων.

Εμβέλεια

Αν υπάρχει σωστό δίκτυο ασύρματης επικοινωνίας, αλλά δεν επαρκεί η εμβέλεια για να φτάσει κάποιο μήνυμα από ένα κόμβο σε κάποιο άλλον, τότε, στην πραγματικότητα, δεν υπάρχει κανένα δίκτυο. Μπορεί, επίσης, μέσω των φύλλων δεδομένων, να φαίνεται ότι υπάρχει επαρκής εμβέλεια για την εφαρμογή που πρόκειται να υλοποιήσει κανείς. Όμως, στην πράξη, τελικά, μπορεί να δειχθεί ότι δεν επαρκεί. Αυτό συμβαίνει διότι η εμβέλεια που αναφέρεται στα φύλλα δεδομένων της συσκευής πιθανώς να αποτελεί την εμβέλεια line of sight, δηλαδή αυτή που θα έχει η συσκευή αν δεν υπάρχει κανένα φυσικό εμπόδιο μεταξύ αυτής και άλλης. Στην πρακτική χρήση, θα υπάρχουν πάντα φυσικά εμπόδια, όπως τοίχοι, λαμαρίνες κλπ, αλλά εμπόδιο θεωρείται επίσης ακόμα και η αυξημένη υγρασία. Είναι, λοιπόν, απαραίτητο να υπάρχει επαρκής εμβέλεια για να μπορούν να γίνουν οι επιθυμητές ασύρματες συνδέσεις κάτω από τις προαναφερόμενες συνθήκες.

Εδώ πρέπει να αναφερθεί ότι, γενικά, το δίκτυο σχεδιάζεται με σκοπό όταν δεδομένα μεταφέρονται μεταξύ κόμβων με μεγάλη απόσταση, τότε θα συμβάλλουν οι ενδιάμεσοι κόμβοι διαβιβάζοντας το μήνυμα. Έτσι, ενώ αρχικά μπορεί να μην υπήρχε αρκετή εμβέλεια, τώρα δημιουργείται μεγαλύτερη ενεργή εμβέλεια (αφού το μήνυμα μεταφέρεται από γειτονικό σε γειτονικό κόμβο, μέχρι να φτάσει στον τελικό προορισμό), ώστε να φτάσει με επιτυχία. Οπότε η πιο σημαντική εμβέλεια που πρέπει να τηρείται είναι αυτή ώστε τελικά στο δίκτυο να υπάρχει δυνατή γραμμή επικοινωνίας μεταξύ όλων των συσκευών.

Η μεγάλη εμβέλεια που προσφέρει μία συσκευή πομποδέκτη, χρησιμεύει ώστε να έχουμε μεγαλύτερη ευλυγισία στη δομή του δικτύου. Πιο συγκεκριμένα, αρχικά, θα μπορούν να υπάρχουν συνδέσεις με συσκευές που δεν είναι διαδοχικές, αλλά βρίσκονται πιο μακριά, προσδίδοντας έτσι στο δίκτυο μεγαλύτερο ρυθμό μεταφοράς δεδομένων, αφού δε θα χρειάζεται να μεταφέρεται η πληροφορία από τις ενδιάμεσες και το μήνυμα θα φτάνει στον τελικό προορισμό πιο σύντομα. Επίσης, γενικά, θα υπάρχουν διαφορετικές δομές των μέσων μεταφοράς (βαγονιών, εμπορευματοκιβωτίων κλπ.) στις διαφορετικές εφαρμογές (πχ. στα τραίνα και σε σειρά φορτηγών είναι σειριακή η δομή του δικτύου, όμως, στην περίπτωση των φορτηγών, οι αποστάσεις κυμαίνονται μεταξύ τους δεν είναι σταθερές, στα πλοία υπάρχει δομή πιο πλεγματική όπου κάθε συσκευή θα έχει περισσότερους γείτονες κλπ.). Η χρήση συσκευής πομποδέκτη με αρκετή εμβέλεια επιτρέπει να σχεδιαστούν εφαρμογές, οι οποίες να μπορούν να χρησιμοποιηθούν για διαφορετικές δομές, χωρίς αλλαγές στο σχεδιασμό των εφαρμογών αυτών.

Ρυθμός μετάδοσης δεδομένων

Γενικά, δεν αποτελεί πολύ σημαντικό κριτήριο στην υλοποίηση που γίνεται σε επόμενο κεφάλαιο, διότι δε στέλνεται μεγάλη ποσότητα δεδομένων. Όμως, πρέπει να συμπεριληφθεί, για δυνατότητα επέκτασης της εργασίας. Δηλαδή, για παράδειγμα, αν χρησιμοποιηθούν πολλοί αισθητήρες σε κάθε πλακέτα και απαιτηθεί τα δεδομένα που δίνουν αυτοί να μεταφέρονται σε κάποιο συγκεκριμένο κόμβο του δικτύου, ο οποίος στέλνει αυτά τα δεδομένα στο Σταθμό Βάσης, τότε αυξάνεται η σημαντικότητα.

Αυτή η παράμετρος γίνεται πιο σημαντική για κόμβους που βρίσκονται σε κεντρικά σημεία του δικτύου, επειδή θα αναγκάζονται να λειτουργούν ως διαμεσολαβητές για πολλές επικοινωνίες. Έτσι, σε σειριακά δίκτυα, όπου ο μέγιστος αριθμός γειτόνων είναι το 2, δεν είναι τόσο σημαντικό. Όμως, αν υπάρχει δίκτυο με πολλές συνδέσεις, όπως ίσως σε πλοίο, στο οποίο υπάρχουν μέχρι και 6 γειτονικά εμπορευματοκιβώτια, τότε φαίνεται η αξία του μεγάλου ρυθμού δεδομένων.

4.2 Επίδειξη εφαρμογής των κριτηρίων σε επιλεγμένες συσκευές της αγοράς

Αφού αναφέραμε τα πιο σημαντικά κριτήρια για να επιλέξει κανείς τις συσκευές για εφαρμογή, παρακάτω έχουμε τις συσκευές που βρέθηκαν και περιγράφονται σε προηγούμενο κεφάλαιο. Σκοπός ήταν όχι η εξαντλητική επιλογή, αλλά να δειχθεί ο τρόπος που γίνεται επιλογή ανάμεσα σε παραπλήσιες συσκευές.

4.2.1 Συσκευές πομποδέκτη

Για να βρούμε την καλύτερη λύση, τοποθετούμε τις συσκευές σε έναν πίνακα, όπου θα δίνονται συνοπτικά τα χαρακτηριστικά τους σύμφωνα με τα κριτήρια που αναφέρθηκαν. Για την κατανάλωση τοποθετήθηκε μία μέση τιμή μετάδοσης και λήψης. Μπορεί να μην είναι ακριβής σε σχέση με τη λειτουργία, όμως έτσι μπορούμε να συγκρίνουμε καλύτερα. Προτιμήθηκε η κατανάλωση αντί για αυτονομία με την αυθαίρετα επιλεγμένη μπαταρία 1000mAh (η οποία αναφέρεται σε προηγούμενο κεφάλαιο στο οποίο δίνεται περιγραφή των συσκευών), επειδή καλύπτει σε μεγαλύτερο βαθμό αυτό το κριτήριο. Η καταλληλότητα είναι απλή βαθμολογία κλίμακας 1 – 3.

Συσκευή	Τιμή (ευρώ)	Εμβέλεια (m)	Data Rate (Kbps)	Κατανάλωση (mA)	Καταλληλότητα (1-3)	Σχόλιο
NRF24L01 L Long Range Wireless Module R2 (1.1KM, PA+SMA)	3.85	200 - 2000	250 - 2000	80	3	
NRF24L01 2.4GHz Wireless Transceiver Module	1.89	10	<2000	11.8	3	
UM801 passthrough (UART) SX1276 wireless module LoRa	15	1700 – 6000	0.3 – 31.2	66.5	2	
RFM95 Ultra-long Range transceiver	6.5	2000	<300	65.15	2	

module/lora module/support 868m frequency						
Adafruit feather 32u4 RFM95 LoRa radio	30	2000	<300	65.15	1	Δεν προσφέρει ελαστικότητα (πλακέτα +συσκευή μαζί)
Digi XBee-PRO s2c ZigBee	28.5	90 - 3200	<1000	82.5	1	
Xbee ZigBee s2c	19.4	60 - 1200	<1000	38	2	
RFM12B	4.5	150	<115.2	17.75	2	
RFM69HW	8	400 - 500	<300	44.5	2	

Ο λόγος που πήραν 3/3 μόνο οι συσκευές nRF24L01 είναι ότι έχουν καλές όλες τις παραμέτρους, ενώ ταυτόχρονα είναι οι πιο εύκολες στη χρήση, αφού απλά συνδέονται και έχουν έτοιμες βιβλιοθήκες. Από τη μία η πρώτη έχει καλή εμβέλεια, ενώ η δεύτερη όχι, αλλά η δεύτερη έχει καλύτερη τιμή. Τελικά, θεωρήθηκε ως καλύτερη επιλογή η δεύτερη, αφού συγκεκριμένα για τα πειράματα της εργασίας μας θέλουμε να μπορούμε σχετικά εύκολα να ελέγχουμε τι γίνεται όταν οι συσκευές φεύγουν εκτός εμβέλειας, και αποτελεί τη πιο φθηνή λύση. Προφανώς, αν επιθυμήσει κανείς να εφαρμόσει την εργασία σε πρακτική εφαρμογή, τότε θα είναι καλύτερη συσκευή η nrf24l01 με τη μεγαλύτερη εμβέλεια (έχουμε δηλαδή και με αυτές τις συσκευές άμεση επεκτασιμότητα, αφού αρκεί να αλλάξει κανείς μόνο τη συσκευή, χωρίς να γίνει καμία άλλη αλλαγή στον κώδικα).

4.2.2 Συσκευές μικροελεγκτή

Τοποθετούμε τις πλακέτες σε έναν πίνακα, προκειμένου να είναι εύκολη η σύγκρισή τους. Συμπεριλαμβάνονται τα κριτήρια που προαναφέρθηκαν, όπως επίσης και μία υποκειμενική συνολική βαθμολογία κλίμακας 1 – 3, ώστε να βρεθεί η καλύτερη επιλογή. Παρόμοια βαθμολογία τοποθετήθηκε και για τα κριτήρια Online Support, επίδοση και περιφερειακά, μιας και δεν υπάρχει απόλυτη μέτρηση. Στην τελευταία στήλη καταγράφονται επιγραμματικά τα μοναδικά χαρακτηριστικά που μπορεί να έχει η κάθε πλακέτα, όπως ενσωματωμένο WiFi.

Πλακέτα	**Τιμή (ευρώ)**	**Κατανάλωση (mA)**	**Υποστήριξη διαδικτύου (1-3)**	**Επίδοση (1-3)**	**Περιφερειακά (1-3)**	**Καταλληλότητα (1-3)**	**Μοναδικά Χαρακτηριστικά**
Arduino UNO	4	46.5	3	1	3	3	
Raspberry Pi 3	40	200 – 400	3	3	3	3	WiFi, Bluetooth

Raspberry Pi Zero	5	80 – 200	3	3	2	2	
BeagleBone Black	50	300	2	3	3	2	2Xmicrocontrollers on board
pcDuino8 Uno	42	n/a	1	3	2	1	
freetronics Goldilocks	70	51.5	2	2	3	1	
Teensy 3.6	30	50	1	2	2	2	
Intel Edison	100	50	1	3	1	1	1Xmicrocontroller on board
Arduino Due	16	130	3	2	3	2	32-bit αρχιτεκτονική
Arduino Nano	3	19	3	1	2	3	

Παρατηρείται ότι οι πιο κατάλληλες πλακέτες είναι οι Arduino και Raspberry. Αυτό επειδή, παρόλο που σε άλλες πλακέτες μπορεί να έχουμε καλύτερα τεχνικά χαρακτηριστικά, δεν έχουν αρκετή υποστήριξη στο διαδίκτυο, κριτήριο το οποίο είναι ίσως το πιο βασικό, διότι επιτρέπει την ευκολότερη και πληρέστερη ανάπτυξη οποιουδήποτε προγράμματος. Τελικά, για την υλοποίηση που πρόκειται να γίνει, επιλέχθηκε Arduino, και πιο συγκεκριμένα τα Arduino Uno και Nano, επειδή πρώτον αποτελούν τις πιο φθηνές λύσεις, δεύτερον υπάρχουν πολλές έτοιμες βιβλιοθήκες για τις συσκευές πομποδέκτη, και τρίτον είναι πιο άμεση η χρήση τους για σκοπούς του επιπέδου της εργασίας μας, όπου επιθυμούμε απλά να γίνει κάποια επικοινωνία, χωρίς χρήση ολόκληρου λογισμικού που έχει το Raspberry.

4.3 Εφαρμογή στην πράξη: Σχεδίαση αυτοοργανούμενου δικτύου και πειραματικές δοκιμές

Επιλέγοντας την συσκευή nRF24L01 για συσκευή πομποδέκτη για ασύρματη επικοινωνία σε τοπικό δίκτυο, και τα Arduino UNO και Nano ως συσκευές μικροελεγκτή με σκοπό την αύξηση της ευφυΐας και την αυτοοργάνωση των τοπικών δικτύων, προχωρούμε στην περαιτέρω ανάλυσή τους και τη σχεδίαση μίας απλής δομής αυτοοργάνωσης. Αρχικά θα δοθούν περαιτέρω πληροφορίες για τις συσκευές που χρησιμοποιήθηκαν, απαραίτητες για την πλήρη κατανόηση της δομής και του αλγορίθμου για το σύστημα.

4.3.1 Επιπλέον πληροφορίες για τον πομποδέκτη nRF24L01

Στη συνέχεια παρατίθενται ορισμένες βασικές πληροφορίες για το συγκεκριμένο module οι οποίες θεωρήθηκαν απαραίτητες ώστε να υπάρξει μία καλύτερη κατανόηση της βιβλιοθήκης που θα χρησιμοποιηθεί για αυτό, των συνδέσεων που έγιναν και γενικότερων λειτουργιών του. Πάρθηκαν από το data sheet του προϊόντος [75].

Ο επόμενος πίνακας περιέχει τις βασικές πληροφορίες όσον αφορά τις απαιτήσεις σε ρεύμα, τις συνθήκες λειτουργίας και τις δυνατότητες ισχύος:

Παράμετρος	**Τιμή**
Minimum supply voltage	1.9 V
Maximum output power	0 dBm
Maximum data rate	2000 kbps
Supply current in TX (transmit) mode @ 0dBm output power	11.3 mA
Supply current in RX (receive) mode @ 2000 kbps	12.3 mA
Temperature range	-40 to +85 ºC
Sensitivity @ 1000kbps	-85 dBm
Supply current in Power Down mode	900 nA

Οι υπόλοιπες παράμετροι θεωρήθηκαν βαθύτερου επιπέδου που δε μας αφορά στην εργασία.

4.3.1.1 Περιγραφή λειτουργίας

Στη συνέχεια έχουμε έναν πίνακα που αναφέρει τους τρόπους με τους οποίους μπορεί να λειτουργεί το nRF24L01, ανάλογα με την τιμή βασικών καταχωρητών του. Αυτοί είναι οι:

- PWR_UP register (power up)

- PRIM_RX register (primary receiver)

- CE (chip enable)

Σημαντικό είναι εδώ να σημειώσουμε σύντομα (θα εξηγηθεί πιο λεπτομερώς στα επόμενα) ότι τα δεδομένα προς αποστολή ή λήψη γίνονται σε ένα pipe, οπότε έχουμε λογική FIFO στους buffers στους οποίους αποθηκεύονται τα δεδομένα προς αποστολή ή που λήφθηκαν. Ως TX και RX συμβολίζονται οι όροι transmit και receive αντίστοιχα.

Κατάσταση	**PWR_UP register**	**PRIM_RX register**	**CE**	**FIFO state**
RX mode	1	1	1	-
TX mode	1	0	1	Data in TX FIFO
TX mode	1	0	1→0	Stays in TX mode until packet transmission is finished
Standby-II	1	0	1	TX FIFO empty
Standby-I	1	-	0	No ongoing packet transmission
Power Down	0	-	-	-

Καταστάσεις Standby:

- Η κατάσταση Standby-I χρησιμοποιείται για να υπάρχει ελάχιστη κατανάλωση ρεύματος από τον πομποδέκτη, ενώ διατηρούνται σχετικά σύντομοι χρόνοι εκκίνησης λειτουργίας του.
- Στην κατάσταση Standby-II υπάρχουν μερικοί περισσότεροι clock buffers του πομποδέκτη σε λειτουργία. Βρισκόμαστε σε αυτήν την κατάσταση όταν είναι CE = 1 δηλαδή το chip είναι αναμμένο αλλά το TX FIFO είναι άδειο, οπότε δεν έχει ο πομποδέκτης δεδομένα για αποστολή.

Κατάσταση Power Down:

Σε αυτήν την κατάσταση έχουμε ελάχιστη κατανάλωση ρεύματος, με την μονάδα να βρίσκεται εκτός λειτουργίας. Παρόλο που δεν είναι ενεργή, διατηρούνται σταθεροί όλοι οι καταχωρητές της από το SPI interface, και έτσι αυτό μπορεί να ενεργοποιηθεί, όταν γίνει επιθυμητό.

Μέθοδοι χειρισμού πακέτων

ShockBurst™:

Η μέθοδος αυτή επιτρέπει τη χρήση του μεγάλου ρυθμού δεδομένων που προσφέρει το nRF24L01, χωρίς την ανάγκη υποστήριξης από μικροελεγκτή για να τα επεξεργάζεται.

Τοποθετούνται όλες οι λειτουργίες επεξεργασίας σημάτων μεγάλης ταχύτητας RF πρωτοκόλλου πάνω στο chip, παρέχοντας έτσι στο μικροελεγκτή που χειρίζεται την εφαρμογή ένα απλό SPI interface για την ενσύρματη επικοινωνία μεταξύ πομποδέκτη και συσκευής μικροελεγκτή.

Κατά τη λήψη μηνύματος, γίνεται interrupt request (IRQ) από τον πομποδέκτη, που ενημερώνει το μικροελεγτή όταν έχει ληφθεί ένα σωστό μήνυμα στη σωστή διεύθυνση. Στη συνέχεια, ο μικροελεγκτής μπορεί να πάρει από τον RX FIFO τα δεδομένα.

Κατά την αποστολή, παράγονται αυτόματα η εισαγωγή του μηνύματος και το CRC για αυτό (CRC – Cyclic Redundancy Check είναι ένας κώδικας εντοπισμού λαθών που χρησιμοποιείται στην επικοινωνία και στην αποθήκευση δεδομένων. Χρησιμοποιείται για να βρεθούν τυχαία λάθη στα δεδομένα.) Το interrupt request (IRQ) ενημερώνει το μικροελεγτή ότι η αποστολή είναι επιτυχής.

Ο μικροελεγτής έχει πρόσβαση στα TX και RX FIFOs όταν το nRF24L01 βρίσκεται σε οποιαδήποτε κατάσταση λειτουργίας, ακόμα και στην power down.

Enhanced ShockBurst™:

Με αυτή τη μέθοδο γίνεται προσπάθεια για πιο εύκολη και αποδοτική επικοινωνία. Πιο συγκεκριμένα, αφήνεται στο nRF24L01 να χειρίζεται και την αναγνώριση (acknowledgement) των ληφθέντων πακέτων και την επαναποστολή των χαμένων, χωρίς να συμβάλλει ο μικροελεγκτής.

Φαίνεται, άρα, ότι έχει γίνει προσπάθεια για τη μείωση της κατανάλωσης ρεύματος, παράμετρος που μας αφορά στην εργασία μας, διότι σκοπός είναι οι μικροελεγτές να τοποθετηθούν στα βαγόνια, τα οποία μπορεί να μην έχουν παροχή ρεύματος, οπότε θα γίνει χρήση μπαταρίας.

4.3.1.2 Διαδικασία λήψης και αποστολής δεδομένων

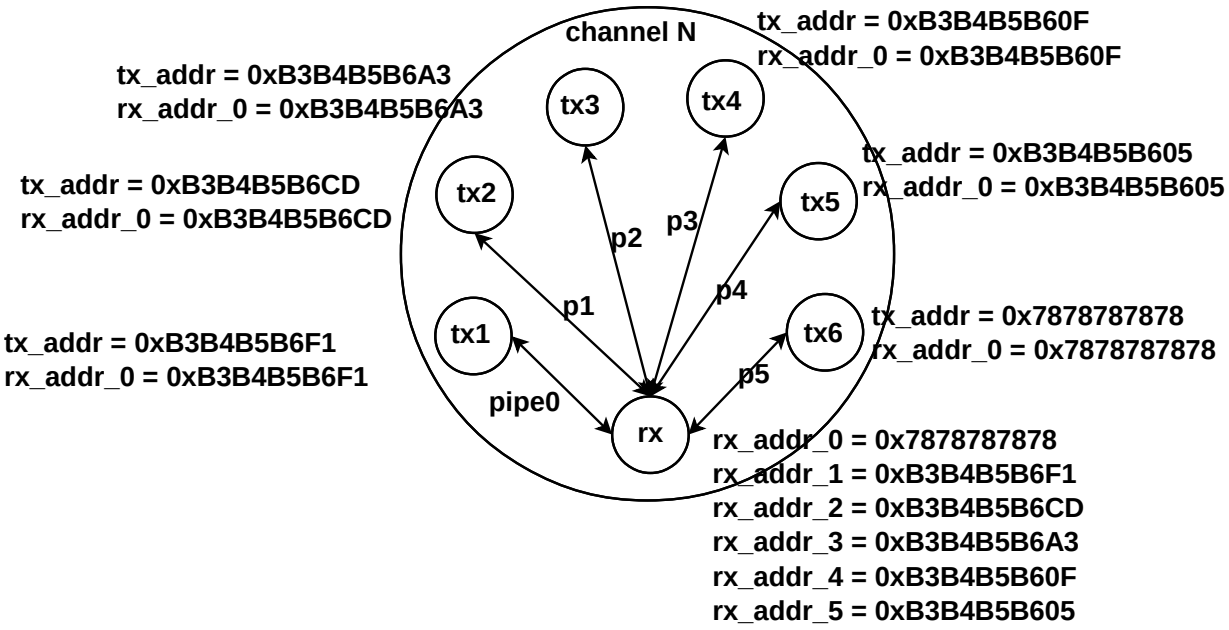
Όταν το nRF24l01 έχει ρυθμιστεί ως βασικός δέκτης (primary RX – PRX) τότε μπορεί να λάβει δεδομένα μέσα από 6 διαφορετικά data pipes. Αυτά τα pipes θα λαμβάνουν δεδομένα από κανάλι ίδιας συχνότητας, αλλά θα έχουν δικές τους διαφορετικές διευθύνσεις. Δηλαδή, γίνεται να υπάρχουν 6 άλλες συσκευές nRF24l01 που να έχουν ρυθμιστεί ως πομποί και να στέλνουν στην προαναφερθείσα κοινή συχνότητα, αλλά στις διαφορετικές διευθύνσεις που ο δέκτης έχει θέσει στα pipes του. Ο δέκτης θα μπορέσει να ξεχωρίσει τα δεδομένα που του έρχονται από τους 6 πομπούς.

Τα data pipes αριθμούνται από 0-5, ώστε ο μικροελεγκτής να μπορεί να αναφέρεται σε αυτά σύμφωνα με την αρίθμηση αυτή. Όσον αφορά τις διευθύνσεις που αντιστοιχούνται σε κάθε pipe για την ασύρματη επικοινωνία μεταξύ τέτοιων συσκευών, έχουμε τα εξής. Το pipe που είναι αριθμημένο με 0 έχει διεύθυνση 40 bits. Τα υπόλοιπα 5 data pipes (1-5) θα έχουν και αυτά 40 bits, όμως θα έχουν κοινά μεταξύ τους τα πρώτα (most significant) 32 bits.

Έτσι, μεταξύ τους θα αποκτούν μοναδικότητα αλλάζοντας τα τελευταία 8 bits (least significant). Απαραίτητο είναι οι διευθύνσεις μεταξύ όλων των pipes της συσκευής να είναι διαφορετικές. Για αποστολή χρησιμοποιείται το pipe 0. Επομένως, ο καταχωρητής TX_ADDR αφορά το pipe 0. Μπορεί, όμως, το pipe 0 να χρησιμοποιηθεί και για λήψη γενικότερων δεδομένων, αλλά χρησιμοποιείται αναγκαστικά για λήψη του ACK (acknowledge), όπως θα δούμε στη συνέχεια.

Για το χειρισμό των δεδομένων, είναι χρήσιμο σε πολλές περιπτώσεις να απαντά ο δέκτης στον πομπό όταν πήρε όλο το πακέτο σωστά με ένα acknowledge (ACK) μήνυμα. Χρησιμοποιείται το ίδιο data pipe από το οποίο λήφθηκε το πακέτο για να σταλθεί το ACK μήνυμα . Ο πομπός που περιμένει να του έρθει το ACK, ώστε να γνωρίζει αν χρειάζεται να ενεργήσει αναλόγως, στέλνοντάς το ξανά ή αλλιώς, διαβάζει από το data pipe 0, μέσω του οποίου είχε στείλει αρχικά το μήνυμα.

Στο επόμενο σχήμα φαίνεται καλύτερα η λογική αυτή:



Οι μικροί κύκλοι αντιπροσωπεύουν ξεχωριστές συσκευές nRF24l01. Παρατηρούμε αρχικά ότι όλοι βρίσκονται μέσα στον κύκλο channel N, δηλαδή χρησιμοποιούν κοινή συχνότητα N προκειμένου να επικοινωνήσουν μεταξύ τους. Έχουμε μία συσκευή που είναι ρυθμισμένη ως κύριος δέκτης (rx) και 6 συσκευές που είναι ρυθμισμένες ως πομποί (tx1 – tx6). Όπως αναφέρθηκε προηγουμένως, η συσκευή rx έχει θέσει τα 6 pipes του σε ορισμένες διευθύνσεις, όπου το pipe 0 έχει μοναδική 40 bit , ενώ τα άλλα έχουν κοινά τα πρώτα 32 bits. Κάθε πομπός που θέλει να στείλει στον rx πρέπει να έχει το tx_addr του ίσο με μία από τις διευθύνσεις. Προκειμένου να μην υπάρχουν συγκρούσεις μηνυμάτων στα ίδια pipes και το rx να μη γνωρίζει ποιος έστειλε τι, κάθε tx στέλνει σε διαφορετικό pipe. Έστω το tx1 στέλνει ένα μήνυμα μέσω του pipe 0xB3B4B5B6F1. Το rx λαμβάνει το μήνυμα και στέλνει μέσω του ίδιου pipe το ACK. Προκειμένου να μπορέσει να λάβει το ACK το tx1, έχει θέσει και την rx_addr_0 με την ίδια τιμή του writing pipe.

Πιο συγκεκριμένα, για κάθε nRF24l01 που έχει ρυθμιστεί ως πομπός, αφού στείλει κάποιο πακέτο, τότε θέτει τη λειτουργία του ως δέκτης, ώστε να μπορέσει να λάβει το ACK. Αν δεν το λάβει μετά από ένα χρονικό διάστημα, το οποίο ρυθμίζεται μέσω προγραμματισμού της συσκευής, τότε γίνεται ξανά αποστολή του πακέτου μέχρι να λάβει το ACK. Αν περάσει ένας αριθμός προσπαθειών (ο οποίος είναι ίσος με τον καταχωρητή SETUP_RETR_ARC) τότε σταματάει τις προσπάθειες αποστολής, γίνεται interrupt και φαίνεται τι έγινε στο STATUS καταχωρητή στο bit MAX_RT.

Γενικά για τα pipes υπάρχουν αρκετές ρυθμίσεις, οι οποίες μπορούν να αλλαχθούν είτε για ένα είτε για όλα. Μερικές συνηθισμένες ρυθμίσεις είναι οι: CRC ενεργοποιημένο ή οχι, κωδικοποίηση του CRC, πλάτος της διεύθυνσης του δέκτη, συχνότητα του καναλιού, ρυθμός δεδομένων, ισχύς σήματος εξόδου κ.ά.

Auto Acknowledgement (RX)

Η λειτουργία auto acknowledgement μειώνει την απαίτηση από τον εξωτερικό μικροελεγκτή. Μπορεί να ρυθμιστεί ανεξάρτητα για κάθε pipe. Όταν ενεργοποιείται για κάποιο pipe και έρθει ένα σωστό πακέτο, δηλαδή να έχει σωστό pipe address και CRC (εξηγείται στη συνέχεια), τότε η συσκευή μπαίνει σε κατάσταση πομπού και στέλνει πακέτο acknowledgement, χωρίς την χρήση του κύριου μικροελεγκτή. Στη συνέχεια επαναφέρεται η λειτουργία της συσκευής ως δέκτης.

Packet Identity (PID) and CRC

Κάθε πακέτο έχει ένα 2 bit μέρος που λέγεται packet identity. Χρησιμοποιείται με σκοπό να καταλάβει ο δέκτης αν το πακέτο που μόλις λήφθηκε είναι νέο ή αν είναι το ίδιο απεσταλμένο ξανά. Το packet identity αρχίζει από το 0 και αυξάνεται κατά 1 κάθε φορά που ο πομπός στέλνει ένα νέο μήνυμα. Για μεγαλύτερη επιτυχία στην αναγνώριση νέου πακέτου, ο δέκτης χρησιμοποιεί και το PID και το CRC του πακέτου. Πιο συγκεκριμένα, αν ένα πακέτο έχει το ίδιο PID με κάποιο προηγούμενο, τότε ο δέκτης θα ελέγξει τα CRC των πακέτων. Αν είναι και αυτά ίδια, τότε είναι το ίδιο πακέτο και μπορεί να αγνοηθεί.

4.3.1.3 Διαμόρφωση συσκευής

Όλη η διαμόρφωση της συσκευής καθορίζεται από τις τιμές μερικών καταχωρητών της, οι οποίοι μπορούν να αλλάξουν τιμή μέσω SPI interface [76] μεταξύ αυτής και της συσκευής μικροελεγκτή.

Στο σημείο αυτό είναι σημαντικό να δοθούν μερικές πληροφορίες για το SPI, που είναι η μέθοδος με την οποία το nRF24l01 συνδέεται με το μικροελεγκτή για τη μεταφορά δεδομένων μεταξύ τους.

Το SPI (Serial Peripheral Interfact bus) αποτελεί μία διεπαφή σύγχρονης σειριακής επικοινωνίας (synchronous serial communication interface specification) για μικρές σε απόσταση επικοινωνίες (ενσύρματες κυρίως), συνήθως σε ενσωματωμένα συστήματα.

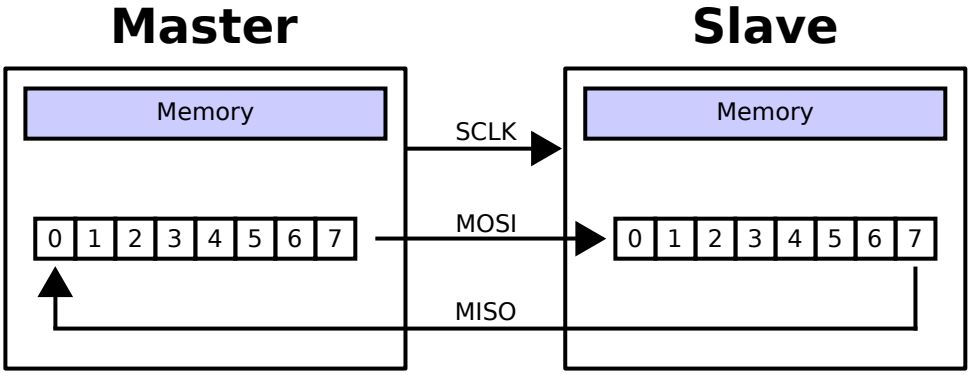
Στη δική μας περίπτωση, χρησιμοποιούνται μόνο 3 σήματα, δηλαδή θα υπάρχουν 3 συνδέσεις μεταξύ του nRF24L01 και του μικροελεγκτή όσον αφορά το SPI:

- SCLK: serial clock (output from master)

- MOSI: master output → slave input

- MISO: master input ← slave input

Στο ακόλουθο σχήμα φαίνεται πώς γίνεται η επικοινωνία.



Για να ξεκινήσει η επικοινωνία, πρέπει πρώτα ο master να θέσει τη συχνότητα του ρολογιού σε κάποια τιμή που να αντέχει ο slave. Για το nRF24l01 είναι μέχρι 10MHz, ή αν αντιστοιχηθεί για δεδομένα θα είναι 10Mbps. Αφού θέσει τη συχνότητα αυτή, μέσω του SCLK, στέλνει το ρολόι ως σήμα και στο slave ώστε να είναι συγχρονισμένα. Ανά κύκλο SPI γίνεται διπλή μεταφορά δεδομένων. Ο master στέλνει το MSB του καταχωρητή που θέλει να στείλει στο LSB του καταχωρητή του slave μέσω του MOSI. Ο slave ταυτόχρονα στέλνει το MSB του δικού του καταχωρητή στο LSB του καταχωρητή του master. Αυτή η διαδικασία γίνεται ακόμα και αν ο slave δε έχει να στείλει κάτι στο master ή αντίστροφα.

Οπότε, ανά κύκλο SPI μπαίνουν ένα ένα τα bits και ο καταχωρητής γίνεται shifted δεξιά κατά ένα, έτσι ώστε, μετά από τόσους κύκλους όσο είναι το μέγεθος του καταχωρητή, (8 κύκλοι στην περίπτωση του σχήματος) να σταλθούν όλα τα περιεχόμενα του καταχωρητή . Στο nRF24L01, ενώ στέλνεται μέσω του MOSI pin κάποια εντολή, ταυτόχρονα επιστρέφεται μέσω του MISO ο καταχωρητής STATUS.

Οι εντολές που υπάρχουν για το nRF24l01 SPI interface είναι, επιγραμματικά, οι εξής:

- Read register
- Write register
- Read RX payload (1-32 bytes)
- Write TX payload (1-32 bytes)
- Flush TX FIFO
- Flush RX FIFO
- Reuse last sent payload.
- NOP – No Operation.

Περιγραφή πακέτου αποστολής ή λήψης

Ένα πακέτο, με χρήση της μεθόδου Enhanced ShockBurst™ που αναφέρθηκε προηγουμένως, με δεδομένα 1-32 bytes έχει την εξής μορφή:

Preamble	Address 3-5 byte	9 flag bits	Payload 1-32 byte	CRC 0-2 byte

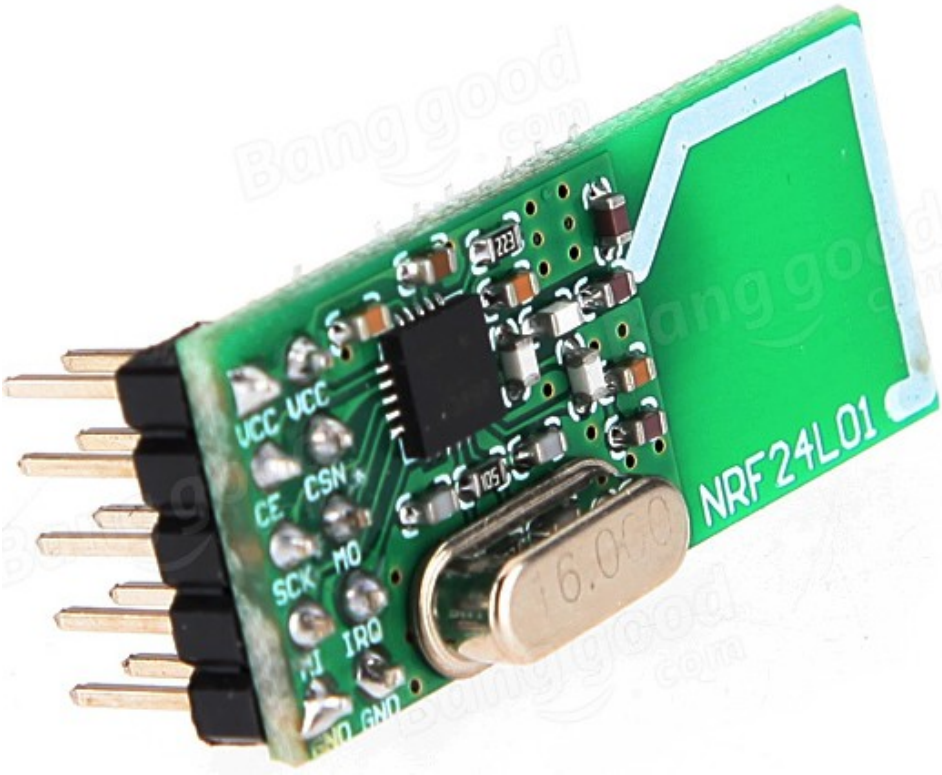
Με τη μέθοδο ShockBurst™ το πακέτο έχει την ίδια μορφή χωρίς όμως τα 9 flag bits.

Preamble	Χρησιμοποιείται για την αναγνώριση των επιπέδων του 0 και 1.
Address	Η διεύθυνση του δέκτη
Flags	- Περιέχει τα 2 bits για το packet identification (PID). - Τα υπόλοιπα 7 bits είναι δεσμευμένα για συμβατότητα με μελλοντικά προϊόντα.
Payload	1-32 bytes μέγεθος
CRC	Είναι προαιρετικό.

4.3.1.4 Σύνδεση του nRF24L01 με Arduino

Πηγαίνοντας, τώρα, σε πιο πρακτικό επίπεδο, χρειάζεται να δούμε τα pins του nRF24L01 και που πρέπει να συνδεθούν αυτά στην πλακέτα με μικροελεγκτή που θα χρησιμοποιήσουμε, το Arduino UNO και Arduino Nano.

Το nRF24L01 που χρησιμοποιούμε φαίνεται στην επόμενη εικόνα.



Τα pins που είναι απαραίτητα για τη σωστή λειτουργία και χρήση του είναι τα εξής:

- Vcc: Τιμές που δέχεται για τάση τροφοδοσίας είναι 1.9V-3.6V, τυπική 3.0V, οπότε συνδέεται απλά στο pin 3.3V των Arduino. Σημαντικό εδώ είναι να σημειωθεί ότι το Arduino UNO έχει και δυνατότητα τροφοδοσίας τάσης 5V, που όμως εδώ δε μας χρειάζεται και θα πρέπει κανείς να προσέξει να μην συνδέσει σε αυτήν την τάση το nRF24L01 διότι υπάρχει πιθανότητα να προκληθεί βλάβη. Στο Arduino Nano έχουμε μόνο τροφοδοσία 3.3V οπότε δεν υπάρχει κίνδυνος. Υπάρχουν δύο συνδέσεις Vcc στο nRF24L01 για άνεση. Τα άλλα pins αντέχουν σήμα λογικής των 5V οπότε δε χρειάζεται να γίνει κάποια αλλαγή για αυτά όταν συνδέονται με arduino uno το οποίο έχει τέτοια λογική.
- GND: Γείωση, αποτελεί απλά αναφορά μηδενικής τάσης ώστε να λαμβάνονται σωστές τιμές μέσω των συνδέσεων και μέσω του ασύρματου. Συνδέεται με το GND του arduino.
- IRQ: Interrupt Request, δεν αποτελεί απαραίτητη σύνδεση στην περίπτωσή μας.

Πρέπει να αναφερθεί ότι τα arduino που χρησιμοποιούμε έχουν ενσωματωμένο SPI στα εξής pins:

- MOSI – pin 11
- MISO – pin 12

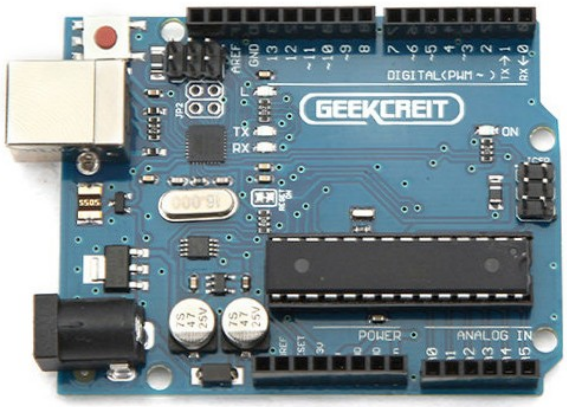
- SCK – pin 13

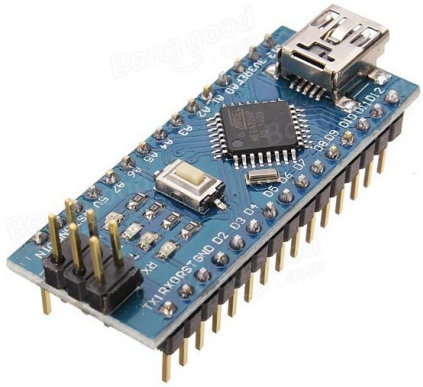
Επομένως γίνονται και αυτές οι συνδέσεις. Οι υπόλοιπες συνδέσεις είναι οι ακόλουθες.

- CE – pin 7: Chip Enable, μέσω αυτής ενεργοποιείται η λειτουργία της συσκευής ως πομπού ή δέκτη. Το pin αυτό του πομποδέκτη συνδέεται σε οποιοδήποτε ψηφιακό pin, το οποίο τίθεται στο μέρος του κώδικα. Στο δικό μας κώδικα συνδέεται με το pin 7.
- CSN – pin 8: Chip Select NOT: Έχουμε το NOT οπότε είναι ενεργό χαμηλά (ενεργοποιείται όταν CSN = 0). Όταν CSN = 0, το nRF24L01 περιμένει εντολή μέσω του SPI. Το pin αυτό του πομποδέκτη συνδέεται επίσης σε οποιοδήποτε ψηφιακό pin, το οποίο τίθεται στο μέρος του κώδικα.

Arduino Uno και Nano χρήσιμες πληροφορίες

Ακολούθως έχουμε εικόνα το arduino uno και nano τα οποία χρησιμοποιήθηκαν για το μέρος της υλοποίησης.





Από τροφοδοσία, το uno μπορεί να πάρει είτε μέσω θύρας usb είτε μέσω dc θύρας. Στην dc θήρα μπορεί να τοποθετηθεί τροφοδοσία 7-12V, η οποία μπορεί να δοθεί είτε μέσω μετασχηματιστή είτε με απλή μπαταρία των 9V. Όσον αφορά το arduino nano, έχουμε μόνο την επιλογή της usb, όμως υπάρχουν πλέον μετασχηματιστές που έχουν ως έξοδο τη θύρα που χρησιμοποιεί. Γενικά, όμως, είναι πιο εύκολο να χρησιμοποιείται απλά η usb, διότι συνδέεται στον υπολογιστή, από όπου περνιούνται τα προγράμματα στις πλακέτες, ενώ επίσης γίνεται και σειριακή επικοινωνία αν χρειαστεί να υπάρξει κάποια είσοδος ή έξοδος πληροφορίας από/προς το χρήστη. Επομένως, χρησιμοποιήθηκε usb για τροφοδοσία στην εργασία.

Για pins, τα arduino έχουν, αρχικά, 6 αναλογικά pins εισόδου. Αυτά δε χρειάζονται άμεσα στην εφαρμογή μας, αλλά αποτελούν βασικό στοιχείο αν κανείς θέλει να αναπτύξει την εφαρμογή αυτή περαιτέρω, διότι οι περισσότεροι αισθητήρες (θερμοκρασίας, υγρασίας κλπ) δίνουν αναλογική έξοδο. Εφόσον αναφερόμαστε σε εφαρμογές σε χώρους που μπορεί να υπάρχουν τρόφιμα ή φάρμακα, λογική είναι μία επέκταση της εφαρμογής με εισαγωγή τέτοιων αισθητήρων. Στη συνέχεια, τα Arduino έχουν 14 ψηφιακά pins, από τα οποία χρησιμοποιούμε τα 11,12,13 για την επικοινωνία μέσω SPI με το nRF24L01 και τα 7,8 για τις τιμές chip enable και chip select NOT αντίστοιχα.

Για τις συνδέσεις χρησιμοποιήθηκαν κοινά jumper cables για τέτοιου είδους συνδέσεις, είτε αρσενικά προς θηλυκά για το arduino uno είτε θηλυκά προς θηλυκά για το arduino nano.

Breadboard δε χρειάστηκε, διότι οι συνδέσεις έγιναν άμεσα από τις πλακέτες στα nRF24L01.

4.3.2 Βιβλιοθήκη του Arduino για τη συσκευή nRF24L01

Θεωρείται αναγκαίο να περιγραφούν οι συναρτήσεις της βιβλιοθήκης που χρησιμοποιήθηκε για τον προγραμματισμό του Arduino που είναι σε σύνδεση με τον πομποδέκτη nRF24L01 [24], ώστε να γίνει πιο κατανοητή στη συνέχεια η υλοποίηση του αλγορίθμου.

Η βιβλιοθήκη βρέθηκε στο διαδίκτυο και έχει ονομασία:

"Optimized High Speed NRF24L01+ Driver Class Documentation"

Παρατηρείται αρχικά ότι αναφέρεται στο όνομα το μοντέλο nRF24L01+ το οποίο αποτελεί μία πιο νέα έκδοση της συσκευής που έχουμε, η οποία δεν έχει το "+" στην ονομασία της. Αυτό, όμως, δεν ενοχλεί τη χρήση της βιβλιοθήκης, αφού, πρώτον, δεν υπάρχουν αρκετά θεμελιώδεις διαφορές ανάμεσα στις δύο συσκευές, ώστε να τις καθιστούν αδύνατες να χρησιμοποιηθούν με τον ίδιο τρόπο, και, δεύτερον, η βιβλιοθήκη έχει σημεία όπου ελέγχει ποιο ακριβώς μοντέλο χρησιμοποιείται και ανάλογα εκτελεί τις σωστές εντολές.

Υπάρχουν κάποιες επεκτάσεις της βιβλιοθήκης οι οποίες αναφέρονται για πληρότητα, με ονομασίες:

- -RF24Network: OSI (open systems interconnection) δίκτυο για επικοινωνία μεταξύ πολλαπλών συσκευών. Σκοπός η δημιουργία δικτύου αισθητήρων στο σπίτι.
- RF24Mesh: Δυναμικό δίκτυο τύπου mesh για το RF24Network
- RF24Ethernet: TCP/IP δικτύωση τύπου mesh
- RF24Audio: Streaming ήχου σε πραγματικό χρόνο

Δε χρησιμοποιήθηκαν οι επεκτάσεις αυτές επειδή θεωρήθηκαν πιο πολύπλοκες από 'τι χρειάζεται.

Η βιβλιοθήκη αυτή μπορεί να χρησιμοποιηθεί στις εξής συσκευές βασισμένες σε μικροελεγτή:

- Arduino

- ATTiny

- Συσκευές βασισμένες σε λειτουργικά συστήματα Linux (Raspberry pi, Intel Edison κλπ.)

Εμείς πρόκειται να χρησιμοποιήσουμε Arduino uno και nano. Όπως έχει αναφερθεί, η βιβλιοθήκη είναι σχεδιασμένη χρησιμοποιώντας τις εξής συνδέσεις μεταξύ των pins του Arduino και του πομποδέκτη nRF24L01:

nRF24L01	**Arduino**
GND (ground)	GND
Vcc	3.3V

CE (chip enable)	digIO (digital input/output) pin 7
CSN (chip select NOT)	digIO pin 8
SCK (serial clock)	digIO pin 13
MOSI (master out slave in)	digIO 11
MISO (master in slave out)	digIO 12
IRQ (interrupt request)	χωρίς σύνδεση

Χρησιμοποιείται ο τρόπος ενσύρματης επικοινωνίας SPI για την επικοινωνία μεταξύ Arduino και nRF24L01, ο οποίος εξηγήθηκε σε προηγούμενη παράγραφο.

Στη συνέχεια παρουσιάζονται, με σύντομη επεξήγηση, οι συναρτήσεις της βιβλιοθήκης που χρησιμοποιήθηκαν. Δεν αναφέρονται με λεπτομέρεια οι τύποι δεδομένων που χρησιμοποιούν, επειδή σκοπός αυτής της επεξήγησης είναι περισσότερο πληρέστερη κατανόηση, στη συνέχεια, του αλγορίθμου:

1. RF24(cepin, cspin)

Αρχικοποιεί τις βασικές παραμέτρους του nRF24L01. Πιο συγκεκριμένα, τίθενται, μέσω των παραμέτρων εισόδου της συνάρτησης, ποια είναι τα pins για τις συνδέσεις με CE και CS της συσκευής. Στη συνέχεια, τίθεται το μέγεθος payload ίσο με 32 bytes. Αυτό είναι το μέγεθος του πραγματικού μηνύματος μέσα στο πακέτο το οποίο κάθε φορά στέλνεται. Για σταθερότητα, απενεργοποιεί τη λειτουργία των δυναμικών payloads, ώστε κάθε φορά να έχουμε πακέτο με ίδιο μέγεθος και να μην υπάρχει πρόβλημα ανάμεσα στη συσκευές κατά τη λήψη, δηλαδή, για παράδειγμα, μία συσκευή να θεωρεί ότι θα λάβει 32 bytes payload, ενώ η άλλη να έχει στείλει 5 bytes.

2. begin()

Ολοκληρώνει τις υπόλοιπες απαραίτητες αρχικοποιήσεις. Πιο συγκεκριμένα:

- Θέτει τα cepin και cspin, τα οποία είναι γνωστά από την αμέσως προηγούμενη συνάρτηση όπου αποθηκεύτηκαν στη μνήμη μέσω των παραμέτρων εισόδου, ως εξόδους σημάτων. Αυτό επειδή η πλακέτα είναι υπεύθυνη για τον έλεγχο της ενεργοποίησης και απενεργοποίησης του πομποδέκτη και για την επικοινωνία μεταξύ της πλακέτας και του πομποδέκτη.
- Θέτει μέγεθος CRC (Cyclic Redundancy Check) ίσο με 16 bits.
- Θέτει τις επαναλήψεις αποστολής μηνύματος που απέτυχε να φτάσει, δηλαδή δεν λήφθηκε ποτέ acknowledgement πακέτο από τον αποστολέα, ίσες με 5 και τα διαστήματα που θα περιμένει ανάμεσα σε αυτές ίσο με 15ms.
- Θέτει το ρυθμό δεδομένων ίσο με 1MBps, τιμή ελάχιστη για όλες τις συσκευές, με σκοπό την εξασφάλιση σωστής λειτουργίας, χωρίς να χάνονται πακέτα λόγω αυξημένης ταχύτητας.
- Τέλος, αδειάζει τους buffers για αποστολή για λήψη (TX, RX).

3. openReadingPipe(number, address)

Ανοίγει το Pipe για ανάγνωση. Όπως έχει αναφερθεί, το nRF24L01 μπορεί να διαβάσει από μέχρι και 6 pipes, οπότε η παράμετρος number ελέγχει ποιο pipe από αυτά θα ανοίξει, ενώ η παράμετρος address ρυθμίζει τη διεύθυνση του pipe. Θυμίζουμε εδώ ότι, γενικά, όλοι οι πομποδέκτες θα στέλνουν στην ίδια συχνότητα, και θα ελέγχεται, από τον αποστολέα, σε ποια συσκευή θα πάει το κάθε πακέτο μέσω της διεύθυνσης αυτής.

4. openWritingPipe(address)

Ανοίγει Pipe για εγγραφή. Όταν γίνεται αυτή η διαδικασία, αυτόματα ανοίγει και reading pipe στη θέση 0 με την ίδια διεύθυνση. Αυτό γίνεται επειδή πρόκειται, εφόσον είναι ενεργοποιημένη η λειτουργία αυτή, να λάβει ο αποστολέας του πακέτου ένα πακέτο acknowledgment, σταλμένο από τον παραλήπτη, το οποίο εξασφαλίζει ότι ο παραλήπτης έλαβε το πακέτο ολόκληρο και σωστό.

5. startListening()

Ξεκινάει να ακούει για πακέτα που στέλνονται στις διευθύνσεις των ανοιγμένων pipes. Σημαντική σημείωση εδώ είναι ότι, αν έχει κληθεί το openWritingPipe προηγουμένως, έχει ανοίξει και reading pipe στη θέση 0 με ίδια διεύθυνση. Επειδή δεν είναι επιθυμητό να λάβουμε κάποιο άλλο πακέτο εκτός από το acknowledgement που πρόκειται να ληφθεί μετά από την επιτυχή αποστολή πακέτου, γίνεται έλεγχος αν είχε καλεστεί αυτή η συνάρτηση πριν. Αν ναι, τότε κλείνει το reading pipe 0, πριν καλεστεί η συνάρτηση startListening.

6. stopListening()

Σταματάει να ακούει τα pipes. Σημαντικό είναι να καλείται αυτή η συνάρτηση πριν από κάθε αποστολή μηνύματος, επειδή ο πομποδέκτης δεν είναι σχεδιασμένος να λειτουργεί ταυτόχρονα ως πομπός και ως δέκτης. Επομένως, ο χειρισμός όσον αφορά σε ποια από τις δύο καταστάσεις, πομπού ή δέκτη, βρίσκεται η συσκευή γίνεται με χρήση αυτής και της αμέσως προηγούμενης συνάρτησης.

7. available()

Ελέγχει αν υπάρχουν δεδομένα έτοιμα για διάβασμα σε κάποιο από τα ανοιχτά reading pipes.

8. read(*buf, len)

Διαβάζει πακέτο από οποιοδήποτε ανοιχτό και με πληροφορία reading pipe. Το πακέτο αποθηκεύεται στο χώρο μνήμης που δείχνει ο δείκτης buf, και το μέγεθος σε bytes που θα διαβαστεί και θα αποθηκευτεί στο χώρο αυτό είναι len.

9. write(*buf, len)

Γράφει πακέτο στο ανοιγμένο writing pipe. Το πακέτο που στέλνεται δείχνεται από το δείκτη buf και τα bytes είναι ίσα με len.

10. closeReadingPipe(pipe)

Κλείνει το reading pipe με αριθμό pipe. Αυτή η συνάρτηση είναι χρήσιμη αν αλλάζει καταστάσεις λειτουργίας η συσκευή (παράδειγμα στην αρχή να λαμβάνει από δύο pipes ενώ μετά να λαμβάνει από ένα μόνο) και δεν επιθυμεί να λαμβάνει πακέτα από πολλαπλές διευθύνσεις.

11. printDetails()

Εκτυπώνει τις σημαντικότερες παραμέτρους του nRF24L01, όπως ποια pipes είναι ανοιχτά και σε ποιες διευθύνσεις, τι ισχύς χρησιμοποιείται για αποστολή και λήψη πακέτων κλπ. Η σχετική συνάρτηση είναι ιδιαίτερα χρήσιμη για debugging, αφού επιτρέπει σε οποιοδήποτε

σημείο επιθυμεί ο χρήστης να γνωρίζει τι ακριβώς γίνεται στον κόμβο του δικτύου, και κατ' επέκταση σε όλο το δίκτυο.

12. powerDown()

Απενεργοποιεί το nRF24L01. Είναι χρήσιμη συνάρτηση όταν αποχωρεί κάποιος κόμβος από το δίκτυο, για να σταματήσει να λαμβάνει πακέτα και να μειώσει την κατανάλωση ρεύματος.

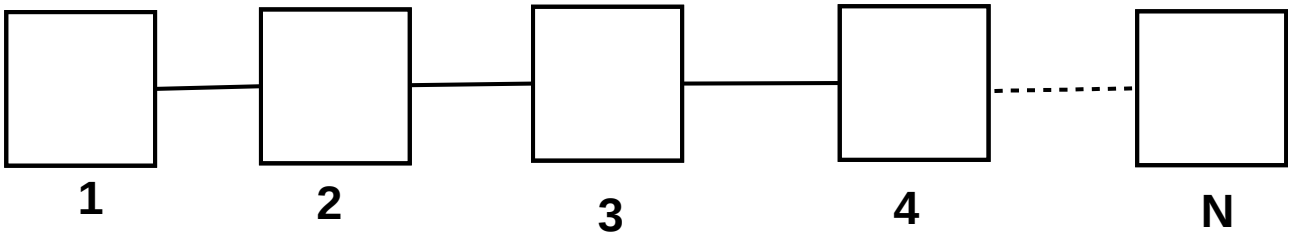
Αυτές είναι οι σημαντικότερες συναρτήσεις που χρησιμοποιήθηκαν στην εφαρμογή που ακολουθεί. Υπάρχουν και άλλες πολλές στη βιβλιοθήκη που καλύπτουν κάθε πιθανή ανάγκη που μπορεί να υπάρξει χρησιμοποιώντας το nRF24L01, όμως για τη δομή του δικτύου της εργασίας χρειάστηκαν αυτές.

5 Υλοποίηση απλού δικτύου αυτοοργάνωσης για κατηγορίες εφαρμογών που μας ενδιαφέρουν

5.1 Δομή δικτύου που υλοποιήσαμε

5.1.1 Σταθερή λειτουργία του δικτύου

Το δίκτυο της εργασίας μας έχει σειριακή δομή, δηλαδή κάθε κόμβος έχει σύνδεση με το πολύ δύο άλλους. Σχηματικά, έχουμε το επόμενο.

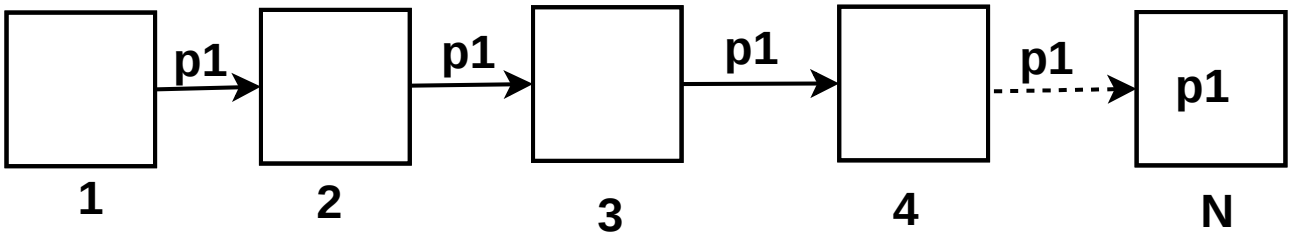


Παρατηρούμε ότι σε κάθε κόμβο έχει δοθεί ένας αριθμός. Αυτός ο αριθμός θα αντιστοιχεί στη διεύθυνση του κόμβου.

Η διεύθυνση κάθε κόμβου μπορεί να αλλάζει κατά τη λειτουργία του δικτύου, όπως θα δούμε στα επόμενα. Κάθε κόμβος στην αρχή της λειτουργίας του δεν έχει κάποια δική του διεύθυνση, αλλά είτε δίνεται σε αυτόν μία διεύθυνση από το ήδη υπάρχον δίκτυο, είτε, αν είναι ο πρώτος κόμβος του δικτύου, τότε θα θέσει στον εαυτό του μία διεύθυνση και θα εκκινήσει το δίκτυο.

Επειδή δεν υπάρχει άμεση σύνδεση μεταξύ όλων των κόμβων, πρέπει να χρησιμοποιηθεί ο,τι υπάρχει αν γίνει επιθυμητό να μεταφερθούν δεδομένα σε μη γειτονικούς κόμβους. Πιο συγκεκριμένα, αυτή η τοπολογία ονομάζεται daisy chain, και ο τρόπος επικοινωνίας μεταξύ μη γειτονικών κόμβων είναι ο εξής:

Κάθε πακέτο που στέλνεται περνάει από τον εικονικό διάδρομο που έχει δημιουργηθεί με τους κόμβους. Αν πρόκειται να σταλθεί πακέτο, θα περάσει από γειτονικό σε γειτονικό κόμβο, μέχρι να φτάσει στον προορισμό του. Αν δούμε το επόμενο σχήμα, ο κόμβος 1 θέλει να στείλει ένα πακέτο με ονομασία p1 στον κόμβο N. Προκειμένου να το μεταφέρει εκεί, το στέλνει αρχικά στον κόμβο 2. Ο κόμβος 2, με τη σειρά του, γνωρίζει ότι αυτό το πακέτο προορίζεται για τον N που βρίσκεται πιο μετά του στη σειρά του δικτύου, οπότε και αυτός το στέλνει στον επόμενο γειτονικό του, ο οποίος είναι ο 3. Ακολουθώντας την ίδια διαδικασία, βήμα βήμα, το πακέτο φτάνει στον κόμβο N-1, ο οποίος το στέλνει στον N. Τελικά, αφού φτάσει στον N, αυτός κρατάει το πακέτο, διαβάζει τα δεδομένα που περιέχει και δρα ανάλογα.



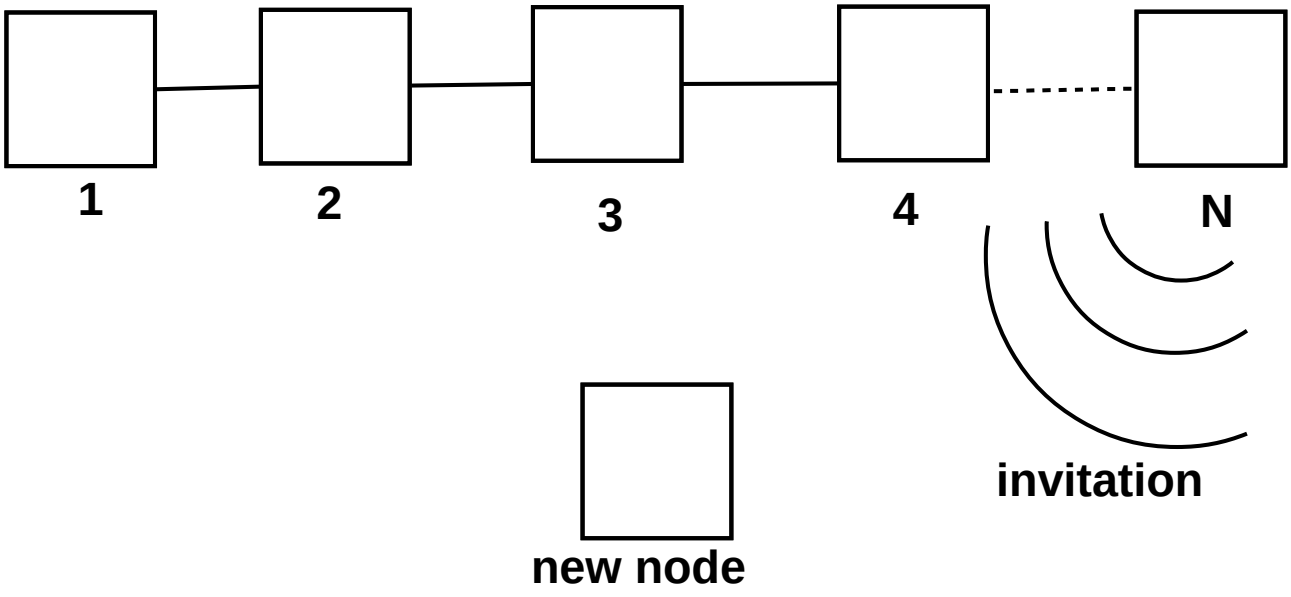
Σημαντικό είναι εδώ να αναφερθεί ότι θα μπορούσε να υπάρχει μία επέκταση του δικτύου αυτού, αν υπήρχε αρκετή εμβέλεια. Δηλαδή, θα μπορούσαν να υπάρχουν περισσότερες συνδέσεις εκτός από τις γειτονικές, ώστε να μπορούσε το πακέτο να διανύσει μεγαλύτερες αποστάσεις και να γίνονται πιο γρήγορα οι επικοινωνίες. Αυτό θα καθιστούσε τον αλγόριθμο πιο γρήγορο, αλλά στο επίπεδο της εργασίας μας εκτελέσαμε τη βασική δομή, που είναι πιο λογικό να γίνει πρώτα αυτό, ώστε να είναι πιο εύκολο να αντιμετωπιστεί το οποιοδήποτε πρόβλημα με μεγαλύτερη ευκολία και να υπάρχει τελικά ένα βασικό αλλά σταθερό πρόγραμμα.

Με τη δομή αυτή έχουμε την πιο μεγάλη ενεργή εμβέλεια, ελαχιστοποιώντας τη πιθανότητα σφάλματος λόγω αυτής.

5.1.2 Σύνδεση νέου κόμβου στο δίκτυο

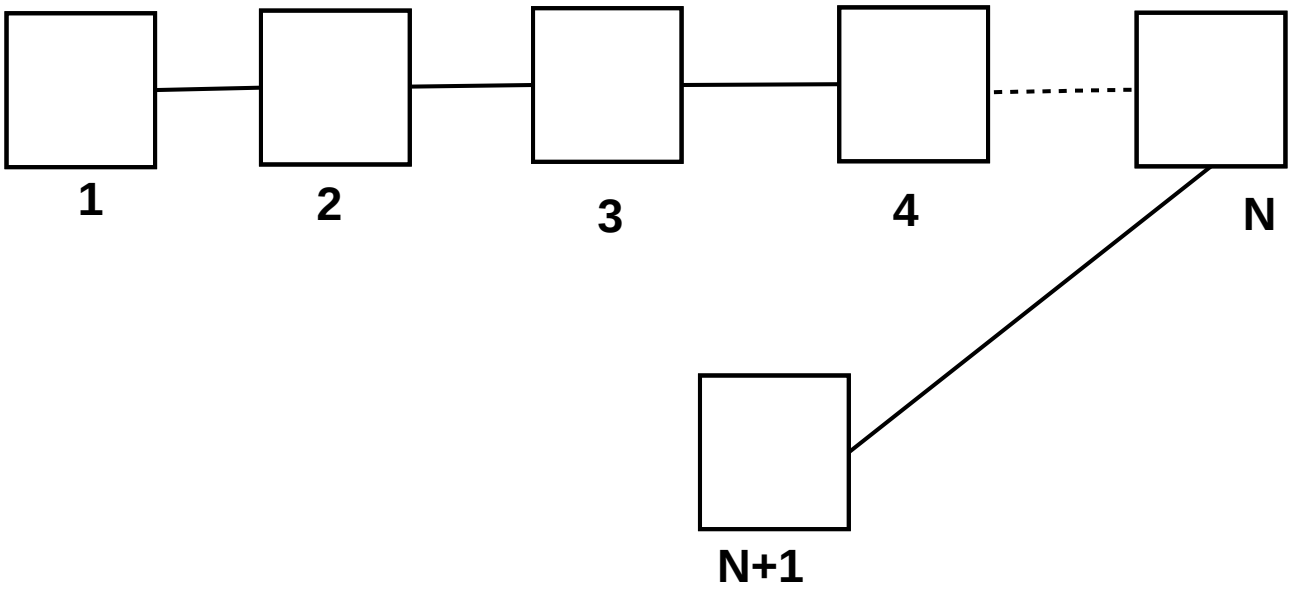
Ένα τέτοιο δίκτυο πρέπει να είναι δυναμικό. Δηλαδή, πρόκειται να συνδεθούν και να φύγουν κόμβοι κατά τη διάρκεια της λειτουργίας του, και πρέπει να γίνονται αυτόματα όλες οι αναγκαίες διαδικασίες προκειμένου να παραμείνουν όλοι οι υπάρχοντες κόμβοι ενήμεροι για το κάθε γεγονός που συνέβη.

Στην περίπτωση που πρόκειται να συνδεθεί κόμβος, πρέπει κάπως να δείξει την επιθυμία του αυτή. Στη δομή μας, αντί να προσπαθήσει ο νέος κόμβος να “πει” στο δίκτυο ότι θέλει να συνδεθεί, το δίκτυο στέλνει προσκλήσεις ανά τακτά χρονικά διαστήματα, προκειμένου να μπορεί να βρει αυτό αν υπάρχουν νέοι κόμβοι που θέλουν να συνδεθούν. Η διαδικασία φαίνεται στο επόμενο σχήμα.

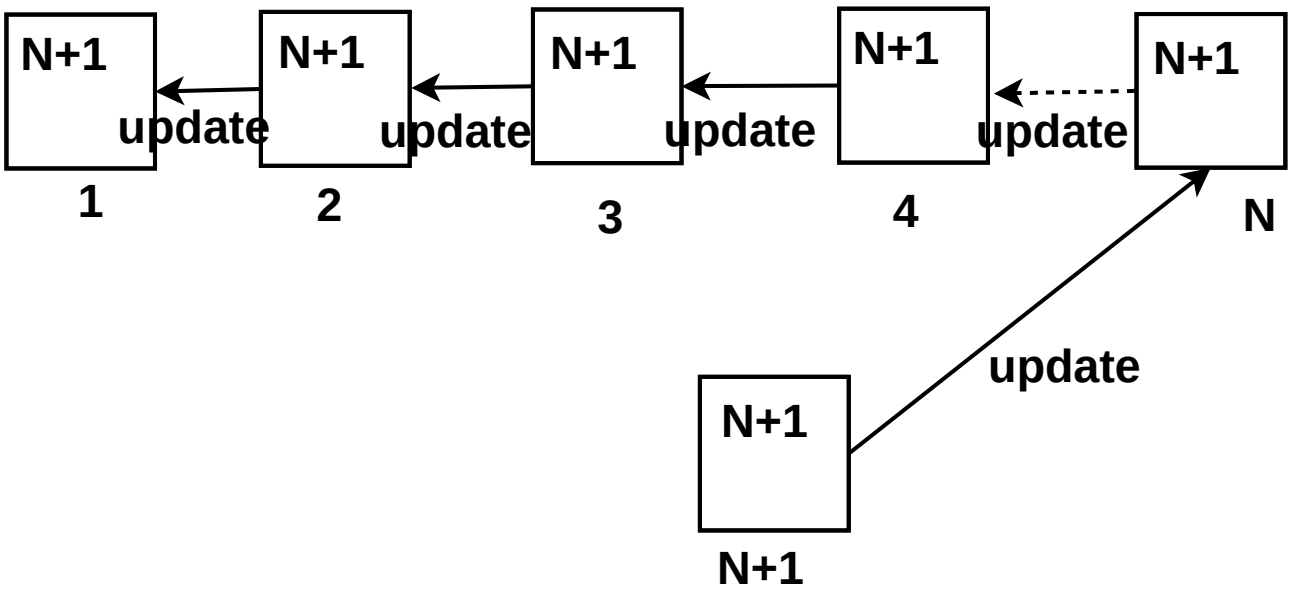


Έχουμε το δίκτυο N κόμβων και ένας νέος κόμβος (new node) επιθυμεί να συνδεθεί. Έχουμε θέσει στον τελευταίο κόμβο του δικτύου, δηλαδή στο σχήμα τον κόμβο N, να στέλνει τις προσκλήσεις ανά τακτά χρονικά διαστήματα. Θα μπορούσαν να στέλνουν και άλλοι κόμβοι, όμως αυτό θα περιέπλεκε το δίκτυο, ειδικά στις περιπτώσεις όπου ο κόμβος άκουγε προσκλήσεις από δύο ή περισσότερους ήδη συνδεδεμένους κόμβους. Ο νέος κόμβος, μόλις ανάψει, η μόνη διαδικασία που εκτελεί είναι περιμένει να ακούσει τέτοια πρόσκληση.

Το πακέτο της πρόσκλησης περιέχει και την πληροφορία του πλήθους των υπαρχόντων κόμβων του δικτύου. Μόλις το ακούσει και το λάβει ο νέος κόμβος, χρησιμοποιεί την πληροφορία ότι υπάρχουν N κόμβοι στο δίκτυο. Προκειμένου να υπάρχει συνέχεια ως προς την αρίθμηση και, ακολούθως, ως προς τις διευθύνσεις και να διατηρείται σωστή η δομή του δικτύου, ο νέος κόμβος θα θέσει τη διεύθυνσή του ίση με N+1. Θα έχουμε, άρα το επόμενο δίκτυο.



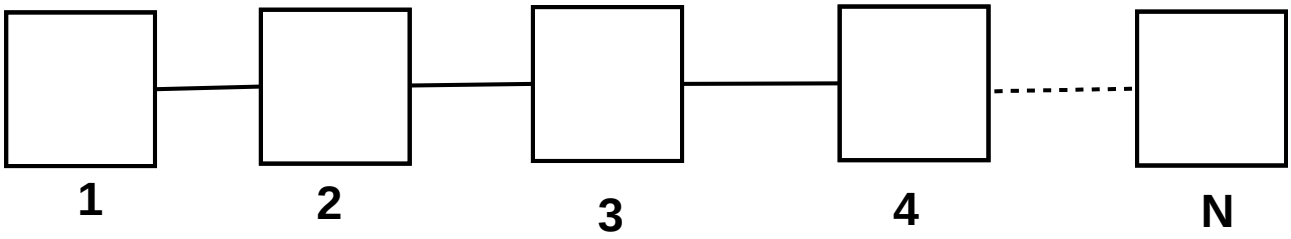
Προκειμένου να ενημερωθεί το υπόλοιπο δίκτυο για τη σύνδεση του νέου κόμβου N+1, αυτός στέλνει στον κόμβο 1 τη νέα αυτή πληροφορία. Επειδή οτιδήποτε στέλνεται περνάει αναγκαστικά από όλους τους ενδιάμεσους κόμβους, αυτό το μήνυμα θα το διαβάσουν όλοι. Αφού το διαβάσουν, πρώτα ανανεώνουν τις γνώσεις τους με την πληροφορία ότι τώρα υπάρχουν N+1 κόμβοι συνολικά και, μετά, προωθούν το μήνυμα. Στη συνέχεια δείχνουμε σε σχήμα αυτήν την διαδικασία, όπου μέσα στα κουτιά είναι η πληροφορία που έχει κάθε κόμβος για το πλήθος του δικτύου.



Αυτή είναι η συνολική διαδικασία που ακολουθείται όταν ένας νέος κόμβος θέλει να συνδεθεί στο υπάρχον δίκτυο, και η τελική κατάσταση είναι ένα δίκτυο με N+1 σειριακά συνδεδεμένους κόμβους.

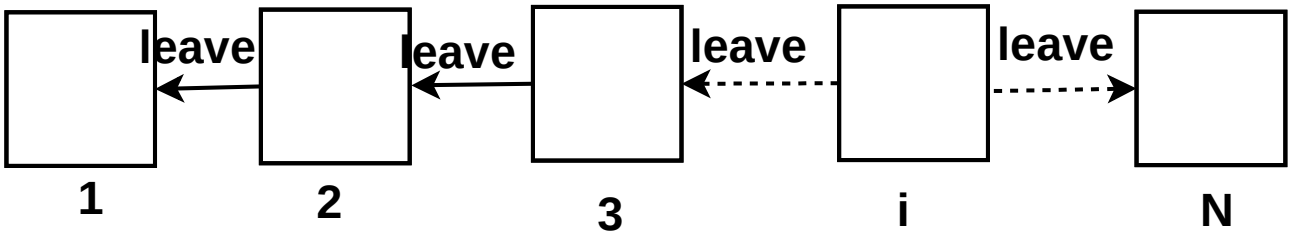
5.1.3 Αναχώρηση κόμβου από το δίκτυο

Ας επιστρέψουμε στο αρχικό δίκτυο με N κόμβους.

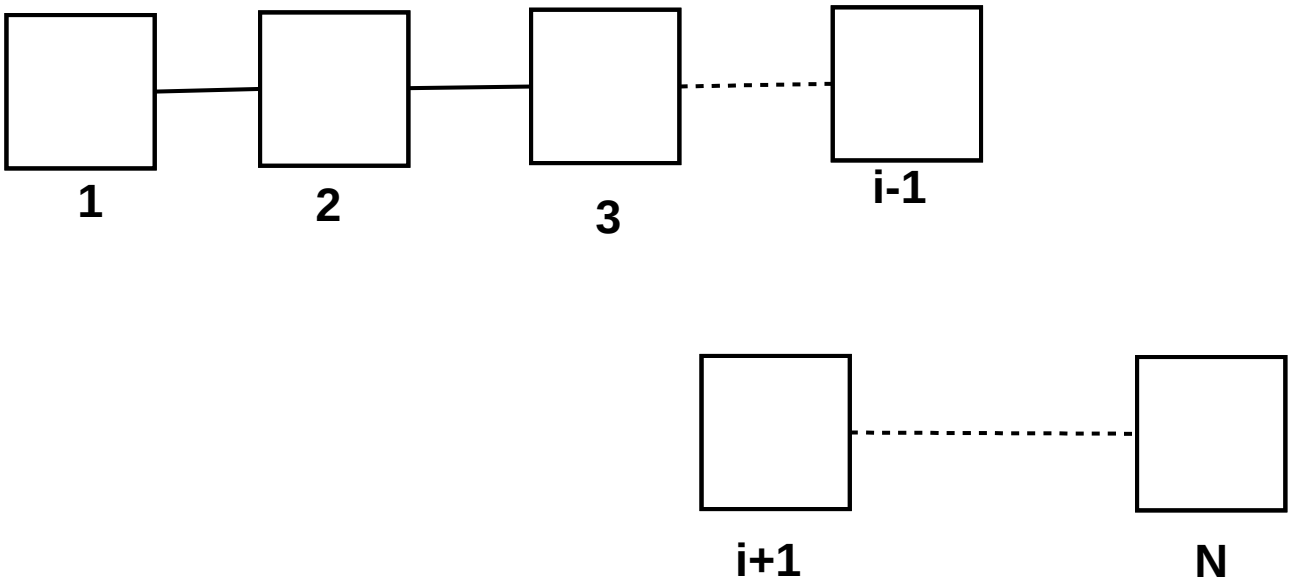


Έστω ότι τώρα ένας κόμβος i θέλει να φύγει από το δίκτυο. Σημαντικό είναι εδώ να πούμε ότι, σε αυτό το σημείο, καλύπτουμε την περίπτωση όπου ο κόμβος θέλει να φύγει και πρόκειται να ενημερώσει το υπόλοιπο δίκτυο. Υπάρχει και η περίπτωση διακοπής λειτουργίας του κόμβου λόγω βλάβης ή άλλου (πχ. κλοπή), της οποίας η αντιμετώπιση εξηγείται στη συνέχεια.

Ο i θα ενημερώσει τα υποδίκτυα που υπάρχουν αριστερά και δεξιά του, στέλνοντας τους τη πληροφορία ότι αυτός θα φύγει και, επίσης, σε κάθε κόμβο θα πει αν βρίσκεται αριστερά του ή δεξιά του (εξηγείται στα επόμενα ο λόγος αυτής της πληροφορίας). Σχηματικά, θα έχουμε το επόμενο.



Αφού φύγει ο κόμβος i, αν δεν ενημέρωνε το δίκτυο, θα προέκυπτε το επόμενο σχήμα.



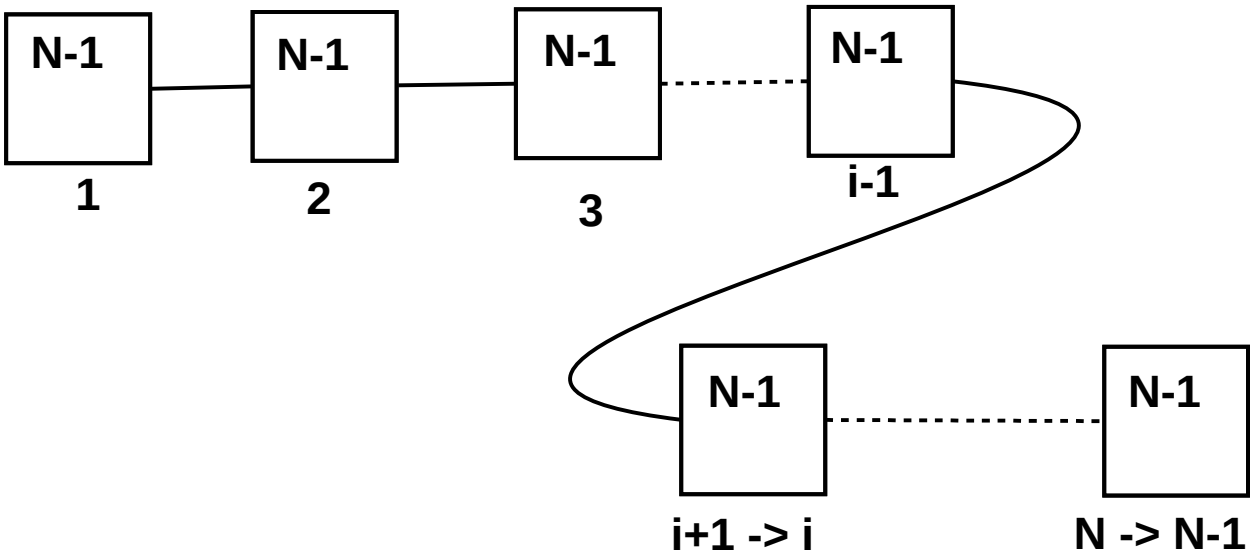
Παρατηρούμε ότι τα δύο υποδίκτυα, από 1 έως i-1 και από i+1 έως N δε μπορούν να επικοινωνήσουν μεταξύ τους, αφού δε γνωρίζουν που θα πρέπει να στείλουν το μήνυμα για να πάει από το ένα υποδίκτυο στο άλλο.

Όπως αναφέρθηκε, βασικό χαρακτηριστικό της δομής που χρησιμοποιούμε είναι ότι οι γειτονικοί κόμβοι θα έχουν διαδοχικές διευθύνσεις. Επομένως, προκειμένου να διατηρηθεί αυτή η ιδιότητα, ο κόμβος i, πριν φύγει, θα στείλει δύο μηνύματα, ένα αριστερά του, με τελικό παραλήπτη τον κόμβο με διεύθυνση 1, και ένα δεξιά του, με τελικό παραλήπτη τον κόμβο με διεύθυνση N.

Οι κόμβοι αριστερά του i θα ενημερωθούν ότι φεύγει ένας κόμβος και, προκειμένου να διατηρηθεί η σωστή δομή του δικτύου, αρκεί απλά να ενημερώσουν τη πληροφορία που έχουν για το πλήθος του δικτύου, μειώνοντάς το κατά ένα.

Όμως, οι κόμβοι δεξιά του δικτύου δεν αρκεί να εκτελέσουν αυτή μόνο την αλλαγή, διότι τότε θα παραβιαστεί ο κανονισμός για τη διαδοχικότητα των διευθύνσεων και μετά τον κόμβο με διεύθυνση i-1 θα ακολουθεί ο κόμβος i+1, ο οποίος διαφέρει κατά 2 από τον i-1. Για να λυθεί αυτό, οι κόμβοι δεξιά του i, εκτός από τη μείωση της πληροφορίας του πλήθους του δικτύου κατά ένα, θα μειώσουν και τη δική τους διεύθυνση κατά ένα.

Το τελικό δίκτυο, μετά την αναχώρηση του i, φαίνεται στο επόμενο σχήμα.



Αν, τώρα, ο κόμβος i φύγει χωρίς να ενημερώσει το δίκτυο, τότε αρχικά δεν γίνεται τίποτα. Μόλις ένας από τους δύο πρώην γειτονικούς κόμβους αυτού που έφυγε προσπαθήσει να στείλει κάποιο μήνυμα στη διεύθυνση του i, η αποστολή θα αποτύχει. Θεωρούμε ότι, όταν η αποστολή αποτυγχάνει, το μόνο σενάριο είναι ότι ο κόμβος έφυγε χωρίς να ενημερώσει το υπόλοιπο δίκτυο. Ο κόμβος που αντιλήφθηκε την αλλαγή αυτή είναι και τώρα ο υπεύθυνος για την ενημέρωση του δικτύου. Προσομοιώνει την αναχώρηση του i στέλνοντας τα αντίστοιχα μηνύματα ενημέρωσης. Το τελικό δίκτυο που προκύπτει είναι το ίδιο με το προηγούμενο.

5.2 Περιγραφή Αλγορίθμου για το υλοποιημένο δίκτυο

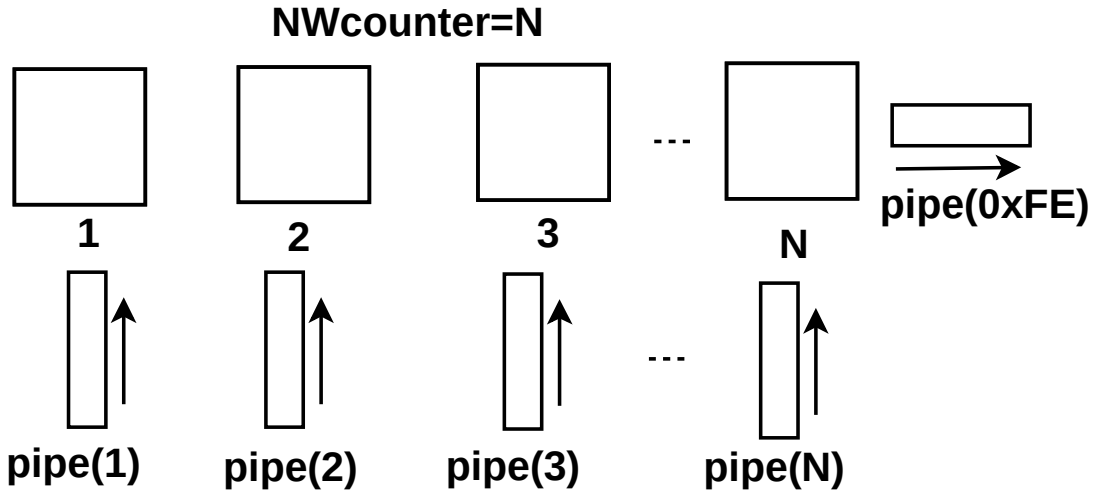
Σε αυτό το σημείο, αφού έχουν εξηγηθεί όλες οι απαραίτητες δομές και λειτουργίες του δικτύου, θα αναλύσουμε τον αλγόριθμο για την αυτοοργάνωση των κόμβων του τοπικού αυτού δικτύου. Τα επόμενα αποτελούν μία επέκταση της δομής του δικτύου που αναφέρθηκε, με περισσότερες λεπτομέρειες, μέσω των οποίων εξηγείται πώς ακριβώς αντιμετωπίζεται η κάθε διαδικασία ώστε να διατηρείται το δίκτυο στην επιθυμητή κατάσταση. Ο αλγόριθμος μπορεί να βρεθεί στο [77], και στο σχετικό παράρτημα. Όπως αναφέρθηκε, για συσκευή βασισμένη σε μικροελεγκτή χρησιμοποιήθηκε το Arduino και για συσκευή πομποδέκτη χρησιμοποιήθηκε η nRF24L01.

5.2.1 Σταθερή λειτουργία του δικτύου

Κάθε κόμβος έχει τη δική του μοναδική διεύθυνση όταν βρίσκεται μέσα στο δίκτυο. Αυτό αντιστοιχίζεται στην εφαρμογή μας με τα pipes, στα οποία βασίζεται η επικοινωνία με τους πομποδέκτες με ονομασία nRF24L01 που χρησιμοποιούμε. Δηλαδή, κάθε κόμβος θα έχει δικό του μοναδικό reading pipe με τη διεύθυνση που θα του έχει ανατεθεί (συμβολισμένη selfad). Οποιοσδήποτε άλλος κόμβος που θέλει να αναφερθεί στον κόμβο αυτόν, θα ανοίγει writing pipe με τη διεύθυνση του και θα γράφει το μήνυμα που θέλει να στείλει. Pipes, όπως το λέει και η ονομασία τους, είναι "σωλήνες" στους οποίους κάποιοι μπορούν να βάλουν δεδομένα στη μία άκρη και άλλοι μπορούν διαβάζουν τα δεδομένα από την άλλη άκρη. Στη δική μας περίπτωση, κάθε κόμβος θα έχει έναν τέτοιο σωλήνα από τον οποίον θα διαβάζει από τη μία άκρη (reading pipe) και αν κάποιος θέλει να στείλει μήνυμα στον κόμβο αυτόν, θα γράφει στην άλλη άκρη του σωλήνα (writing pipe).

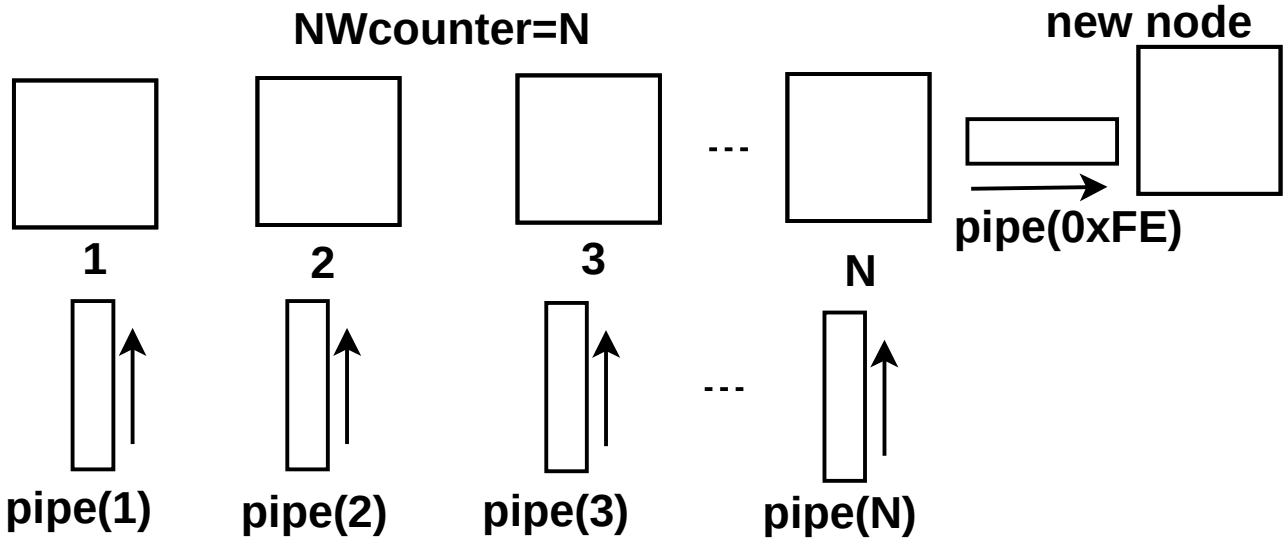
Όταν ένας νέος κόμβος ενεργοποιείται, όπως αναφέρθηκε προηγουμένως, θα ακούει για προσκλήσεις (invitations) για ένα σύντομο χρονικό διάστημα (5 sec, τιμή η οποία μπορεί να ρυθμιστεί εύκολα στον κώδικα του προγράμματος). Όταν λέμε ότι "ακούει" για προσκλήσεις, γενικά, έχουμε θέσει τη διεύθυνση 0xFE ως διεύθυνση για broadcasts. Επομένως, όταν ενεργοποιείται ένας νέος κόμβος, ανοίγει και reading pipe στη διεύθυνση 0xFE.

Σχηματικά, όταν το δίκτυο βρίσκεται σε κανονική λειτουργία, θα έχουμε το επόμενο. Τα βέλη δείχνουν την κατεύθυνση των δεδομένων των pipes, δηλαδή, στο σχήμα, τα pipes 1 έως και N είναι για διάβασμα (reading pipes) για τους κόμβους στους οποίους έχουν ανατεθεί, και για αυτόν το λόγο τα αντίστοιχα βέλη δείχνουν προς αυτούς. Στο pipe 0xFE γράφει ο τελευταίος κόμβος τα μηνύματα για προσκλήσεις, και για αυτό το βέλος αρχίζει από τον κόμβο αυτόν.

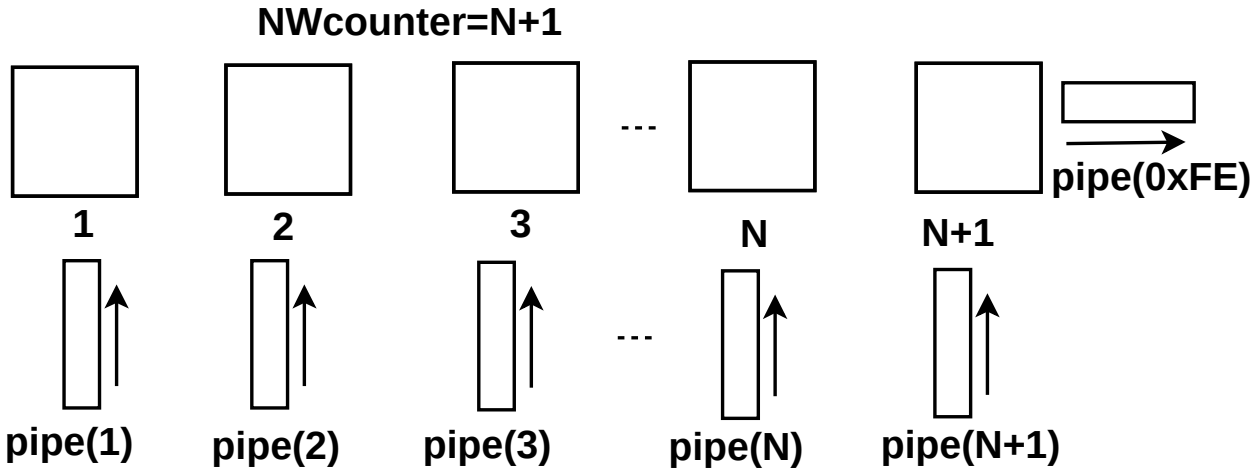


5.2.2 Διαδικασία εισαγωγής νέου κόμβου στο δίκτυο

Κάθε κόμβος έχει στη μνήμη του την πληροφορία του πλήθους κόμβων του δικτύου (Nwcounter = N). Κάθε κόμβος έχει δικό του ανοιχτό reading pipe και περιμένει να "ακούσει" για μηνύματα. Ο τελευταίος κόμβος N έχει ανοιχτό και ένα writing pipe στη διεύθυνση για broadcasts (0xFE), και στέλνει ανά τακτά χρονικά διαστήματα προσκλήσεις προς νέους κόμβους που θέλουν να συνδεθούν στο δίκτυο. Συνολικά, όταν ένας νέος κόμβος θέλει να συνδεθεί, θα έχουμε το επόμενο σχήμα.

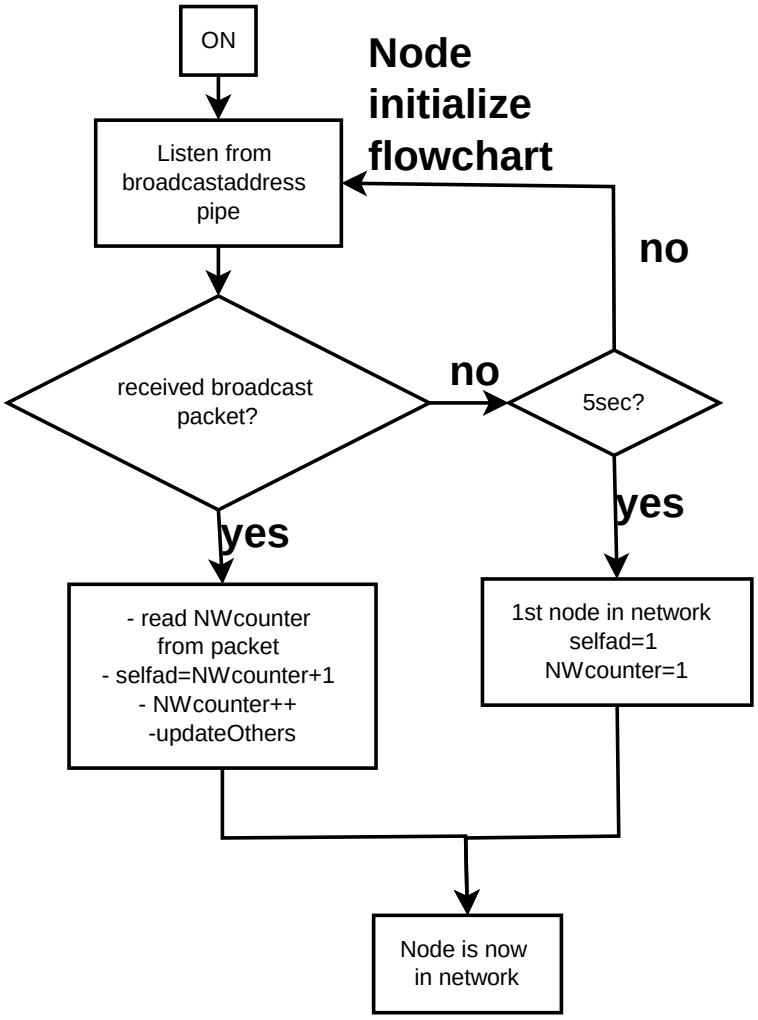


Ο νέος κόμβος έχει ανοίξει reading pipe στη διεύθυνση για broadcasts, στην οποία ο κόμβος N πρόκειται να στείλει πρόσκληση. Αυτό φαίνεται και στο σχήμα, στο οποίο το βέλος του pipe 0xFE αρχίζει από τον κόμβο N και καταλήγει στο νέο κόμβο. Ο νέος κόμβος (i) διαβάζει από το pipe το μήνυμα πρόσκλησης, (ii) παίρνει την πληροφορία του πλήθους του δικτύου, την οποία ονομάζουμε Nwcounter (network counter) και είναι ίση με N στο συγκεκριμένο σχήμα, (iii) την αυξάνει κατά ένα, (iv) θέτει τη δική του διεύθυνση ίση με τη νέα τιμή Nwcounter και (v) ανανεώνει τους υπόλοιπους κόμβους με τη νέα αυτή πληροφορία. Οπότε, το τελικό σχήμα θα έχει ως εξής, όπου θα είναι γνωστό σε όλους τους κόμβους ότι NWcounter=N+1, κάθε κόμβος έχει ανοιχτό reading pipe που του αντιστοιχεί και ο νέος και, πλέον, τελευταίος κόμβος έχει ανοιχτό το writing pipe για προσκλήσεις.



5.2.3 Διάγραμμα ροής αρχικοποίησης νέου κόμβου

Έχοντας εξηγήσει τα βασικά χαρακτηριστικά των κόμβων, στη συνέχεια εξηγούμε τον αλγόριθμο αρχικοποίησης ενός νέου κόμβου, δηλαδή τις διαδικασίες που θα ακολουθήσει μόλις ανάψει. Δίνεται το διάγραμμα ροής του.

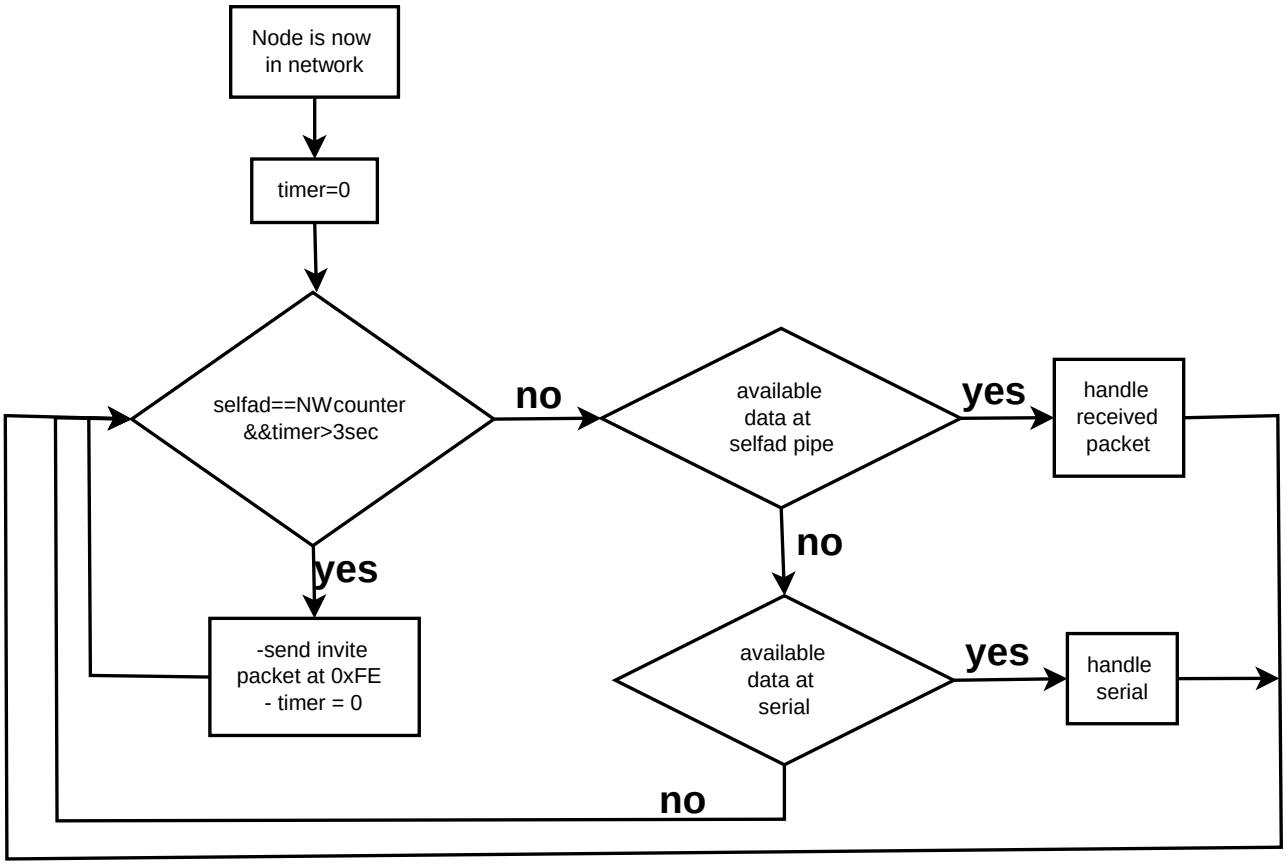


Έστω ένας νέος κόμβος μόλις ενεργοποιήθηκε. Προκειμένου να συνδεθεί με το ήδη υπάρχον δίκτυο, αν υπάρχει, θα ανοίξει reading pipe στη διεύθυνση 0xFE και θα διαβάζει

από εκεί. Αν δε διαβάσει τίποτα για 5 δευτερόλεπτα (η τιμή αυτή μπορεί να αλλάξει αν χρειαστεί), τότε θεωρεί ότι δεν υπάρχει άλλος κόμβος που να στέλνει προσκλήσεις, οπότε είναι ο πρώτος κόμβος του δικτύου. Αρχικοποιεί, τότε, το δίκτυο, ανοίγοντας reading pipe στη διεύθυνση 1 και θέτοντας NWcounter=1. Αν, αντίθετα, διαβάσει κάτι από το pipe 0xFE και είναι όντως σωστό πακέτο πρόσκλησης, τότε παίρνει από το πακέτο αυτό την πληροφορία του πλήθους κόμβων του ήδη υπάρχοντος δικτύου, το αυξάνει κατά ένα, ανοίγει reading pipe στη διεύθυνση ίση με το νέο πλήθος του δικτύου και ανανεώνει τους άλλους κόμβους. Στη συνέχεια, και στις δύο περιπτώσεις, ο κόμβος βρίσκεται μέσα στο δίκτυο.

5.2.4 Διάγραμμα ροής κόμβου που βρίσκεται στο δίκτυο

Στη συνέχεια έχουμε το διάγραμμα ροής του κόμβου που βρίσκεται μέσα στο δίκτυο. Θυμίζουμε ότι δεν αποτελεί αλγόριθμο με τέλος, αφού ο κόμβος βρίσκεται σε συνεχή λειτουργία και πρέπει να εκτελεί συνέχεια εντολές και να διαβάζει μηνύματα που του στέλνονται, εκτός αν δοθεί εντολή να απενεργοποιηθεί, διαδικασία που θα δούμε στα επόμενα. Οπότε, και στο διάγραμμα ροής, δε θα υπάρχει κουτί "τέλους".



Στην αρχή, τίθεται ένα χρονόμετρο σε τιμή μηδέν. Σκοπός του είναι για τον τελευταίο μόνο κόμβο, ο οποίος πρόκειται να στέλνει προσκλήσεις ανά 3 δευτερόλεπτα και θα ξαναμηδενίζει κάθε φορά το timer. Σκοπός της πρώτης συνθήκης "selfad==NWcounter && " timer>3sec" είναι ακριβώς αυτός. Ο κόμβος για τον οποίον ισχύει η πρώτη ισότητα είναι ο τελευταίος, αφού αυτός έχει διεύθυνση ίση με το πλήθος όλων των κόμβων. Οπότε, αν έχουν περάσει 3 δευτερόλεπτα, τότε στέλνει πρόσκληση στη διεύθυνση για broadcasts και μηδενίζει το timer, ώστε να στείλει πάλι μετά από 3sec. Η τιμή 3 sec είναι ρυθμίσιμη παράμετρος και μπορεί να αλλάξει ανάλογα με τις ανάγκες της εφαρμογής. Πρέπει, όμως,

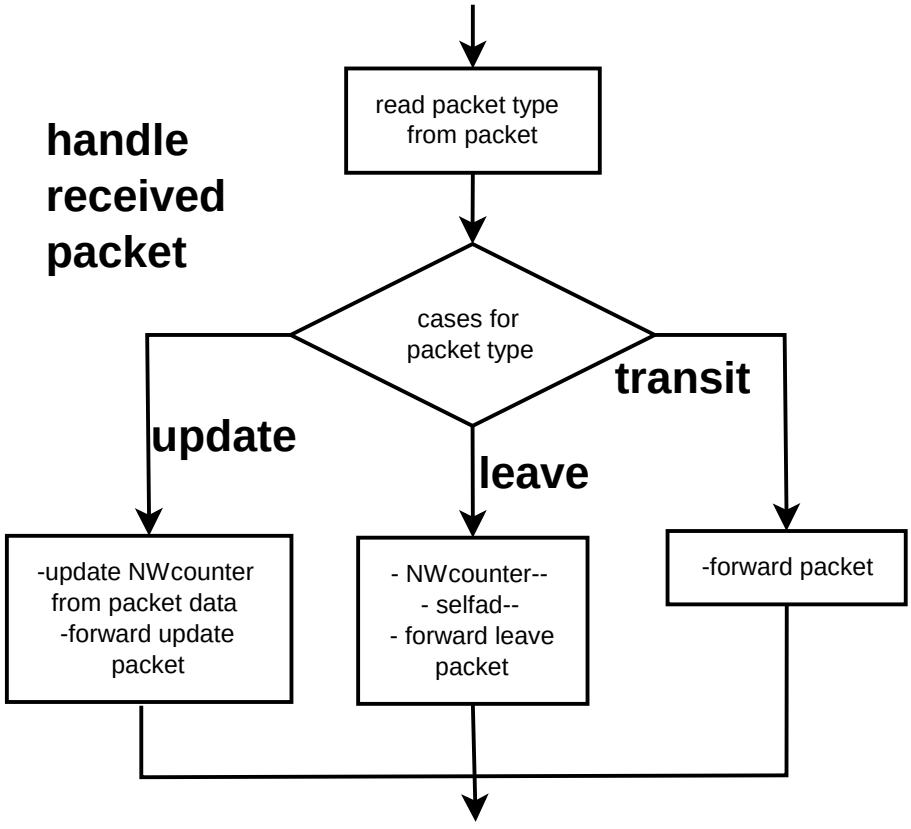
πάντα να είναι μικρότερη από το χρόνο που ο νέος κόμβος περιμένει να ακούσει για πρόσκληση, επειδή αλλιώς υπάρχει η πιθανότητα να μην προλάβει να σταλθεί η πρόσκληση και να νομίζει ο νέος κόμβος ότι δεν υπάρχει δίκτυο.

Αν η συνθήκη αυτή είναι ψευδής, τότε είτε είμαστε σε κόμβο που δεν είναι ο τελευταίος, είτε αν είναι ο τελευταίος δεν έχουν περάσει 3 sec από την τελευταία πρόσκληση. Σε αυτήν την περίπτωση, ο κάθε κόμβος του δικτύου θα εκτελέσει δύο λειτουργίες.

- Πρώτα θα ελέγξει το reading pipe του. Αν υπάρχουν δεδομένα διαθέσιμα για διάβασμα, τα διαβάζει και χειρίζεται το ληφθέν πακέτο. Η διαδικασία αυτή εξηγείται σε διάγραμμα ροής στα επόμενα.
- Στη συνέχεια, αν δεν υπάρχει τίποτα στο reading pipe, ελέγχει αν υπάρχουν δεδομένα στο serial line. Έχουμε προσθέσει ένα πολύ βασικό interface, προκειμένου να μπορεί ο χρήστης, αν επιθυμεί, να στείλει δικά του πακέτα και να ενημερώσει από μόνος του το δίκτυο. Το serial line, στην περίπτωσή μας, αποτελεί απλά τη θύρα usb με την οποία συνδέονται τα arduino στον υπολογιστή, και δεδομένα στο serial έρχονται από το πληκτρολόγιο. Αν, λοιπόν, υπάρχει κάτι στο serial, το διαβάζει και το χειρίζεται. Περισσότερες λεπτομέρειες σε διάγραμμα ροής στη συνέχεια.

Αυτή είναι η λειτουργία που ακολουθεί η κάθε συσκευή όσο βρίσκεται μέσα στο δίκτυο σε φυσιολογική κατάσταση. Στη συνέχεια εξηγούμε με διάγραμμα ροής τον τρόπο με τον οποίο χειρίζεται η καθεμιά τα πακέτα που λαμβάνει στο pipe της.

5.2.4.1 Χειρισμός ληφθέντος πακέτου από το ασύρματο



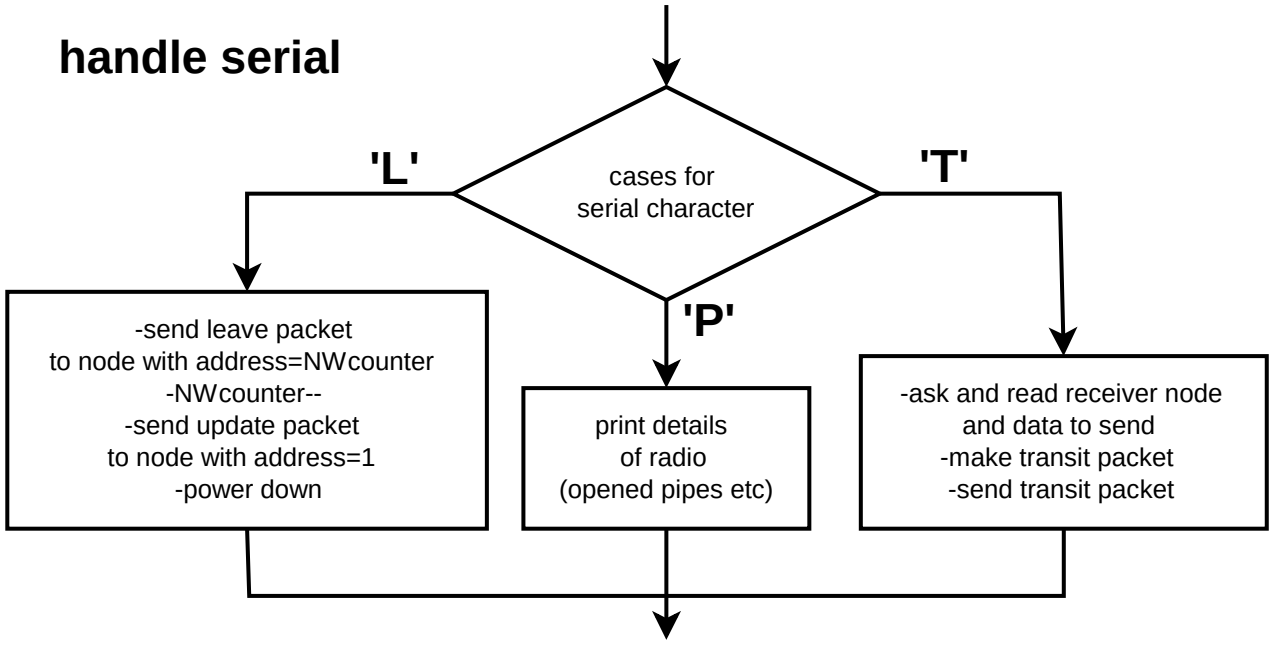
Αφού ο κόμβος λάβει το πακέτο, στην αρχή διαβάζει τον τύπο του πακέτου, προκειμένου να δράσει αναλόγως. Υπάρχουν τρεις κατηγορίες πακέτων.

- “update”: Σε αυτήν την περίπτωση, έχει ληφθεί πακέτο με νέα δεδομένα για ανανέωση των δεδομένων που είναι αποθηκευμένα στον κόμβο. Πιο συγκεκριμένα, η μόνη πληροφορία που έχει και που χρειάζεται να έχει κάθε κόμβος για το υπόλοιπο δίκτυο είναι το συνολικό πλήθος κόμβων. Οπότε ανανεώνει αυτό, από τα περιεχόμενα του πακέτου που έλαβε, και στη συνέχεια προωθεί το πακέτο στον επόμενο κόμβο, αν δεν έχει φτάσει το πακέτο στον τελικό προορισμό του. Επόμενος κόμβος μπορεί να είναι είτε ο δεξής του είτε ο αριστερός, ανάλογα με τη διεύθυνση του αρχικού αποστολέα κόμβου.
- “leave”: Σε αυτήν την περίπτωση, κάποιος κόμβος έφυγε από το δίκτυο, και πρέπει να αλλαχθούν οι διευθύνσεις κάποιων κόμβων, όπως παρουσιάστηκε στο προηγούμενο υποκεφάλαιο. Ο λόγος που αλλάζουν αυτές οι διευθύνσεις είναι ότι ο αλγόριθμος απαιτεί να υπάρχει αριθμητική συνέχεια των διευθύνσεων σε διαδοχικούς κόμβους του δικτύου. Τέτοιο πακέτο στέλνεται πάντα προς τα δεξιά από τον κόμβο που έφυγε, οπότε όλοι οι κόμβοι που θα το λάβουν θα μειώσουν τη διεύθυνσή τους κατά ένα και θα μειώσουν το Nwcounter κατά ένα. Ποια ακριβώς είναι η διαδικασία που ακολουθείται όταν ένας κόμβος φεύγει εξηγείται στα επόμενα.
- “transit”: Τέτοιο πακέτο είναι απλά για μεταφορά πληροφορίας που προέρχεται από τον χρήστη, μέσω της σειριακής γραμμής (serial line), δηλαδή του πληκτρολογίου. Δεν απαιτείται κάποια άλλη ενέργεια πέραν της προώθησης του πακέτου προς τον τελικό παραλήπτη.

Κάθε πακέτα περιέχει και τη διεύθυνση του τελικού παραλήπτη, οπότε, αν πρόκειται να γίνει προώθηση, ελέγχεται πρώτα μήπως έχει φτάσει στον τελικό προορισμό, και αν ναι τότε σταματάει.

5.2.4.2 Χειρισμός σειριακής σύνδεσης με τον υπολογιστή

Στο επόμενο διάγραμμα ροής παρουσιάζεται η διαδικασία για το χειρισμό δεδομένων που έρχονται από το serial line, δηλαδή από το πληκτρολόγιο.

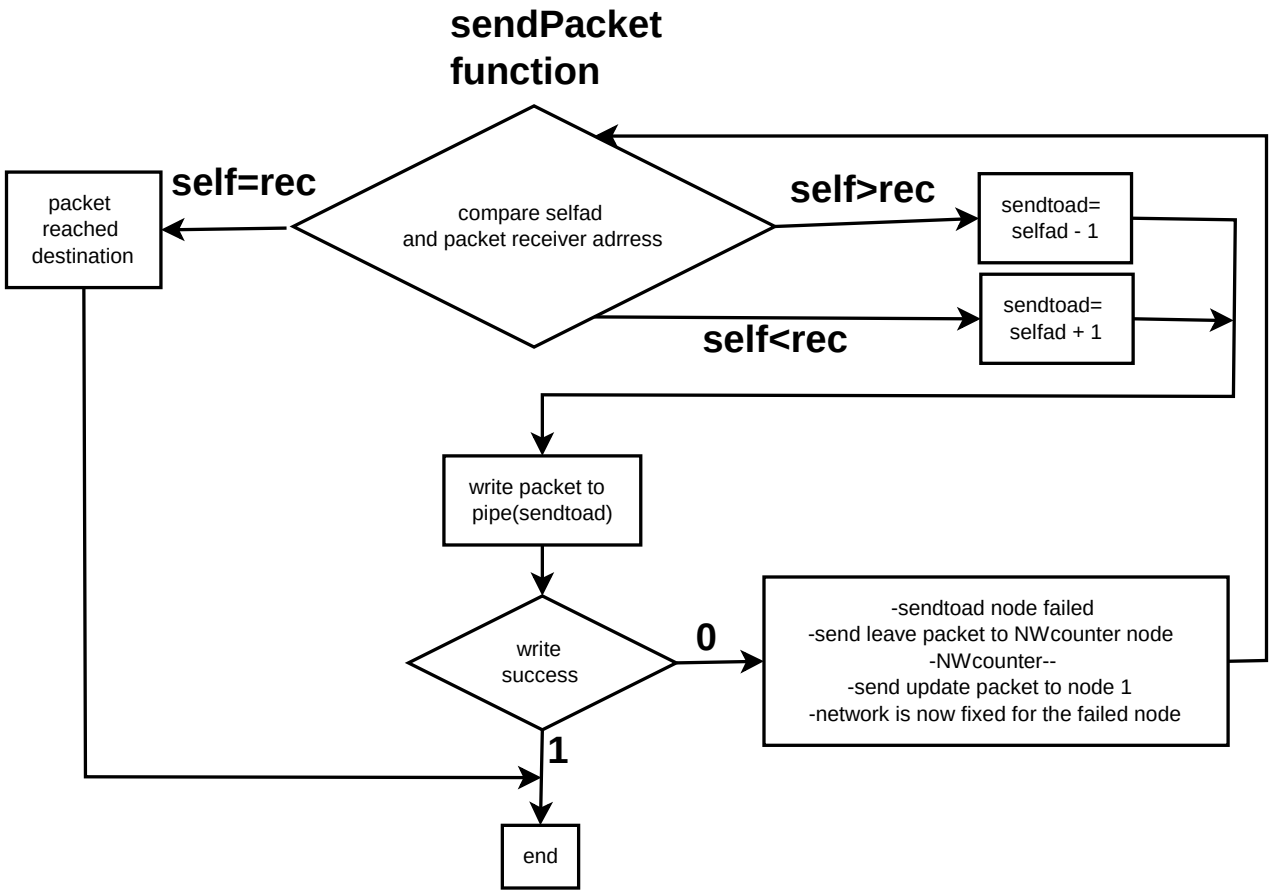


Έχουμε ένα απλό interface, όπου ο χρήστης πληκτρολογεί ένα γράμμα και αναλόγως εκτελείται μία διαδικασία. Οι δυνατές διαδικασίες είναι οι εξής:

- - 'L'/Leave: Δείχνει ο χρήστης την επιθυμία του να φύγει ο κόμβος. Αρχικά φτιάχνει ένα πακέτο τύπου “leave” και το στέλνει στον τελευταίο κόμβο του δικτύου, δηλαδή σε αυτόν με διεύθυνση ίση με το πλήθος του δικτύου (Nwcounter). Στη συνέχεια, μειώνει κατά ένα το Nwcounter και στέλνει πακέτο τύπου “update” στον κόμβο 1. Στη συνέχεια, μπορεί να απενεργοποιηθεί και να αποσυνδεθεί από το δίκτυο. Σημαντικό είναι εδώ να σημειωθεί ότι, όπως έχει αναφερθεί, όταν στέλνεται ένα πακέτο από ένα κόμβο σε έναν άλλον, για να φτάσει στον τελικό προορισμό, θα περαστεί από όλους τους ενδιάμεσους κόμβους. Έτσι, στέλνοντας τα πακέτα “update” και “leave” στους κόμβους 1 και Nwcounter αντίστοιχα, θα περάσουν από όλους τους ενδιάμεσους. Αυτοί, με τη σειρά τους, θα εκτελέσουν όλες τις διαδικασίες που αναφέρθηκαν ανάλογα με τον τύπο πακέτου πριν το προωθήσουν. Έτσι, τελικά, θα έχουν ενημερωθεί όλοι οι κόμβοι για την αναχώρηση του κόμβου στον οποίον ο χρήστης έγραψε 'L'.
- 'P'/Print Details: Αποτελεί μία απλή διαδικασία, κατά την οποία εκτυπώνεται στο serial (δηλαδή στην οθόνη του υπολογιστή), προκειμένου να τα διαβάσει ο χρήστης, κάποια βασικά χαρακτηριστικά του πομποδέκτη, όπως ποια pipes έχει ανοιχτά και σε ποιες διευθύνσεις, ποια ρύθμιση έχει ως προς την κατανάλωση ισχύος για λειτουργία κλπ. Η πιο χρήσιμη πληροφορία είναι αυτή των pipes, επειδή κανείς μπορεί να κάνει debugging με ευκολία αν καλεί τη συνάρτηση αυτή σε πολλά σημεία της λειτουργίας του δικτύου.
- 'T'/Transit: Σε αυτήν την περίπτωση, ο χρήστης επιθυμεί να στείλει ένα πακέτο με πληροφορίες που αυτός εισάγει, χωρίς να χρειάζεται ο κάθε κόμβος που λαμβάνει και προωθεί το πακέτο αυτό να εκτελέσει κάποια διαδικασία εκτός από την προώθησή του, αν αυτή χρειάζεται (δηλαδή αν το πακέτο δεν έχει φτάσει στον τελικό προορισμό του).

5.2.4.3 Διάγραμμα ροής συνάρτησης για αποστολή πακέτου

Σε αυτό το σημείο, πρέπει να εξηγηθεί η συνάρτηση sendPacket που χρησιμοποιείται για την αποστολή πακέτων, κυρίως επειδή μέσου αυτής γίνεται ο χειρισμός κόμβων που έφυγαν από το δίκτυο χωρίς να ενημερώσουν τους υπόλοιπους κόμβους. Στη συνέχεια φαίνεται το διάγραμμα ροής της.



Η συνάρτηση αυτή καλείται από τον αρχικό κόμβο όταν πρόκειται να στείλει ένα πακέτο με κατεύθυνση τον επιθυμητό τελικό κόμβο. Όπως έχουμε πει, η αποστολή πακέτων γίνεται μέσω των ενδιάμεσων κόμβων, οπότε άμεσες αποστολές γίνονται μόνο σε διευθύνσεις που έχουν διαφορά 1 από τον κόμβο που πρόκειται να κάνει την αποστολή. Εξηγείται καλύτερα στη συνέχεια.

Αρχικά, ο κόμβος που έχει το πακέτο, πρέπει να δει προς τα ποια κατεύθυνση πρέπει να το στείλει, έτσι ώστε να πλησιάσει κατά ένα κόμβο προς την τελική διεύθυνση. Συγκρίνει τη προσωπική του διεύθυνση (selfad) με τη διεύθυνση του τελικού κόμβου, την οποία βρίσκει στο πακέτο που έχει λάβει (packet.receiverNode). Υπάρχουν τρεις περιπτώσεις που προκύπτουν από αυτήν την σύγκριση:

- Αν οι δύο διευθύνσεις είναι ίσες, τότε το πακέτο έφτασε στον τελικό προορισμό του, και η συνάρτηση απλά επιστρέφει. Εδώ μπορούν να προστεθούν λειτουργίες όπως να αποθηκεύεται κάπου το πακέτο ή όταν διαβάζεται από τον κόμβο αυτός να δρα αναλόγως κλπ.
- Αν ο τελικός κόμβος βρίσκεται δεξιά του κόμβου με το πακέτο (selfad<packet.receiverNode), τότε η διεύθυνση που πρέπει να στείλει το πακέτο είναι ίση με την προσωπική του, επαυξημένη κατά ένα (sendtoad=selfad+1, όπου sendtoad = send to address είναι η διεύθυνση που πρόκειται να γίνει η αποστολή).
- Αν ο τελικός κόμβος βρίσκεται αριστερά του κόμβου με το πακέτο (selfad>packet.receiverNode), τότε η διεύθυνση που πρέπει να στείλει το πακέτο είναι ίση με την προσωπική του μειωμένη κατά ένα (sendtoad=selfad+1).

Στη συνέχεια γράφει το πακέτο στο pipe με διεύθυνση ίση με sendtoad. Αν γίνει η εγγραφή με επιτυχία, σημαίνει ότι έγινε η λήψη σωστά, οπότε η συνάρτηση επιστρέφει.

Αν όμως δεν πετύχει η εγγραφή, αρχικά γίνονται αυτόματα μερικές ακόμα προσπάθειες, ο αριθμός των οποίων μπορεί να ρυθμιστεί μέσω συνάρτησης της βιβλιοθήκης του πομποδέκτη. Αν δεν υπάρξει καμία επιτυχία, τότε δεν έγινε η λήψη. Θεωρούμε ότι ο κόμβος με διεύθυνση sendtoad έχει σταματήσει τη λειτουργία χωρίς ενημέρωση, ή έφυγε εκτός εμβέλειας. Και στις δύο περιπτώσεις, *έχει φύγει από το δίκτυο χωρίς να έχει ενημερώσει τους υπόλοιπους κόμβους*, για να μπορέσουν αυτοί να προσαρμοστούν και αν υπάρχει συνέχεια στη δομή του δικτύου.

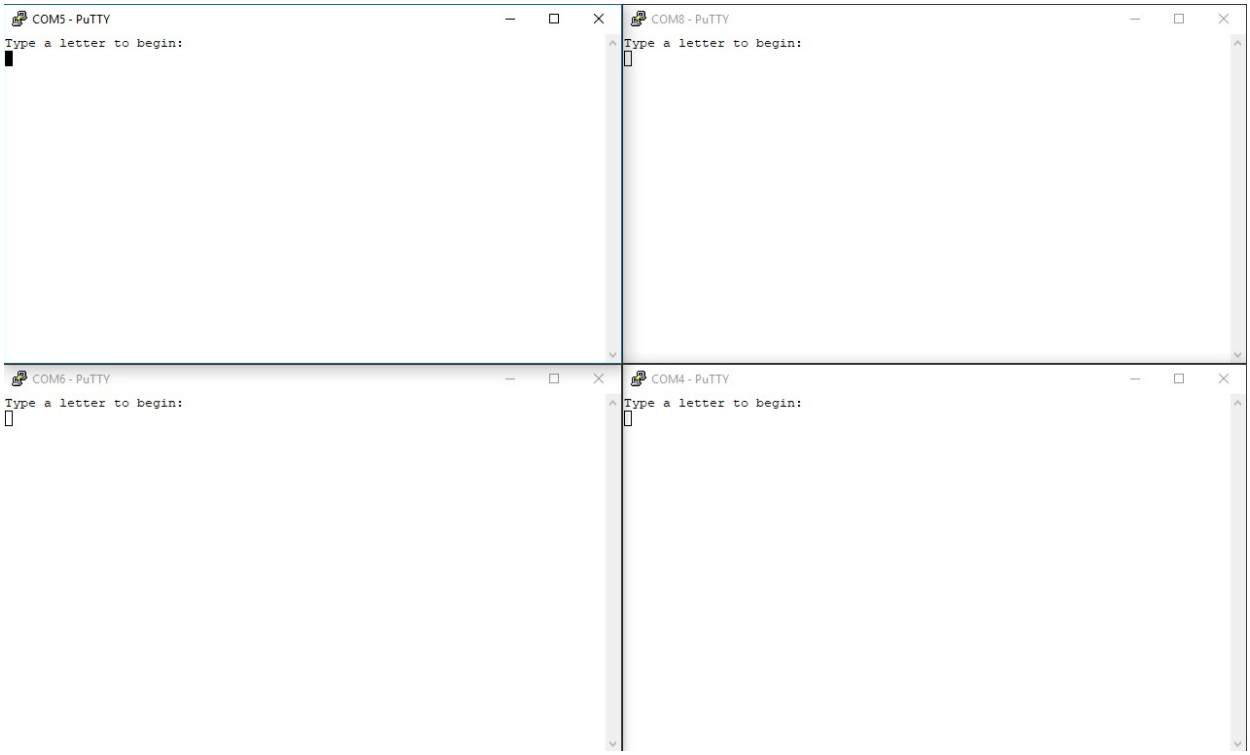
Τότε, πρέπει να γίνουν *όλες οι απαραίτητες αλλαγές για να επιστρέψει το δίκτυο στη σωστή κατάσταση*. Αυτό γίνεται, πρακτικά, ακολουθώντας τις ενέργειες που θα εκτελούσε ο κόμβος αν έκανε σωστά την ενημέρωση πριν φύγει, απλά τώρα τις εκτελεί ο κόμβος που προσπάθησε να του στείλει το πακέτο και έτσι κατάλαβε ότι ο κόμβος έχει φύγει. Δηλαδή, πρέπει να σταλθεί ένα πακέτο τύπου leave προς τα δεξιά στο δίκτυο από τη διεύθυνση του κόμβου που έφυγε, ώστε οι κόμβοι αυτοί να μειώσουν τη διεύθυνσή τους κατά ένα. Στη συνέχεια στέλνεται πακέτο τύπου update προς τα αριστερά από τον κόμβο που έφυγε, ώστε να ενημερωθούν και οι αριστεροί κόμβοι του δικτύου. Τελικά, όλοι οι κόμβοι γνωρίζουν ότι πλέον υπάρχει ένας λιγότερος κόμβος στο δίκτυο, και διατηρήθηκε η αριθμητική συνέχεια στις διευθύνσεις του δικτύου.

5.3 Αποτύπωση δοκιμαστικού τρεξίματος του αλγορίθμου

Στη συνέχεια παρουσιάζουμε τον αλγόριθμο σε λειτουργία σε δίκτυο με 4 συνολικά κόμβους, καθένας από τους οποίους αποτελείται, όπως έχει αναφερθεί, από ένα arduino (uno ή nano) και συνδεδεμένο στο καθένα έναν πομποδέκτη nRF24L01.

Κάθε κόμβος του δικτύου συνδέεται στον υπολογιστή με σύνδεση τύπου USB, με σκοπό να εκτυπώνει μηνύματα πληροφόρησης της κατάστασής του και να δέχεται είσοδο από το πληκτρολόγιο. Για τη σειριακή σύνδεση των κόμβων με τον υπολογιστή χρησιμοποιήθηκε το πρόγραμμα PuTTY, το οποίο επιτρέπει να πραγματοποιηθούν πολλών ειδών συνδέσεις, είτε ενσύρματες είτε ασύρματες. Η παρουσίαση αυτή γίνεται με χρήση στιγμιότυπων οθονών (screenshots) που έχουν παρθεί σε σημαντικά σημεία της λειτουργίας του δικτύου. Η οθόνη του υπολογιστή έχει χωριστεί σε τέσσερα ίσα μέρη και σε κάθε τέταρτο υπάρχει ένα παράθυρο PuTTY στο οποίο αποτυπώνει ο κάθε κόμβος αυτά που στέλνει μέσω της σειριακής σύνδεσης με τον υπολογιστή.

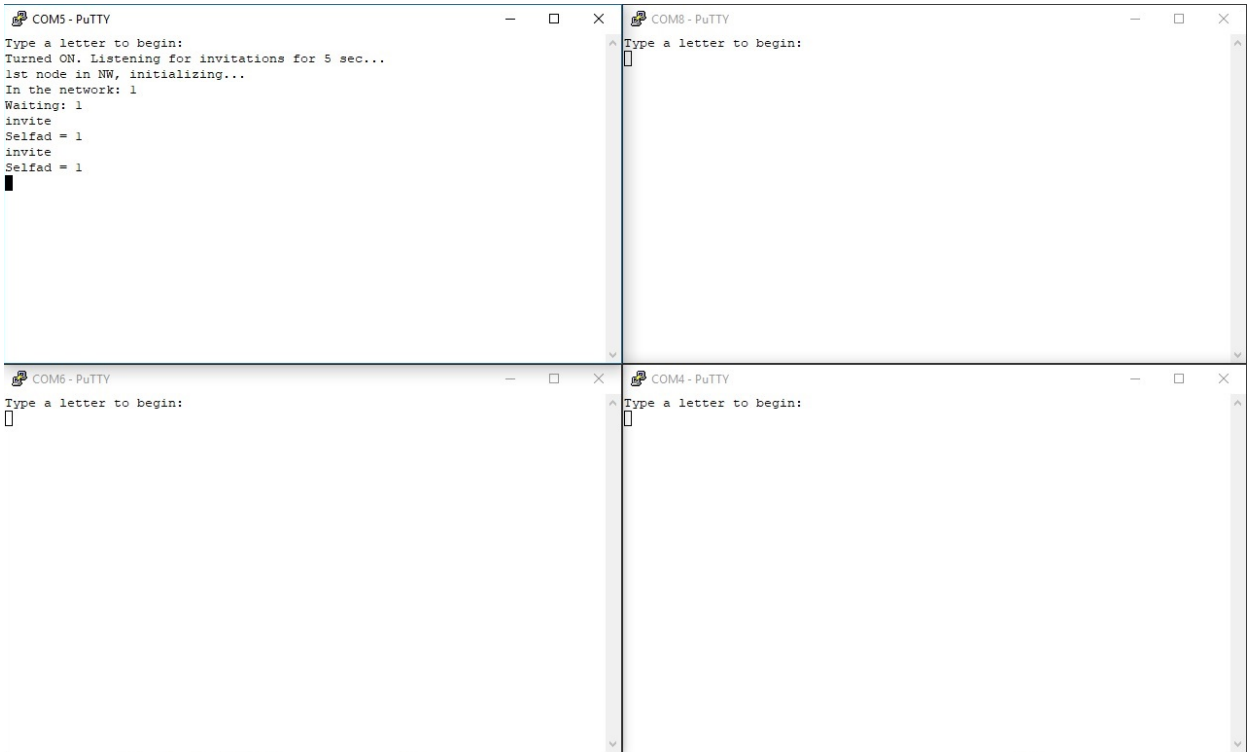
Γίνονται επομένως 4 συνδέσεις μέσω USB με τις συσκευές αυτές (μεταξύ τους δεν έχουν συνδεθεί ούτε έχουν δημιουργήσει κάποιο δίκτυο, ακόμα). Η κατάσταση, μόλις γίνουν οι συνδέσεις και ενεργοποιηθούν τα τέσσερα παράθυρα PuTTY, έχει ως εξής.



Έχει γίνει προγραμματισμός ώστε οι συσκευές στην αρχή να περιμένουν για την είσοδο ενός οποιουδήποτε πλήκτρου από το πληκτρολόγιο του υπολογιστή. Ο σκοπός αυτού είναι καθαρά για τον προγραμματισμό, τις δοκιμές και την παρουσίαση του δικτύου, ώστε να μπορεί να ελέγχεται ο χρονική στιγμή που θα ξεκινήσει τη λειτουργία της ως μέρος του τοπικού ασύρματου δικτύου η κάθε συσκευή, αντί να ξεκινάει αμέσως με τη σύνδεσή του με πηγή ρεύματος (δηλαδή με τον υπολογιστή στην περίπτωσή μας).

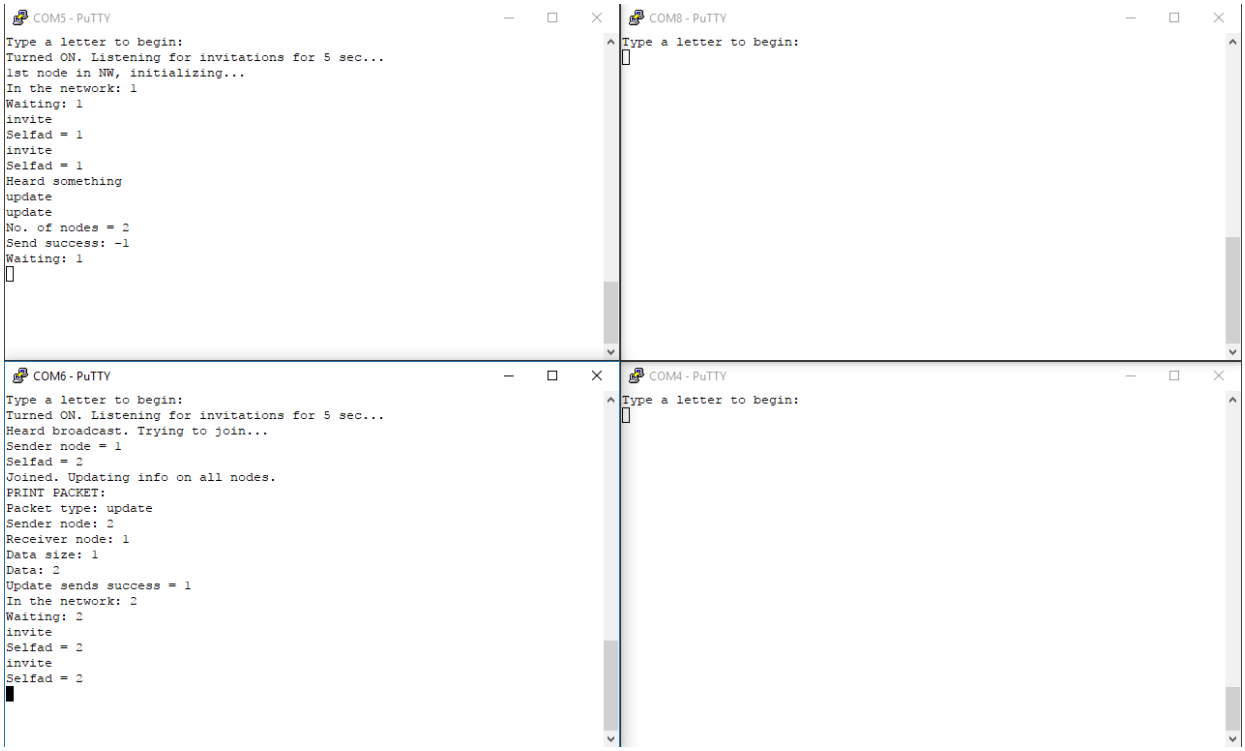
5.3.1 Σύνδεση κόμβων στο δίκτυο

Ξεκινάμε με το να πατήσουμε ένα τυχαίο πλήκτρο στον πρώτο κόμβο, και έχουμε τα εξής.



Ο πρώτος κόμβος ενεργοποιείται (αρχίζει τη λειτουργία του ως κόμβος που επιθυμεί να συνδεθεί στο αυτοοργανούμενο δίκτυο) και ακούει για προσκλήσεις για 5 δευτερόλεπτα. Αφού περάσει ο χρόνος αυτός, θεωρεί σωστά ότι δεν υπάρχει ήδη κάποιο δίκτυο, οπότε αρχικοποιεί τον εαυτό του ως κόμβο με διεύθυνση 1 και θέτει την τιμή του πλήθους κόμβων το δικτύου ίση με 1. Σημαντικό είναι, σε αυτό το σημείο, να υπενθυμιστεί ότι κάθε κόμβος του δικτύου έχει δική του τοπική μεταβλητή, στην οποία καταχωρείται η τιμή του συνολικού πλήθους των συνδεδεμένων κόμβων στο δίκτυο. Αυτή η μεταβλητή, σε κανονική λειτουργία του δικτύου, θα έχει ίδια τιμή σε κάθε κόμβο. Αυτό επιτυγχάνεται με ανανέωση του δικτύου κάθε φορά που συνδέεται ή αναχωρεί κόμβος στο/από το δίκτυο. Η διαδικασία γίνεται πιο κατανοητή από τα επόμενα screenshots. Στη συνέχεια, ο κόμβος 1 θεωρεί ότι είναι μέσα στο δίκτυο (που μόλις αρχικοποίησε) και γράφει ότι είναι έτοιμο και περιμένει για εντολές (Waiting: 1), είτε αυτές είναι από το ασύρματο είτε είναι από τη σειριακή σύνδεση (το πληκτρολόγιο). Εφόσον είναι ο τελευταίος κόμβος του δικτύου, είναι υπεύθυνος για να στέλνει προσκλήσεις κάθε 3 δευτερόλεπτα, με σκοπό τη σύνδεση νέων κόμβων στο δίκτυο. Κάθε φορά που στέλνει πρόσκληση, εκτυπώνει στην οθόνη του (δηλαδή στο παράθυρο PuTTY που του αντιστοιχεί) τα μηνύματα “invite” και “Selfad=1”, με σκοπό να δείξει τη διεύθυνσή του και ότι πραγματοποιείται η αποστολή πρόσκλησης, για καλύτερη παρουσίαση της λειτουργίας. Στην προηγούμενη εικόνα έχει στείλει 2 φορές προσκλήσεις (δηλαδή πέρασαν συνολικά 3+3=6 δευτερόλεπτα από τη στιγμή που μπήκε στο δίκτυο μέχρι τη στιγμή που καταγράφηκε το screenshot αυτό) και δεν έχει λάβει κάποια απάντηση, οπότε δεν έχει γίνει κάποια αλλαγή.

Αν, τώρα, πατήσουμε ένα τυχαίο πλήκτρο και στο δεύτερο κόμβο, αυτός επίσης θα ενεργοποιηθεί και θα ακούει για προσκλήσεις για 5 δευτερόλεπτα. Θα έχουμε το επόμενο.



Σε αυτήν την εικόνα, ο 2ος κόμβος άκουσε (δηλαδή έλαβε σχετικό πακέτο) την πρόσκληση από τον πρώτο. Είδε ότι αυτός που έστειλε την πρόσκληση έχει διεύθυνση 1, οπότε εκτελεί τις εξής διαδικασίες:

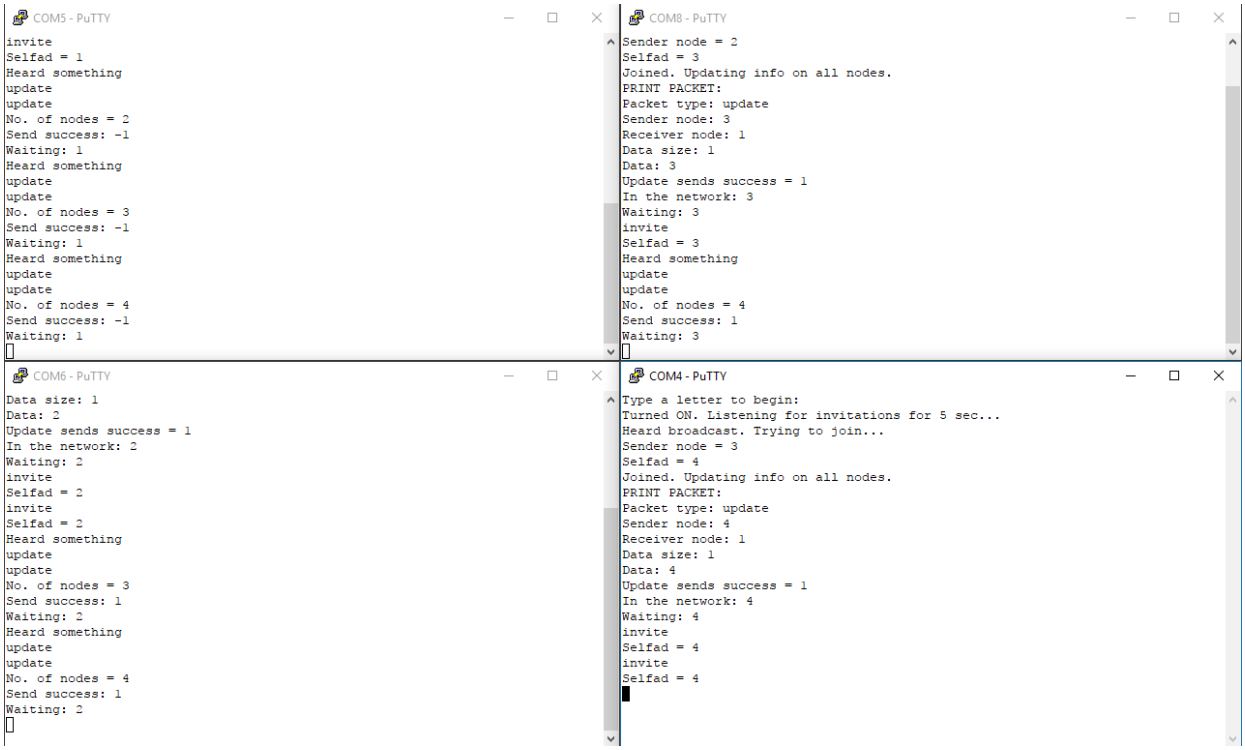
- Θέτει τη διεύθυνσή του ίση με 2.
- Θέτει την τιμή της μεταβλητής του πλήθους του δικτύου επίσης ίση με 2.
- Δημιουργεί πακέτο τύπου update και το στέλνει στο υπόλοιπο δίκτυο (το οποίο περιέχει μόνο τον κόμβο 1 στη συγκεκριμένη περίπτωση). Το πακέτο update περιέχει το νέο πλήθος κόμβων του δικτύου.

Το μήνυμα στο παράθυρο του κόμβου 2 “Update sends success = 1” δείχνει μόνο ότι το πακέτο στάλθηκε επιτυχώς στον γειτονικό κόμβο (στη συγκεκριμένη περίπτωση είναι και ο τελικός παραλήπτης).

Ο κόμβος 1 έλαβε το πακέτο update και ανανέωσε τη τιμή του πλήθους κόμβων του δικτύου ώστε να έχει τη σωστή τιμή ίση με 2. Φαίνεται στο παράθυρο του κόμβου 1 το μήνυμα “Send success = -1”. Αυτό δείχνει ότι ο κόμβος 1 ήθελε αρχικά να στείλει το πακέτο τύπου update, το οποίο έλαβε από τον κόμβο 2, σε κόμβο ακόμα πιο αριστερά του. Όμως, σύγκρινε, πρώτα, τη διεύθυνσή του με τη διεύθυνση του τελικού παραλήπτη του πακέτου update, πληροφορία η οποία περιέχεται στο κάθε πακέτο και συμπέρανε ότι το πακέτο έφτασε στον τελικό του προορισμό, οπότε και σταμάτησε. Η τιμή -1 έχει τεθεί με σκοπό να φαίνεται αυτή η περίπτωση.

Ο κόμβος 2, μόλις ολοκληρώσει όλες τις διαδικασίες σύνδεσής του με το δίκτυο, ως τελευταίος πλέον κόμβος του δικτύου, είναι υπεύθυνος για να στέλνει προσκλήσεις, οπότε κάνει και αυτό, όπως φαίνεται και από το παράθυρο PuTTY του κόμβου αυτού, στο οποίο γράφονται τα σχετικά μηνύματα αποστολής προσκλήσεων.

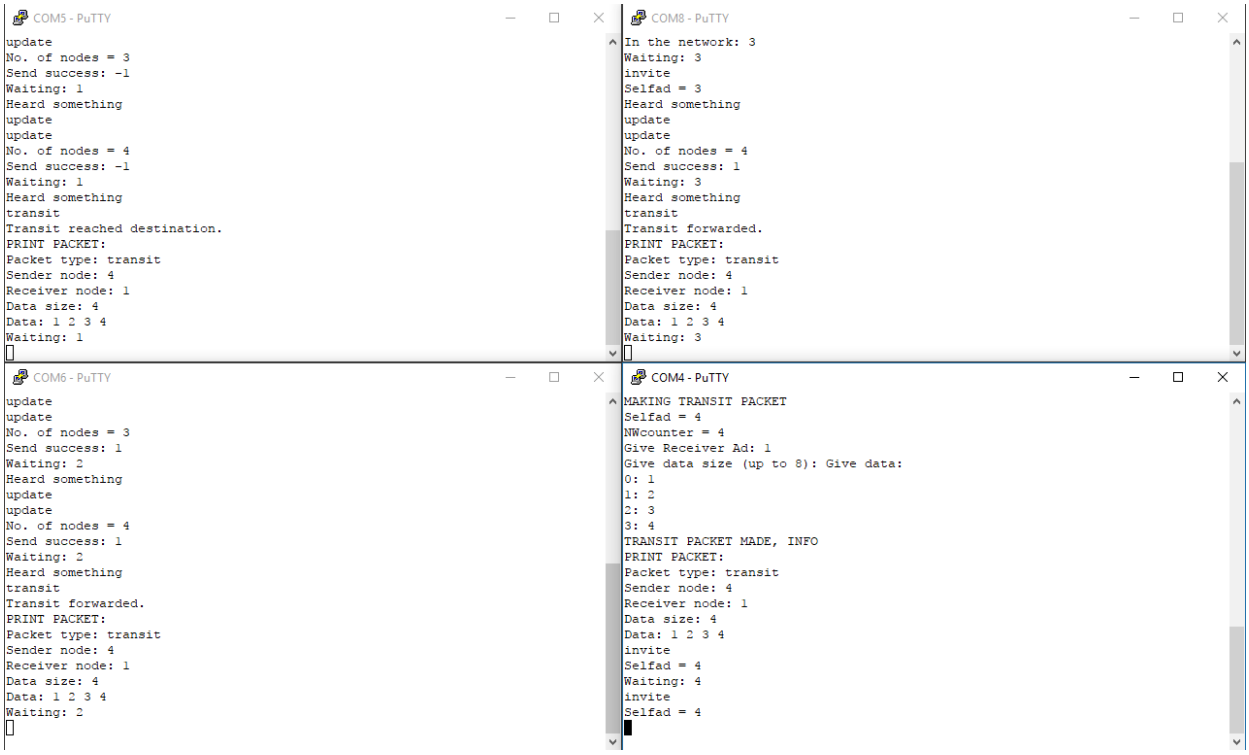
Επαναλαμβάνεται η ίδια διαδικασία λήψης μηνύματος πρόσκλησης, αρχικοποίησης κόμβου και ανανέωσης του υπόλοιπου δικτύου για τους άλλους δύο κόμβους που πρόκειται να συνδεθούν, οπότε στην τελική κατάσταση έχουμε το επόμενο στιγμιότυπο.



Κάθε φορά που συνδέεται νέος κόμβος, θέτει τη διεύθυνσή του με τιμή ίση με τη διεύθυνση του τελευταίο μέχρι στιγμής κόμβου, επαυξημένη κατά ένα και ανανεώνει τους υπόλοιπους κόμβους, ώστε να γνωρίζουν για την ύπαρξη του και για το νέο πλήθος κόμβων του δικτύου. Τελικά έχουμε 4 κόμβους στο δίκτυο και ο 4ος κόμβος, ως τελευταίος, στέλνει προσκλήσεις για σύνδεση περαιτέρω συσκευών στο δίκτυο.

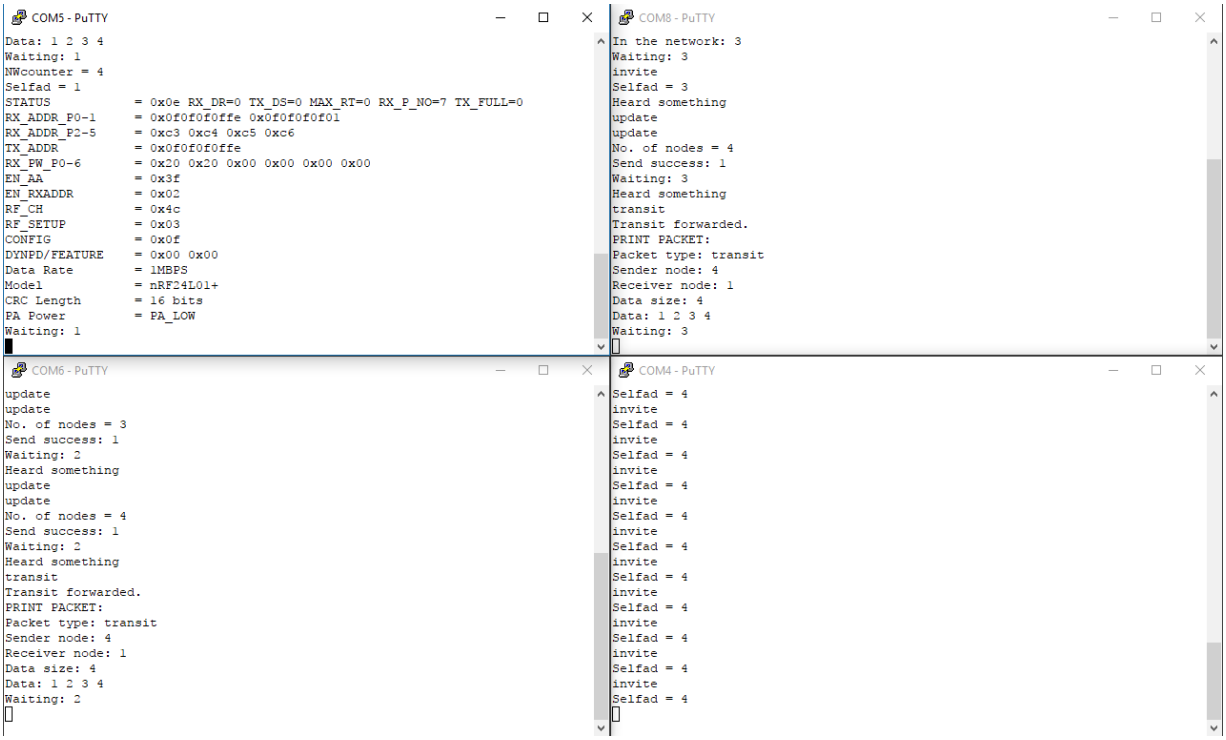
5.3.2 Αποστολή απλού πακέτου μεταξύ κόμβων του δικτύου

Μία από τις λειτουργίες που μπορεί να εκτελεστεί είναι η δημιουργία πακέτου για απλή αποστολή. Σε τέτοιου είδους πακέτο, ο κάθε κόμβος που είναι διαμεσολαβητής χρειάζεται απλά να προωθήσει το πακέτο. Για τον τελευταίο παραλήπτη, ο οποίος ήταν και ο στόχος του πακέτου, έχουμε θέσει ως λειτουργία απλά να εκτυπώνει στην οθόνη όλα τα περιεχόμενα του πακέτου που έλαβε. Προφανώς, ανάλογα με τις απαιτήσεις που θα μπορούσε να έχει το δίκτυο, ο κόμβος θα εκτελούσε ανάλογες διαδικασίες με το πακέτο που έλαβε. Ως δεδομένα στο πακέτο, έχουμε θέσει απλά έναν πίνακα μέχρι 8 στοιχείων, σκοπός του οποίου είναι μόνο συμβολικός και για να φαίνεται καλύτερα η κάθε λειτουργία του δικτύου. Τα δεδομένα αυτά μπορούν επίσης να επεκταθούν, ώστε να ικανοποιούν τις απαιτήσεις και τις ανάγκες του δικτύου και των κόμβων αυτού. Έστω ότι ο κόμβος 4 θέλει να στείλει ένα τέτοιο πακέτο στον κόμβο 1, με δεδομένα πλήθους 4 και τιμές 1, 2, 3 και 4. Έχουμε την εξής οθόνη.



Αφού πληκτρολογήσουμε 'T' στη σειριακή είσοδο του κόμβου 4, ενεργοποιείται η λειτουργία δημιουργίας και αποστολής τέτοιου πακέτου απλής μεταφοράς. Δίνονται στην αρχή οι πληροφορίες της διεύθυνσης του κόμβου, στον οποίον ενεργοποιήθηκε η λειτουργία αυτή και του πλήθους κόμβων του δικτύου και ζητείται ο επιθυμητός τελικός παραλήπτης, στο οποίο σημείο πληκτρολογούμε τον αριθμό 1. Ζητείται, στη συνέχεια, το πλήθος δεδομένων για αποστολή, όπου πληκτρολογούμε 4, και για τα δεδομένα γράφουμε 1 2 3 4. Δημιουργείται το πακέτο, εκτυπώνεται στο παράθυρο του κόμβου 4, για έλεγχο, και γίνεται η αποστολή. Στη συνέχεια, φαίνεται στα παράθυρα των ενδιάμεσων κόμβων, 3 και 2, ότι παραλαμβάνουν αυτοί το πακέτο, το εκτυπώνουν για έλεγχο και το προωθούν. Τελικά, το λαμβάνει ο κόμβος 1, εκτυπώνει στην οθόνη ότι έφτασε στον τελικό παραλήπτη και εκτυπώνει τα περιεχόμενά του.

Μία άλλη λειτουργία είναι η εκτύπωση λεπτομερειών (print details) του κόμβου. Δηλαδή, όταν γραφεί 'P' σε κάποιο κόμβο, εκτυπώνονται λεπτομέρειες για τα χαρακτηριστικά του πομποδέκτη, όπως ποια pipes έχει ανοιγμένα, ή τι ισχύς καταναλώνεται για την αποστολή και λήψη μηνυμάτων. Έχουμε το επόμενο screenshot, όπου πατήθηκε 'P' στον κόμβο 1.

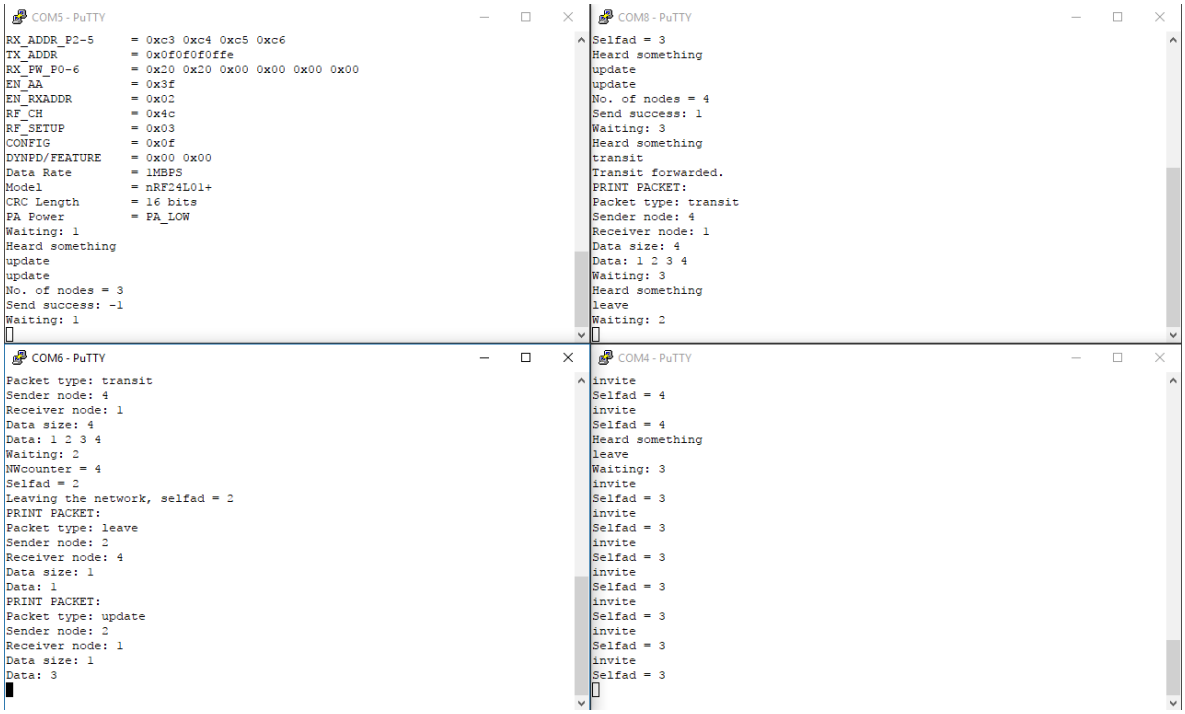


Μόλις πληκτρολογήθηκε το γράμμα αυτό, εμφανίστηκε μία σειρά δεδομένων του πομποδέκτη. Θυμίζουμε ότι η λειτουργία του πομποδέκτη που χρησιμοποιήθηκε εξηγήθηκε σε προηγούμενο κεφάλαιο. Τα πιο σημαντικά, για την υλοποίηση αυτή, χαρακτηριστικά του πομποδέκτη, που εκτυπώνονται στην οθόνη, είναι τα εξής:

- RX_ADDR_P0-1: είναι οι διευθύνσεις των reading pipes 0 και 1 (με μέγεθος διεύθυνσης 40 bits).
- RX_ADDR_P2-5: είναι οι διευθύνσεις των reading pipes 2 έως και 5. Θυμίζουμε ότι αυτά τα pipes είναι επίσης 40 bits, αλλά παίρνουν τα 32 υπόλοιπα bits από το pipe 1.
- TX_ADDR: είναι η διεύθυνση του writing pipe. Θυμίζουμε ότι όταν ανοίγει writing pipe σε μία διεύθυνση, ταυτόχρονα ανοίγει και reading pipe στο 0 με ίδια διεύθυνση, από όπου λαμβάνονται τα acknowledgements.
- Data Rate: Ο ρυθμός δεδομένων, με τον οποίον γίνονται αποστολές.
- PA Power: Η ισχύς που καταναλώνει όταν είναι ενεργοποιημένο το ασύρματο. Έχει τις δυνατές τιμές RF24_PA_MIN, RF24_PA_LOW, RF24_PA_HIGH και RF24_PA_MAX που αντιστοιχούν σε dBm στις τιμές -18dBm, -12dBm,-6dBm και 0dBm. Χρησιμοποιείται η χαμηλότερη τιμή, διότι τα πειράματα εκτελούνται με τους κόμβους συνδεδεμένους σε έναν υπολογιστή, οπότε οι αποστάσεις είναι αρκετά μικρές.

5.3.3 Αναχώρηση κόμβου από το δίκτυο, με ενημέρωση του δικτύου

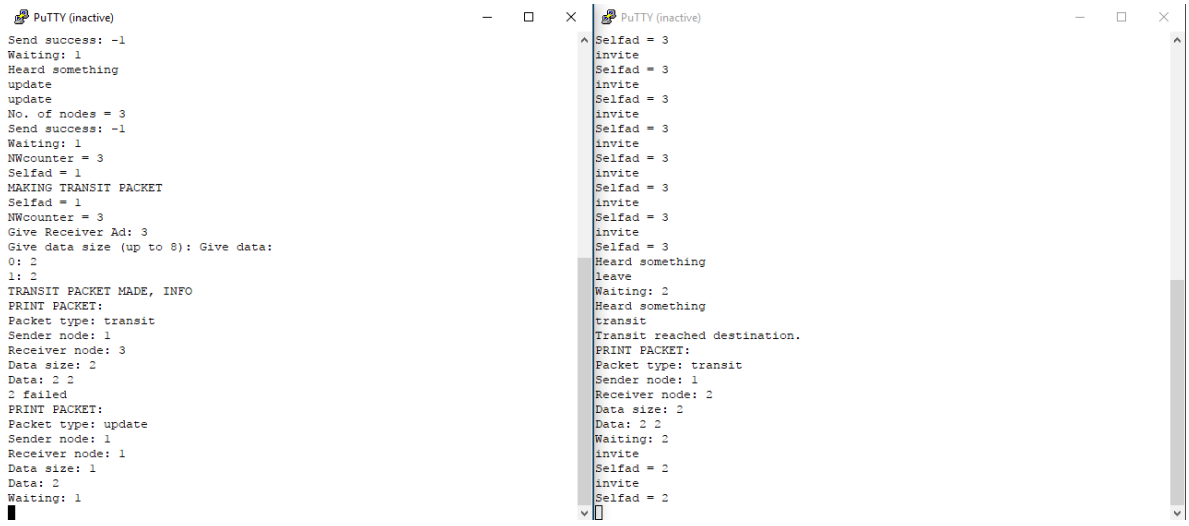
Τελευταία λειτουργία με τη χρήση πληκτρολογίου είναι η αναχώρηση κόμβου από το δίκτυο με ενημέρωση το δικτύου, δηλαδή ο κόμβος που πρόκειται να αναχωρήσει από το δίκτυο ενημερώνει πριν το υπόλοιπο δίκτυο για το γεγονός αυτό, έτσι ώστε να μπορέσουν οι υπόλοιποι κόμβοι να γνωρίζουν και να προσαρμοστούν στην αλλαγή. Γράφουμε στον κόμβο 2 το γράμμα 'L', και προκύπτει η επόμενη εικόνα.



Ο κόμβος 2 δημιουργεί δύο πακέτα. Ένα πακέτο τύπου leave το οποίο στέλνει στον τελευταίο κόμβο το δικτύου (στην περίπτωσή μας στον κόμβο 4) και ένα πακέτο τύπου update που στέλνει στον πρώτο κόμβο του δικτύου. Οι κόμβοι που λαμβάνουν το πακέτο leave μειώνουν τη διεύθυνσή τους κατά ένα, τη τιμή του πλήθους κόμβων δικτύου κατά ένα και προωθούν το πακέτο. Οι κόμβοι που λαμβάνουν το πακέτο update απλά διαβάζουν από τα δεδομένα του πακέτου τη νέα τιμή του πλήθους κόμβων του δικτύου και την αποθηκεύουν. Τελικά το δίκτυο θα έχει 3 κόμβους οι οποίοι θα έχουν αριθμητικά διαδοχικές διευθύνσεις, 1, 2 και 3, αντίστοιχα. Ο κόμβος που είχε διεύθυνση 3 τώρα έχει διεύθυνση 2, όπως και φαίνεται στο παράθυρό του, όπου πριν είχε εκτυπωθεί “Waiting: 3”, ενώ, αφού έλαβε το πακέτο leave, εκτύπωσε στην οθόνη “Waiting: 2”. Ο κόμβος που είχε διεύθυνση 4 έχει τώρα διεύθυνση 3 και συνεχίζει να είναι ο τελευταίος του δικτύου, οπότε και συνεχίζει να είναι υπεύθυνος για τις αποστολές μηνυμάτων προσκλήσεων. Φαίνεται και στο παράθυρό του, όπου, πριν λάβει το μήνυμα τύπου leave έστελνε προσκλήσεις ως κόμβος 4, ενώ στη συνέχεια στέλνει ως κόμβος 3.

5.3.4 Αναχώρηση κόμβου από το δίκτυο, χωρίς ενημέρωση του δικτύου

Επομένως, στην παρουσίαση αυτή, τώρα έχουμε 3 κόμβους. Για να δειχθεί η διαδικασία που ακολουθείται όταν ένας κόμβος φεύγει από το δίκτυο χωρίς ενημέρωση των υπόλοιπων κόμβων του δικτύου, αποσυνδέουμε τον κόμβο 2 από τον υπολογιστή χωρίς να εκτελέσει κάποια εντολή. Οι κόμβοι 1 και 3 νομίζουν ότι ο κόμβος 2 υπάρχει κανονικά. Το δίκτυο αντιλαμβάνεται ότι κάποιος κόμβος έφυγε χωρίς να ενημερώσει μόλις κάποιος γειτονικός κόμβος αυτού που έφυγε προσπαθήσει να του στείλει κάτι (πρώην γειτονικός, αφού ο κόμβος που έφυγε δεν υπάρχει πια στο δίκτυο). Έτσι έχουμε την ακόλουθη κατάσταση, όπου δείχνουμε μόνο τα παράθυρα των κόμβων 1 και 3 οι οποίοι είναι οι μόνοι που έχουν μείνει στο δίκτυο.



Ο κόμβος 1 δημιουργεί ένα απλό πακέτο για μεταφορά προς τον κόμβο 3, με την ίδια διαδικασία που περιγράφηκε σε προηγούμενη παράγραφο. Τα περιεχόμενά του πακέτου δεν έχουν σημασία. Μόλις το πακέτο δημιουργείται, γίνεται η αποστολή του. Γίνεται εκτύπωση “2 failed”, με την οποία δείχνεται ότι η αποστολή το πακέτου προς τον κόμβο 2 απέτυχε. Ο κόμβος 1 είναι τώρα υπεύθυνος να ανανεώσει όλο το δίκτυο πριν επαναληφθεί η προσπάθεια αποστολής του πακέτου. Εκτελεί τις διαδικασίες που θα εκτελούσε ο κόμβος 2, αν ενημέρωνε το υπόλοιπο δίκτυο πριν φύγει, και τελικά το δίκτυο έχει πλήθος κόμβων 2 και ο κόμβος που είχε διεύθυνση 3 τώρα έχει διεύθυνση 2 (συνεχίζει να είναι αυτός υπεύθυνος για προσκλήσεις, αφού συνεχίζει να είναι τελευταίος κόμβος του δικτύου). Επαναλαμβάνεται η αποστολή του πακέτου με τελικό παραλήπτη, πλέον, τον κόμβο 2, το λαμβάνει και το εκτυπώνει στην οθόνη.

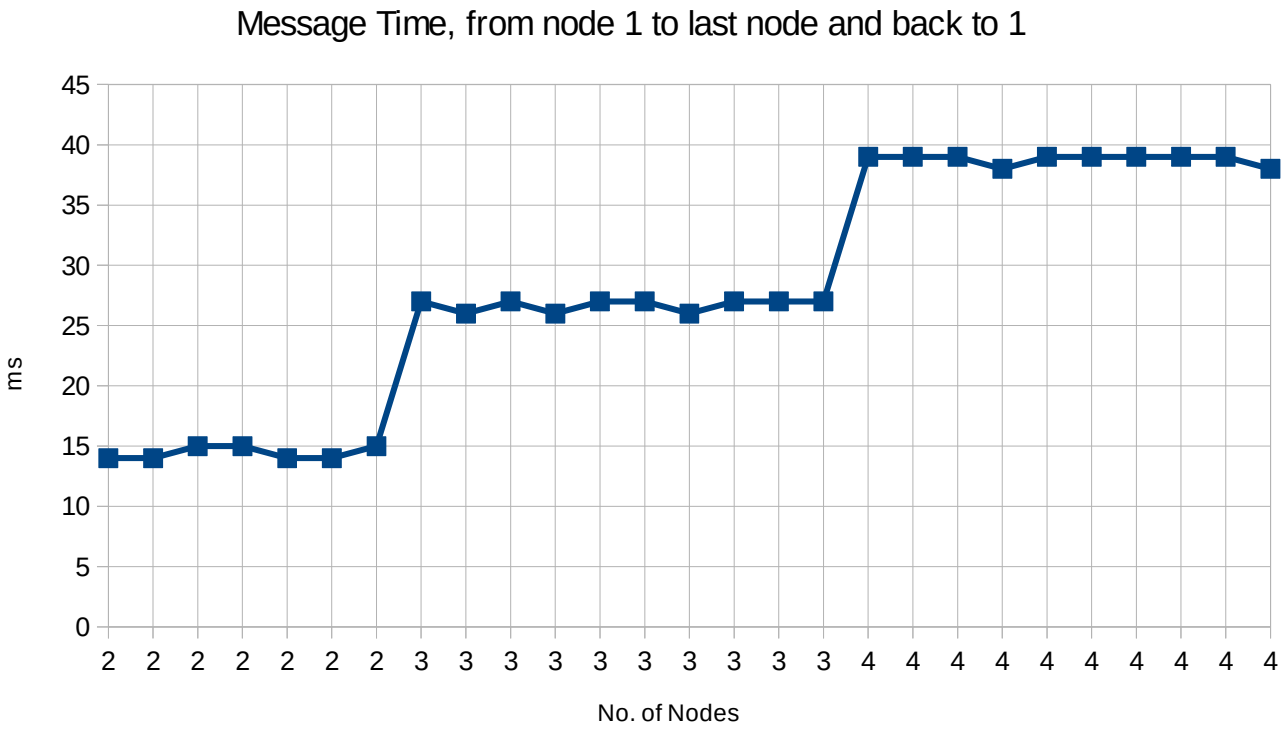
Στα προηγούμενα αποτυπώθηκε μία δοκιμαστική εκτέλεση του αλγορίθμου που υλοποιήθηκε στο κεφάλαιο αυτό. Μέσω αυτής έγινε προσπάθεια να γίνουν κατανοητές όλες οι λειτουργίες που προσφέρονται. Σκοπός κάθε λειτουργίας ήταν η ευκολότερη μεταχείριση του δικτύου που αναφέρουμε, μέσω της αυτοματοποίησης διαδικασιών προσαρμογής αυτού.

6 Επιμέρους αποτελέσματα της υλοποίησης

Σε αυτό το κεφάλαιο δίνεται καταγραφή στοιχείων διαφορετικών πειραματικών τρεξιμάτων με σκοπό να εξεταστούν οι επιδόσεις του όλου συστήματος και οι πιθανές αδυναμίες του αλγορίθμου που υλοποιήθηκε.

Στα επόμενα έχουμε μερικά διαγράμματα με σκοπό να δειχθεί ο χρόνος που χρειάζεται για να ολοκληρωθούν οι αποστολές πακέτων δεδομένων.

Το πρώτο πείραμα που εκτελέστηκε είχε σκοπό να βρεθεί αν είναι δυνατό να λειτουργήσει το δίκτυο με μεγάλο πλήθος κόμβων. Έγιναν υπολογισμοί ώστε να βρεθεί αν, όταν αυξάνεται το πλήθος των κόμβων, αυξάνεται και ο χρόνος αποστολής πακέτου δεδομένων, και, αν ναι, κατά πόσο. Έχουμε ένα διάγραμμα που στον οριζόντιο άξονα έχουμε το πλήθος των κόμβων, και στον κατακόρυφο έχουμε το χρόνο που χρειάζεται ένα μήνυμα για να πάει από τον κόμβο 1 στον τελευταίο κόμβο και πάλι πίσω στον κόμβο 1. Χρησιμοποιήθηκε αυτή η συγκεκριμένη διαδρομή αποστολών του δοκιμαστικού πακέτου διότι είναι ο πιο εύκολος τρόπος να μετρηθεί ο χρόνος της, αφού απλά αφαιρείται χρονική στιγμή που ο κόμβος 1 έστειλε το μήνυμα από τη χρονική στιγμή που το έλαβε πίσω.



Στο διάγραμμα αυτό υπάρχουν 10 μετρήσεις για κάθε διαφορετικό δυνατό πλήθος κόμβων. Παρατηρούμε ότι η αύξηση του χρόνου είναι σχετικά γραμμική. Σημαντικό είναι εδώ να σημειώσουμε ότι το πλήθος αποστολών που γίνονται ανάλογα με το πλήθος κόμβων αυξάνεται κατά δύο κάθε φορά. Έχουμε τον ακόλουθο πίνακα. Στην πρώτη στήλη έχουμε το πλήθος των κόμβων. Στη δεύτερη έχουμε το συνολικό πλήθος αποστολών που γίνονται για να αποσταλεί το δοκιμαστικό πακέτο από τον κόμβο 1 στον τελευταίο κόμβο του δικτύου και να επιστρέψει πάλι στον κόμβο 1. Στην τρίτη στήλη έχουμε τον μέσο χρόνο αυτής της διαδρομής του δοκιμαστικού πακέτου, ο οποίος είναι υπολογισμένος από τις τιμές του

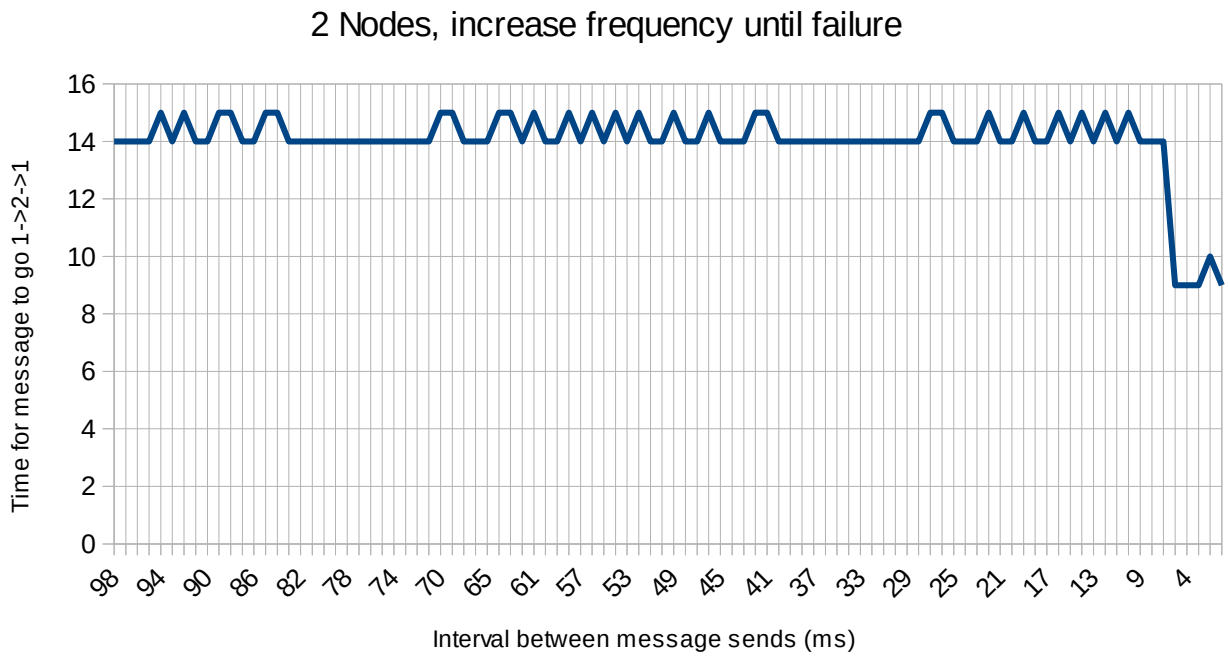
προηγούμενου διαγράμματος. Στην τέταρτη στήλη υπάρχει ο μέσος χρόνος μίας αποστολής, ο οποίος υπολογίστηκε διαιρώντας τα στοιχεία της τρίτης προς τα στοιχεία της δεύτερης στήλης.

Πλήθος κόμβων	**Πλήθος αποστολών**	**Μέσος χρόνος για να πάει το μήνυμα 1->τέλος->1 (ms)**	**Μέσος χρόνος 1 αποστολής (ms)**
2	2	14.5	7.25
3	4	26.7	6.675
4	6	38.8	6.467

Από τον πίνακα αυτό, φαίνεται ότι υπάρχει μία μικρή μείωση του χρόνου που χρειάζεται για μία αποστολή του δοκιμαστικού πακέτου. Αυτή θεωρείται ότι οφείλεται στις επιπλέον εντολές που εκτελεί ο τελευταίος κόμβος, προκειμένου να αλλάξει την κατεύθυνση και να στείλει το δοκιμαστικό πακέτο πίσω, προς τον κόμβο 1. Ο χρόνος που χρειάζεται για να εκτελεστούν αυτές οι εντολές διαιρείται κάθε φορά με μεγαλύτερο αριθμό, και έτσι προκύπτει αυτή η μικρή μείωση. Τελικά, πρακτικά είναι σταθερός ο χρόνος αποστολής ανεξάρτητα από το μέγεθος του δικτύου, καθιστώντας την εφαρμογή δυνατή για μεγαλύτερα δίκτυα από αυτή τη σκοπιά.

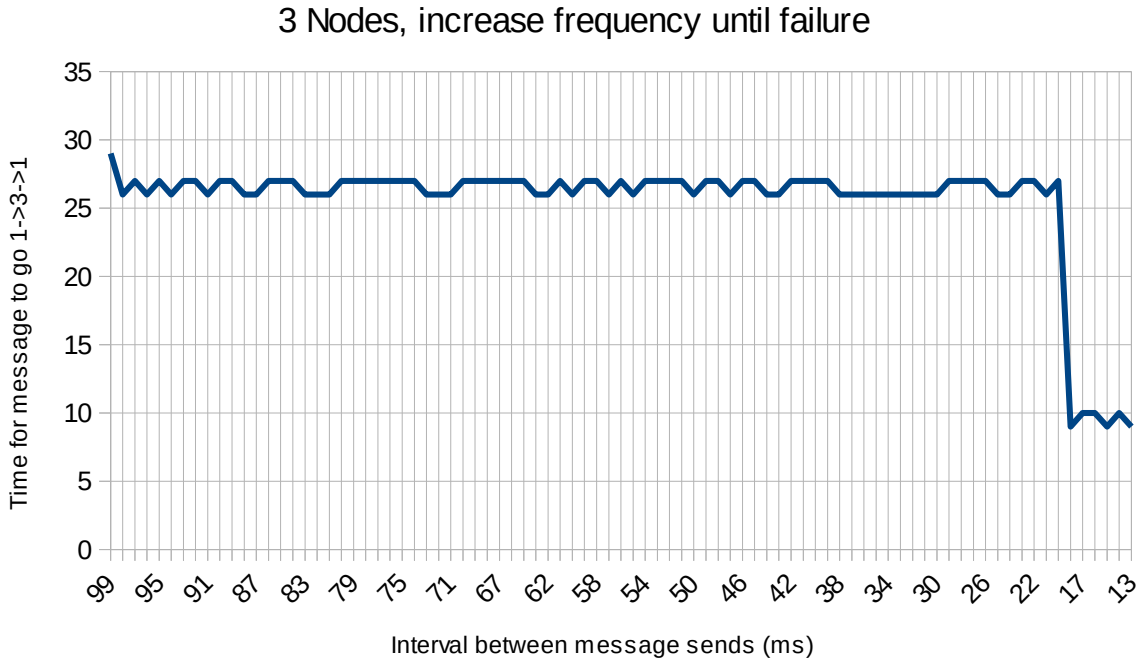
Το επόμενο πείραμα πάνω στην εφαρμογή μας ήταν να βρούμε την κρίσιμη συχνότητα αποστολής μηνυμάτων, η οποία, αν ξεπεραστεί, τότε το δίκτυο δε θα έχει αρκετό χρόνο να κάνει όλες τις αποστολές, με αποτέλεσμα να αποτύχουν αυτές. Αυτή η συχνότητα υπολογίστηκε ως εξής. Έχουμε την ίδια δομή με αυτήν του προηγούμενου πειράματος, στην οποία στέλνεται ένα δοκιμαστικό πακέτο ανά ένα καθορισμένο χρονικό διάστημα. Προκειμένου να υπολογιστεί η κρίσιμη συχνότητα, μειώνουμε αυτό το διάστημα (επομένως η συχνότητα αυξάνεται), μέχρι να δούμε ότι το δίκτυο απέτυχε στην αποστολή. Στα επόμενα δεν καταγράφεται η συχνότητα, αλλά προτιμήθηκε να καταγραφεί το ίδιο το χρονικό διάστημα μεταξύ των αποστολών του δοκιμαστικού πακέτου. Έγινε η καταγραφή αυτού, έτσι ώστε να υπάρχει μία καλύτερη διαίσθηση, διότι καταγράφεται και ο χρόνος που χρειάζεται για να διανύσει το πακέτο τη συγκεκριμένη αυτή διαδρομή. Είναι πιο εύκολο να γίνει σύγκριση μεταξύ χρόνων, αντί να έχουμε από τη μία συχνότητα και από την άλλη χρόνο.

Για δίκτυο με 2 κόμβους έχουμε το εξής διάγραμμα.



Παρατηρούμε ότι περίπου στα 7ms αρχίζει και μειώνεται ο χρόνος με ανώμαλο ρυθμό, και μετά το δίκτυο αποτυγχάνει. Η τιμή 7ms διάστημα μεταξύ αποστολών είναι φυσιολογική, αφού 7.25ms περίπου κάνει το μήνυμα να σταλεί, οπότε με μικρότερο διάστημα, πριν προλάβει ο κόμβος 2 να λάβει πλήρως το πακέτο, γίνεται αποστολή και άλλου, άρα αποτυγχάνει.

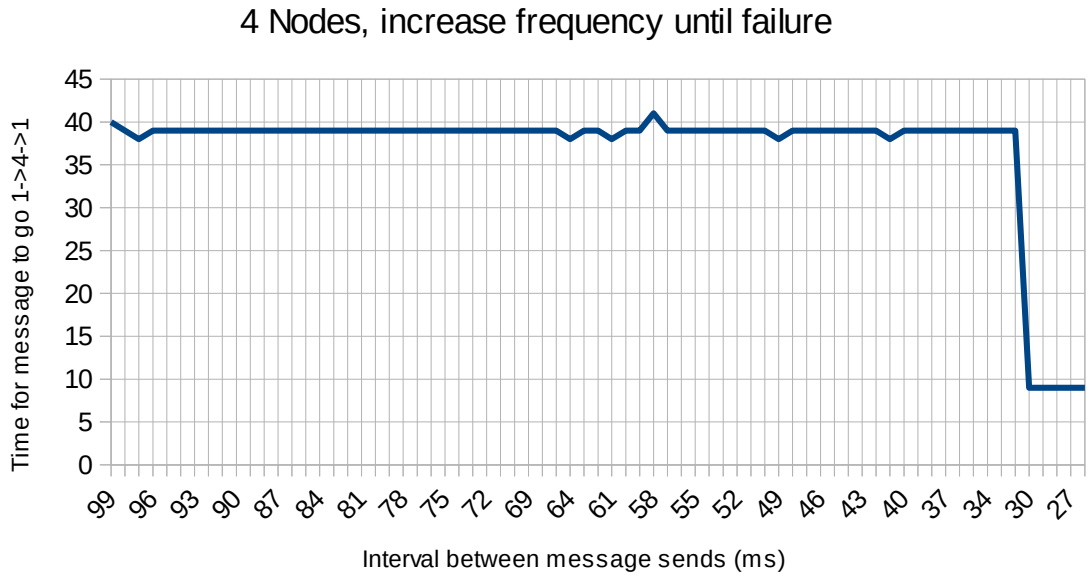
Για 3 κόμβους, έχουμε το επόμενο σχήμα.



Πάλι, παρατηρούμε σταθερή και σωστή λειτουργία μέχρι τα 20ms. Αυτός ο χρόνος είναι περίπου 7ms μικρότερος από τα 26.7ms που είναι ο μέσος χρόνος που χρειάζεται το μήνυμα για να πάει από τον κόμβο 1 στον κόμβο 3 και πίσω στον 1, όπως καταγράφηκε στο προηγούμενο πείραμα. Επομένως, βρισκόμαστε πάλι στην ίδια κατάσταση, όπου δεν

προλαβαίνει το μήνυμα να κάνει τον κύκλο πριν σταλθεί επόμενο μήνυμα. Η αποτυχία οφείλεται στο ότι υπάρχουν τόσα πολλά μηνύματα που στέλνονται, οπότε σε κάποιο κόμβο κάποια στιγμή γίνεται αποστολή και λήψη ταυτόχρονα, λειτουργία την οποία δε μπορεί να εκτελέσει ο πομποδέκτης, αφού μπορεί να λειτουργεί μόνο ως πομπός ή δέκτης και όχι και τα δύο ταυτόχρονα.

Τέλος, ας δούμε και το δίκτυο με 4 κόμβους.



Παρατηρείται η ίδια συμπεριφορά, αυτή τη φορά στα 31ms, τιμή η οποία είναι περίπου 7ms μικρότερη από τα 38.8ms που χρειάζεται για να γίνει ο πλήρης κύκλος.

Συνεπώς, ένας εμπειρικός τύπος που θα μπορούσε να υπολογιστεί από τα προηγούμενα διαγράμματα, ο οποίος θα δίνει το χρονικό διάστημα που χρειάζεται το μήνυμα για να πάει από τον κόμβο 1 στον τελευταίο κόμβο και πίσω στον 1, αν θεωρήσουμε ότι για μία αποστολή χρειάζονται 7ms, προκύπτει από τον ακόλουθο τύπο. Ν είναι το πλήθος κόμβων του δικτύου.

$$Time = (N-1) * 2 * 7$$

Η τιμή (N-1)2 είναι το συνολικό πλήθος αποστολών και 7 είναι τα 7ms που θεωρούμε ότι χρειάζεται περίπου κάθε αποστολή. Από τα πειράματα παρατηρήσαμε ότι το κρίσιμο διάστημα μεταξύ αποστολών είναι περίπου 7ms μικρότερο από το συνολικό χρονικό διάστημα το οποίου δίνεται ο προηγούμενος τύπος. Επομένως, αν θέλουμε να δείξουμε έναν εμπειρικό τύπο του κρίσιμου αυτού διαστήματος, αυτός θα είναι ο εξής (η κρίσιμη συχνότητα είναι το αντίστροφο του κρίσιμου διαστήματος).

$$Critical\ Interval = (2N-3) * 7$$

Ο τύπος αυτός είναι ίσος με τον προηγούμενο, μειωμένο κατά 7ms. Εδώ σημειώνουμε ότι το 7ms είναι πιο πραγματικό για το μικρό δίκτυο των 2 κόμβων. Μεγαλώνοντας το δίκτυο η τιμή μειώνεται, όμως επιλέχθηκε το 7ms ώστε να είμαστε σε ασφαλή τιμή. Αν το δούμε από πλευράς πολυπλοκότητας, το κρίσιμο διάστημα έχει γραμμική πολυπλοκότητα O(N), οπότε από αυτή τη σκοπιά, η εφαρμογή μας δεν είναι πολύ καλή καθώς μεγαλώνει το δίκτυο.

Ο τύπος αυτός αφορά το μέγιστο χρόνο που πιθανώς ένα πακέτο πρόκειται να χρειαστεί για να μετακινηθεί μέσα στο δίκτυο, αφού ο χρόνος που υπολογίζεται είναι αυτός που χρειάζεται για να πάει το πακέτο από τον κόμβο 1 στον τελευταίο και πίσω στον 1,

απόσταση η οποία είναι μέγιστη. Επομένως, αφορά τη χειρότερη περίπτωση, και γενικά κάθε πακέτο θα χρειάζεται λιγότερο χρόνο. Αν κανείς ξεπεράσει αυτήν την συχνότητα μηνυμάτων και παραμείνει σε τέτοια συχνότητα, το δίκτυο, στη χειρότερη περίπτωση, θα αρχίσει να δημιουργεί μηνύματα με μεγαλύτερο ρυθμό από το ρυθμό που μηνύματα φτάνουν στον τελικό προορισμό, οπότε το δίκτυο μπαίνει σε αστάθεια και μετά από κάποια χρονική στιγμή θα γεμίσουν όλοι οι buffers των συσκευών και δε θα μπορεί να στέλνεται τίποτα και το δίκτυο θα καταρρεύσει.

Τελευταίο πείραμα ήταν να μετρηθεί πόσος χρόνος χρειάζεται για να σταλθεί ένα μήνυμα, αν φύγουν ενδιάμεσοι κόμβοι χωρίς να ενημερώσουν το υπόλοιπο δίκτυο. Λόγω περιορισμένου αριθμού διαθέσιμων κόμβων για τα πειράματα (τέσσερις κόμβοι) μπόρεσαν να πραγματοποιηθούν λίγες δοκιμές με τη χρήση αυτών. Το πείραμα που εκτελέστηκε ήταν όπως τα προηγούμενα, όπου στέλνεται ένα πακέτο από τον κόμβο 1 στον τελευταίο κόμβο και πίσω στον 1, και μετριέται ο χρόνος που χρειάστηκε για να γίνει αυτό. Όμως, πριν σταλθεί το μήνυμα, έχει αποσυνδεθεί ένας ή και οι δύο ενδιάμεσοι κόμβοι, έτσι μπορεί να υπολογιστεί ο χρόνος που χρειάζεται για να γίνει η κάθε διαδικασία. Τα αποτελέσματα είναι τα εξής.

Αρχικό πλήθος κόμβων	**Πλήθος αναχωρήσεων**	**Τελικό πλήθος κόμβων**	**Χρόνος αποστολής (ms)**	**Μέσος χρόνος αποστολής χωρίς κάποια αναχώρηση (ms)**
3	1	2	124	26.7
4	1	3	144	38.8
4	2	2	197	38.8

Παρατηρούμε ότι όταν γίνεται μία μόνο αναχώρηση χρειάζονται περίπου 100ms παραπάνω από το μέσο χρόνο αποστολής χωρίς κάποια αναχώρηση για να προσαρμοστεί το δίκτυο και να γίνει η αποστολή. Αν γίνουν δύο αναχωρήσεις χωρίς ενημέρωση και στη συνέχεια γίνεται προσπάθεια αποστολής, τότε χρειάζονται περίπου 158ms.

Συνολικά, παρατηρούμε ότι το δίκτυο που υλοποιήθηκε έχει, από τη μία, γραμμική κλιμάκωση του ελάχιστου χρονικού διαστήματος μεταξύ αποστολών, αλλά αυτή η κλιμάκωση είναι σχετικά μικρή. Οπότε, για ένα μεσαίο μέγεθος δικτύου, περίπου 50 κόμβων, θα μπορούσε να χρησιμοποιηθεί με άνεση. Το μέγιστο μέγεθος τέτοιου δικτύου, όπως αναφέρθηκε, είναι 128 κόμβοι (λόγω της αρχιτεκτονικής του πομποδέκτη), οπότε ακόμα και σε αυτό το πλήθος μπορεί να γίνει η χρήση του προγράμματος, απλά με την προϋπόθεση ότι θα στέλνονται μηνύματα λίγο πιο σπάνια, αλλά όχι σε απαγορευτικό βαθμό.

7 Συμπεράσματα και αναγκαίες / προτεινόμενες μελλοντικές εργασίες

7.1 Συμπεράσματα από την παρούσα εργασία

Η παρούσα εργασία είχε δύο κύρια τμήματα:
1.Ένα πρώτο, εν πολλοίς, ανιχνευτικό τμήμα προς το οποίο έγινε μία σχετικά εκτεταμένη βιβλιογραφική αναζήτηση προς περιοχές τεχνολογιών και τεχνολογικών προϊόντων, με χρήση των οποίων μπορούμε να επιτύχουμε στο μέλλον αποδοτικότερη σχεδίαση και υλοποίηση ευφυών δικτύων παρακολούθησης και ελέγχου γεωγραφικά διεσπαρμένων οντοτήτων. Όπως αναφέρθηκε στην εισαγωγή της εργασίας (και επαναλαμβάνεται εδώ για λόγους νοηματικής πληρότητας του κειμένου) η ανάγκη των δικτύων αυτών προκύπτει πολύ συχνά στην πράξη σήμερα. Δύο ιδιαίτερα σημαντικές περιοχές εφαρμογών της πράξης του τύπου αυτού είναι οι ακόλουθες:

α) Παρακολούθηση και έλεγχος κινητών υπομονάδων σε περιπτώσεις αποστολών, όπως μεταφορά ασθενών, αποστολές έρευνας και διάσωσης κλπ.

β) Εθνικές και διεθνείς μεταφορές εμπορευμάτων και επιβατών. Οι λόγοι για αυτό είναι οι εξής:

   i. Στις μεταφορές επιβατών η ομαδοποίηση σε μονάδες μεταφοράς είναι προφανής, δεδομένου ότι οι επιβάτες κινούνται υποχρεωτικά σε οχήματα.

   ii. Στις συντριπτικό αριθμό περιπτώσεων μεταφορών αγαθών σήμερα χρησιμοποιούνται μονάδες μεταφοράς σαν τα εμπορευματοκιβώτια, τα φορτηγά οχήματα, τα βαγόνια σιδηροδρόμων, τα πλοία, τα αεροπλάνα κλπ.

Στην πράξη έχει αποδειχθεί ότι, όχι μόνο είναι επιθυμητή αλλά και, σε ιδιαίτερα μεγάλο αριθμό περιπτώσεων, αναγκαία η παρακολούθηση των επιμέρους κινητών μονάδων που αναφέρονται στις προηγούμενες δύο περιοχές εφαρμογών. Στην παρούσα εργασία κινηθήκαμε προς την κατεύθυνση αύξησης της ευφυΐας του όλου υποστηρικτικού συστήματος παρακολούθησης και ελέγχου γεωγραφικά διεσπαρμένων οντοτήτων, με αρχική επιλογή των ακολούθων:

1. Έρευνα της δυνατότητας αύξησης της υπολογιστικής ισχύος των ανεξάρτητων, ολοκληρωμένων υπολογιστικών σταθμών (embedded systems) που τοποθετούνται συνήθως στις προς παρακολούθηση γεωγραφικά διεσπαρμένες οντότητες. Στο πλαίσιο ανίχνευσης δυνατοτήτων του τύπου αυτού εξετάστηκε ένας αριθμός από συχνά χρησιμοποιούμενες στην πράξη συσκευές μικροελεγκτών, η οποίες μπορούν να χρησιμοποιηθούν ως ενσωματωμένα συστήματα και να δεχθούν κατάλληλες επεκτάσεις για αύξηση των δυνατοτήτων τους, όπως εντοπισμός θέσης μέσω χρήσης GPS, ασύρματη ζεύξη με αντίστοιχα κοντινά ενσωματωμένα συστήματα, επικοινωνία με άλλους σταθμούς ή με κεντρικό Σταθμό Βάσης μέσω του δικτύου GSM/GPRS κλπ. Έγινε επίδειξη μιας μεθοδολογίας επιλογής ανάμεσα σε συσκευές αυτού του τύπου με χρήση κατάλληλων κριτηρίων επιλογής.

   Μία δυνατότητα των ενσωματωμένων συστημάτων που εξετάστηκε ήταν το κατά πόσον είναι εφικτή η επαναφόρτωση μέρους προγράμματος της μνήμης προγράμματος και δεδομένων της συσκευής, με λήψη του σχετικού κώδικα και δεδομένων από ένα κεντρικό Σταθμό Βάσης. Προς την κατεύθυνση αυτή, εξετάστηκε η δυνατότητα ιδιοκατασκευής με χρήση dual-port RAM. Στο πλαίσιο

της έρευνας, εξετάστηκαν dual-port RAMs της πράξης και εξετάστηκαν και παρετέθηκαν επιλεγμένες ιδιότητες και χαρακτηριστικά τους.

2. Έγινε έρευνα της δυνατότητας αυτοοργάνωσης κινητών σταθμών, τοποθετημένων σε ομάδες μονάδων μεταφοράς αγαθών ή επιβατών (όπως λεωφορεία, βαγόνια τρένων, εμπορευματοκιβώτια, φορτηγά αυτοκίνητα κλπ.). Όπως αναφέρθηκε και προηγουμένως, όταν οι συγκεκριμένες ομάδες μονάδων μεταφοράς μπορούν να χαρακτηριστούν ως διακριτές οντότητες συμφέρει οι σταθμοί που είναι τοποθετημένοι στις μονάδες μεταφοράς της ομάδας να οργανώνονται σε ένα τοπικό τηλεπικοινωνιακό υποδίκτυο. Σκοπός του υποδικτύου αυτού είναι (i) η διευκόλυνση της παροχής υπηρεσιών σχετικών με τις κινητές μονάδες της ομάδας και (ii) η απλοποίηση της εργασίας ενός συνολικού "ευφυούς" δικτύου, το οποίο θα βρίσκεται σε κάποιο κεντρικό σταθμό βάσης και θα παρακολουθεί και ελέγχει σημαντικό αριθμό κινούμενων μονάδων μεταφοράς ή ομάδων μονάδων μεταφοράς.

   Προς αυτήν την κατεύθυνση κινηθήκαμε σε ένα κύριο κλάδο της παρούσας εργασίας όπου εξετάστηκαν πρωτόκολλα αυτοοργάνωσης σταθμών σε τηλεπικοινωνιακά υποδίκτυα. Κατ' αρχήν επελέγη για δοκιμή το πρωτόκολλο Chord, δεδομένου ότι αυτό αποτελεί τη βάση των περισσότερων δικτύων τύπου peer-to-peer στο διαδίκτυο σήμερα. Προκειμένου να αποκτήσουμε μία πρώτη εκτίμηση τέτοιων δικτύων υλοποιήθηκαν προγράμματα για την αυτοοργάνωση τεσσάρων σταθμών, εφοδιασμένων με πομποδέκτες RF συχνοτήτων. Ο ένας από τους σταθμούς αυτούς είναι βασίζεται σε ένα Arduino UNO, ο οποίος περιγράφεται στα επόμενα, ενώ οι υπόλοιποι τρεις σταθμοί βασίζονται σε Arduino Nano.

   Τα αποτελέσματα της δυνατότητας αυτοοργάνωσης που κατ' αρχήν είναι πολύ ικανοποιητικά και ενθαρρυντικά για τη μελλοντική συνέχεια της εργασίας στην περιοχή αυτή. Στην εργασία παρατίθεται ο σχετικός κώδικας σε γλώσσα C, η περιγραφή των επιμέρους τμημάτων των σχετικών αλγορίθμων, στιγμιότυπα οθονών (screenshots) από την εκτέλεση των σχετικών προγραμμάτων στους πραγματικούς κινητούς σταθμούς που αυτοοργανώνονται μέσω ασύρματης ζεύξης, καθώς και σχετικές μετρήσεις χρόνων και επιδόσεων ταχύτητας του πραγματικού συστήματος δοκιμής.

3. Εξετάστηκε, αλλά με πολύ απλό και εισαγωγικό τρόπο και χωρίς δοκιμές, η δυνατότητα κατασκευής ενός αντίστοιχου "σταθμού βάσης" για την παρακολούθηση και τον έλεγχο των γεωγραφικά διεσπαρμένων οντοτήτων, όπως αναφέρθηκε ήδη. Εξετάστηκαν κατ' αρχήν τα προβλήματα που παρουσιάζονται σε ένα τέτοιο σύστημα και πιθανή αρχιτεκτονική προγραμμάτων και δεδομένων του συστήματος αυτού που βοηθάει σημαντικά στην επίλυση των συγκεκριμένων προβλημάτων.

7.2 Προτάσεις για συνέχιση της εργασίας

1. Κατ' αρχήν προτείνεται η επέκταση της μεθοδολογίας Chord για αυτοοργάνωση των κοντινών δικτύων κινητών συσκευών που ανήκουν στην ίδια θεματική ομάδα σε δεύτερο ή και τρίτο επίπεδο, όπως προτείνεται στο Παράρτημα 2. Επίσης, προτείνεται η μελέτη ανάπτυξης ειδικού λογισμικού για ισχυρότερους τοπικούς σταθμούς, ώστε οι συγκεκριμένοι σταθμοί να συμμετέχουν σε περισσότερα του ενός δίκτυα Chord. Παραδείγματος χάριν: Ένας ισχυρός σταθμός τοποθετημένος σε μία μηχανή ενός τρένου (η οποία έχει τη δυνατότητα παραγωγής ή διάθεσης ηλεκτρικής ισχύος) θα ήταν χρήσιμο να μπορεί να συμμετέχει σε περισσότερα του ενός

αυτοοργανωμένα θεματικά δίκτυα βαγονιών. Με τον τρόπο αυτό θα μπορούσε να εξυπηρετήσει διαφορετικούς χρήστες ή/και εν γένει φορείς. Τέλος δε, προτείνεται η μελέτη εφικτότητας επέκτασης του δικτύου Chord σε δομές τύπου πλέγματος (mesh), όπως επίσης αναφέρεται στο Παράρτημα 2.

2. Προτείνεται η αγορά προσεκτικά επιλεγμένου υλικού (hardware), το οποίο πρέπει να περιέχει ισχυρό κεντρικό κινητό σταθμό (δηλαδή ενσωματωμένο σύστημα με ισχυρό μικροεπεξεργαστή) και σχετικά ισχυρούς επί πλέον κινητούς σταθμούς. Όλοι οι κινητοί σταθμοί δε πρέπει να έχουν πομποδέκτες επικοινωνίας με εμβέλεια που επιτρέπει πραγματικά πειράματα σε περιβάλλοντα ελεγχόμενης κυκλοφοριακής κίνησης (πχ. Πολυτεχνειούπολη, Πανεπιστημιούπολη κλπ.). Αξίζει να εξεταστούν ιδιαίτερα κινητοί σταθμοί που έχουν τη δυνατότητα, μέσω ενδεχόμενης χρήσης του λειτουργικού συστήματός τους, να κατεβάζουν (download) από κάποιο απομακρυσμένο σταθμό νέα προγράμματα προς εκτέλεση. Η δυνατότητα αυτή μπορεί να επιτρέψει στους κινητούς σταθμούς εύκολη και αποτελεσματική ολοκλήρωσή τους σε συστήματα υποβοήθησης λήψης αποφάσεων που εδράζονται σε κάποιο Σταθμό Βάσης. Προφανώς δε, προτείνεται εκτενέστερος πρακτικός πειραματισμός με βάση τις προαναφερθείσες νέες συσκευές και δοκιμές ολοκλήρωσης των συσκευών αυτών σε κινητά υποδίκτυα και στοιχειώδη συστήματα υποστήριξης λήψης αποφάσεων με συμμετοχή (εικονικού) Σταθμού Βάσης. Πρώτα αποτελέσματα θα μπορούσαν να ληφθούν αν οι προαναφερθέντες σταθμοί τοποθετηθούν σε οχήματα, πχ. αυτοκίνητα ΙΧ, τα οποία θα κινηθούν σε συνθήκες ελεγχόμενης κίνησης (πχ. εντός της Πολυτεχνειούπολης).

3. Ένα επόμενο ενδιαφέρον σημείο αποτελεί η ολοκλήρωση συστήματος GIS στον (εικονικό) Σταθμό Βάσης, ώστε η σχετική πληροφορία για τους σταθμούς να απεικονίζεται και σε κατάλληλους χάρτες.

4. Τέλος, θα είχε πολύ μεγάλο ενδιαφέρον η δοκιμαστική ανάπτυξη τμήματος ενός Σταθμού Βάσης, στηριγμένου στην Τελεολογική Δομή, σε επίπεδο προγραμμάτων για την απόδειξη της ορθότητας των προτάσεων για την αρχιτεκτονική του συστήματος (POCs, programs for Proof Of Concepts). Η δοκιμαστική αυτή, μερικά ανεπτυγμένη πλατφόρμα ανάπτυξης του Σταθμού Βάσης θα μπορούσε να χρησιμοποιηθεί για πολύ ενδιαφέρουσες δοκιμές ενός πρωτολειακού συνολικού ευφυούς συστήματος για την υποστήριξη της παρακολούθησης και του ελέγχου γεωγραφικά διεσπαρμένων μονάδων μεταφορών.

8 Παράρτημα

Για τη συγγραφή της διπλωματικής εργασίας χρησιμοποιήθηκε το πρόγραμμα LibreOffice. Αποτελεί ένα δωρεάν και open source πρόγραμμα τύπου office, και περιλαμβάνει δυνατότητες επεξεργασίας λέξεων, δημιουργίας και επεξεργασίας υπολογιστικών φύλλων (spreadsheets), διαφανειών, σχεδίων, εργασίας με βάσεις δεδομένων και σύνθεσης μαθηματικών τύπων.

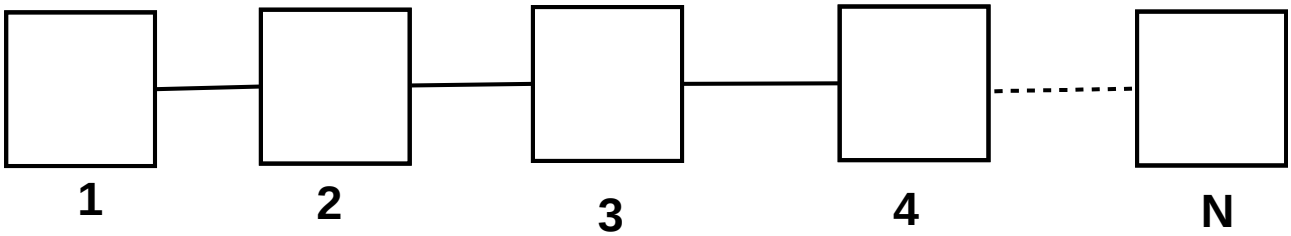
Για τη δημιουργία όλων των σχεδίων για τα δίκτυα, τα διαγράμματα ροής κλπ. χρησιμοποιήθηκε το Dia. Είναι επίσης ένα δωρεάν και open source λογισμικό για σχεδίαση διαγραμμάτων γενικού σκοπού.

Για τη δημιουργία, επεξεργασία και αποθήκευση του κώδικα χρησιμοποιήθηκε το GitHub. Είναι αποθήκη για έλεγχο εκδοχών, βασισμένη στο διαδίκτυο (web-based version control repository). Χρησιμοποιείται κυρίως για κώδικα, και η χρησιμότητά του φαίνεται στην προσπάθεια προσθήκης νέων κομματιών στον κώδικα. Αν δε δουλεύει ή για οποιοδήποτε λόγο θέλουμε να πάμε σε προηγούμενη εκδοχή του κώδικά μας, υπάρχει ιστορικό αλλαγών και μπορούμε να πάμε σε όποια εκδοχή θέλουμε.

Για την εγγραφή του κώδικα για τα Arduino χρησιμοποιήθηκε το περιβάλλον ανάπτυξης Arduino IDE. Αποτελεί το βασικό πρόγραμμα για Arduino, αφού περιέχει έτοιμες όλες τις απαραίτητες βιβλιοθήκες για αυτό, και έχει δημιουργηθεί από την ίδια την εταιρεία παραγωγής Arduino.

8.1 Παράρτημα 1: Προτάσεις για πιο λεπτομερή υλοποίηση του πρωτοκόλλου Chord

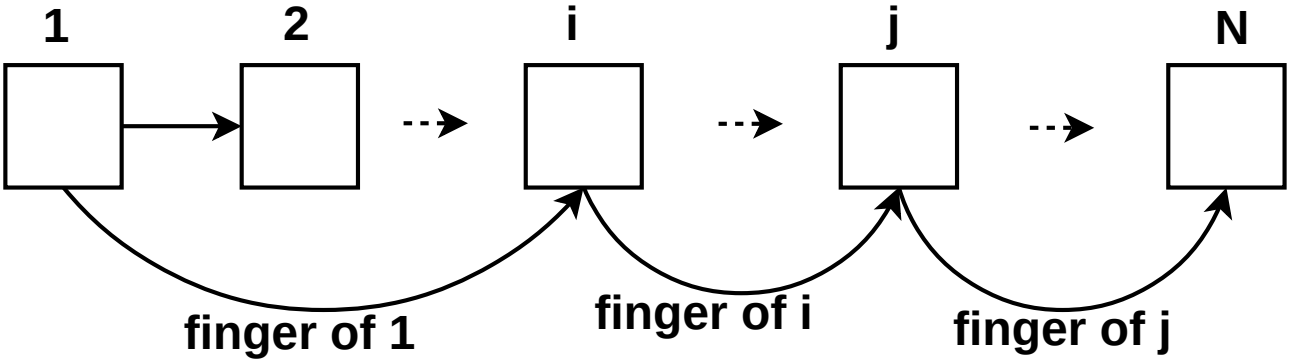
Στην εφαρμογή μας έχουν χρησιμοποιηθεί στοιχεία του Chord Paper, το οποίο εξηγήθηκε σε προηγούμενο κεφάλαιο. Τα στοιχεία που χρησιμοποιήθηκαν είναι πρώτον ότι κάθε κόμβος γνωρίζει ποιος είναι ο επόμενος του. Αυτό επιτεύχθηκε σε εμάς με την απαίτηση να υπάρχει αριθμητική συνέχεια μεταξύ των διαδοχικών κόμβων.



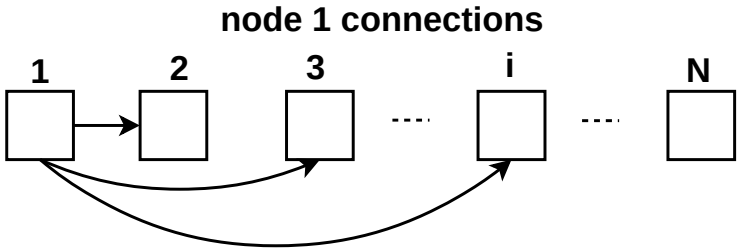
Με την ικανοποίηση αυτής τις ιδιότητας, κάθε κόμβος γνωρίζει ποιος είναι ο επόμενος του άμεσα, αφού χρειάζεται μόνο να αυξήσει τη διεύθυνσή του κατά ένα. Όμοια και για να βρει τον προηγούμενό του κόμβο, την μειώνει κατά ένα. Αυτό από τη μία καθιστά πολύ εύκολο να βρίσκει κάθε κόμβος τους γειτονικούς του, όμως η πολυπλοκότητα εμφανίζεται κάθε φορά που αλλάζει το δίκτυο, δηλαδή φεύγει ή έρχεται κάποιος κόμβος. Σε αυτήν την περίπτωση ενημερώνεται όλο το δίκτυο και ίσως κάποιοι κόμβοι να αλλάξουν τις διευθύνσεις τους.

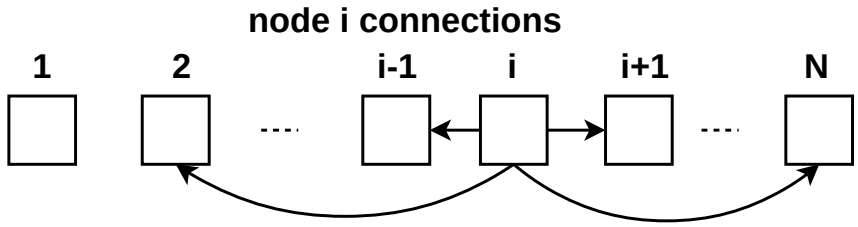
Ένα άλλο χαρακτηριστικό που θα μπορούσε η εφαρμογή μας να υιοθετήσει από το Chord ως επέκταση είναι τα finger tables. Όπως εξηγήθηκε, στο Chord, κάθε κόμβος έχει έναν πίνακα που ονομάζεται finger table. Με λίγα λόγια αυτός ο πίνακας περιέχει κόμβους με εκθετικά αυξανόμενη απόσταση από τον κόμβο που έχει τον πίνακα. Έτσι, κάθε κόμβος έχει περισσότερες από μία συνδέσεις με αποτέλεσμα τη μείωση διαμεσολαβητών όταν υπάρχει επιθυμία επικοινωνίας.

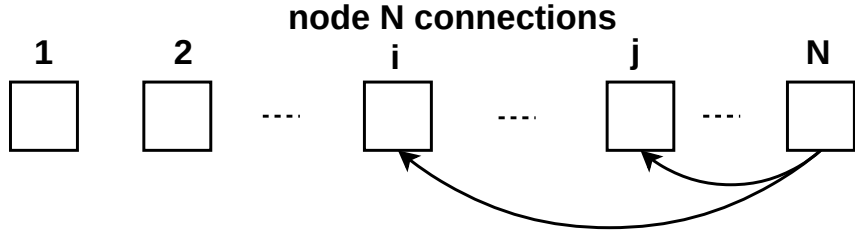
Στην εφαρμογή μας, θα μπορούσε κάθε κόμβος να είχε κάτι αντίστοιχο. Πιο συγκεκριμένα, κάθε κόμβος θα είχε έναν πίνακα με κάποιους από τους κόμβους με το οποίους έχει επαρκή εμβέλεια. Εδώ, σε αντίθεση με το Chord, έχουμε τον περιορισμό της εμβέλεια, οπότε δεν υπάρχει η ελευθερία να υπάρχουν συνδέσεις μεταξύ οποιωνδήποτε κόμβων. Σε υπάρχει λοιπόν ένα πίνακας που θα προσφέρει περισσότερες από μία συνδέσεις, αντίστοιχα με το finger table. Έτσι, αν για παράδειγμα είμαι ο κόμβος 1 και θέλω να στείλω ένα μήνυμα στον κόμβο N του δικτύου, ψάχνω στο finger table μου ποιος είναι ο πιο μακρινός κόμβος που βρίσκεται πριν τον N, και στέλνω σε αυτόν το μήνυμα. Αυτός ο κόμβος j με τη σειρά του πάει επίσης στον πιο μακρινό κόμβο πριν τον N και το στέλνει. Τελικά, ο κόμβος i έχει στο finger table του άμεσα τον κόμβο N, οπότε στέλνει το μήνυμα στον τελικό προορισμό. Φαίνεται και στο παρακάτω σχήμα.



Σε αντίθεση με το Chord, δεν υπάρχει κυκλική δομή, οπότε όσο προχωρούμε στο δίκτυο προς τους τελευταίους κόμβους, αυτοί θα μπορούν να έχουν σύνδεση με όλο και λιγότερους επόμενους κόμβους, όχι μόνο λόγο εμβέλειας, αλλά τώρα και επειδή δεν υπάρχουν αρκετοί επόμενοι. Μία λύση που θα μπορούσε να το αντιμετωπίσει αυτό, ώστε κάθε κόμβος να έχει περίπου ίσες σε πλήθος συνδέσεις, θα ήταν να υπάρχουν και συνδέσεις με προηγούμενους στη σειρά κόμβους. Έτσι, οι πρώτοι κόμβοι θα έχουν συνδέσεις με κυρίως επόμενους, οι κεντρικοί κόμβοι να έχουν και με επόμενους και με προηγούμενος, και οι τελευταίοι κόμβοι να έχουν συνδέσεις κυρίως με προηγούμενους. Φαίνεται καλύτερα και στο παρακάτω σχήμα.





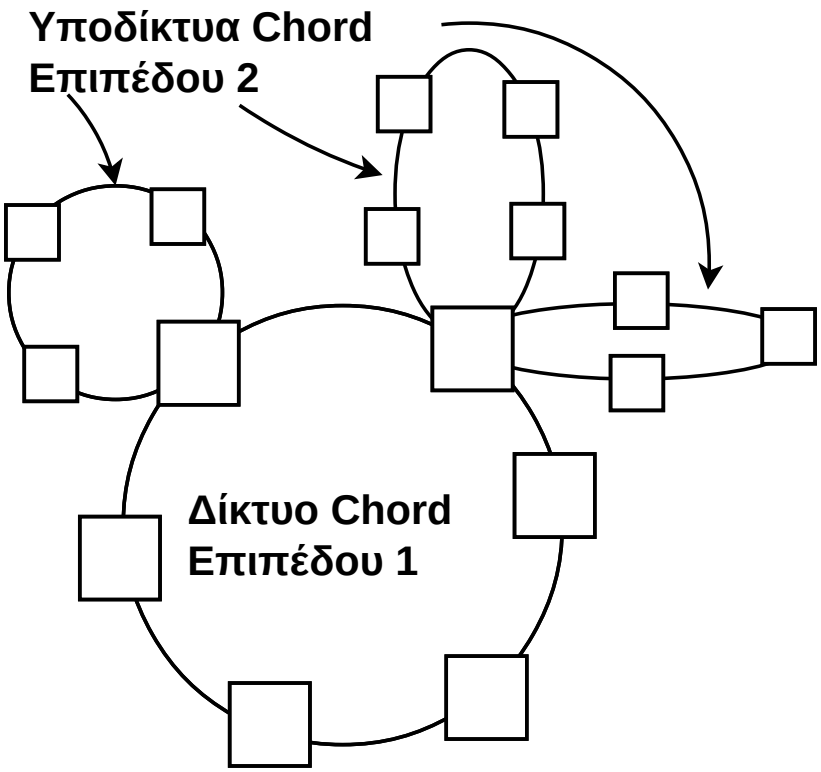


Η χρησιμότητα αυτής της επέκτασης φαίνεται περισσότερο σε δίκτυα στα οποία υπάρχει ανάγκη συχνής μεταφοράς δεδομένων μεγάλου όγκου. Απαίτηση για να είναι δυνατή αυτή η επέκταση είναι οι πομποδέκτες να έχουν αρκετά μεγάλη εμβέλεια ώστε να υπάρχουν αρκετές συνδέσεις. Αν η εμβέλεια επαρκεί για να υπάρχει, για παράδειγμα, μία μόνο παραπάνω σύνδεση, τότε ο αλγόριθμος γίνεται πιο πολύπλοκος για πολύ μικρό πλεονέκτημα, γεγονός που μπορεί να καταστήσει την επέκταση πρακτικά άχρηστη, αφού ίσως να μην εξοικονομείται χρόνος.

8.2 Παράρτημα 2: Δυνατή επέκταση του πρωτοκόλλου Chord σε δύο ή τρεις διαστάσεις

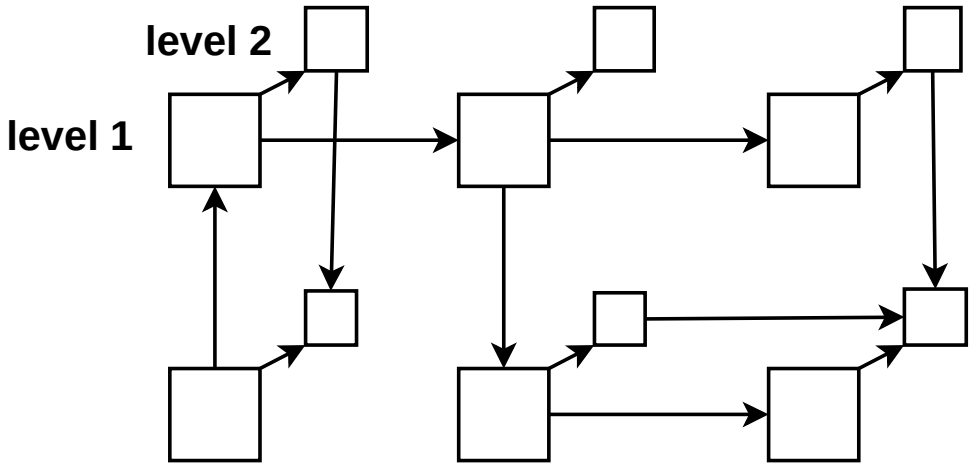
Μπορεί να φανταστεί κανείς περιπτώσεις όπου ένας σταθμός του δικτύου Chord, τον οποίον ας ονομάσουμε Κ, μπορεί να έχει ανάγκη να δημιουργήσει τοπικά υποδίκτυα (ένα η περισσότερα), τα οποία θα έχουν ως κοινό κόμβο το συγκεκριμένο κόμβο Κ του δικτύου Chord αλλά θα είναι τα ίδια ανεξάρτητα υποδίκτυα τύπου Chord με τοπική δομή. Τα τοπικά αυτά υποδίκτυα θα περιέχουν μεν το συγκεκριμένο κόμβο Κ του αρχικού δικτύου, αλλά δεν θα έχουν καμία σύνδεση με τους υπόλοιπους κόμβους του αρχικού δικτύου. Στην περίπτωση αυτή ας ονομάσουμε το τοπικό δίκτυο Chord ως δίκτυο "Επιπέδου 1", στο οποίο ανήκει και ο κόμβος Κ, και τα επιμέρους, τοπικά, ανεξάρτητα του δικτύου Επιπέδου 1, υποδίκτυα Chord ως υποδίκτυα Επιπέδου 2. Στην πραγματικότητα, είναι σαν να προσθέτουμε μία δεύτερη διάσταση στο δίκτυο Chord. Ας δώσουμε ένα απλοϊκό παράδειγμα, που όμως μπορεί να φανταστεί κανείς ότι είναι δυνατόν να παρουσιαστεί στην πράξη. Στην περίπτωση που έχουμε πακέτα υλικών, σε ένα εμπορευματοκιβώτιο ή σε ένα βαγόνι τρένου, τα οποία έχουν σημαντική αξία ή έχουν κοντινό χρόνο λήξης της

καταλληλότητας των περιεχομένων τους, μπορεί να υπάρξει ανάγκη να δημιουργήσουμε παραπάνω από ένα τοπικά υποδίκτυα Επιπέδου 2, τα οποία θα έχουν ως κοινό τον κόμβο Κ του δικτύου Επιπέδου 1, αλλά δε θα επικοινωνούν απαραίτητα με άλλον κόμβο του δικτύου Επιπέδου 1. Προφανώς, θα χρειαστεί ειδικό λογισμικό για τον κόμβο Κ, ο οποίος θα είναι κοινός στα υποδίκτυα 1ου και 2ου επιπέδου. Μία δομή της περίπτωσης αυτής φαίνεται στο επόμενο σχήμα.



Το προηγούμενο παράδειγμα αφορά επέκταση του δικτύου Chord σε δεύτερη διάσταση. Θα μπορούσε κάλλιστα να προκύψει στην πράξη αργότερα ανάγκη δημιουργίας επιμέρους τοπικών δικτύων σε κάποιο κόμβο δεύτερης διάστασης, οπότε θα είχαμε ανάγκη σε τρίτη διάσταση κ.ο.κ.

Εκτός από αυτής της άποψης επέκταση ως προς τις διαστάσεις, άλλη μία προσέγγιση είναι η εξής. Αντί κάθε δίκτυο να είναι σε μορφή γραμμής, όπου υπάρχουν το πολύ δύο γειτονικοί κόμβοι, να έχουμε συνδέσεις και με κόμβους που είναι γειτονικοί σε άλλες διαστάσεις. Χαρακτηριστικό παράδειγμα είναι τα εμπορευματοκιβώτια που βρίσκονται σε πλοίο, όπου καθένα έχει γειτονικά έως και 6 άλλα εμπορευματοκιβώτια. Ένα σχηματικό παράδειγμα είναι το παρακάτω. Έχουν γραφεί τα “level 1” και “level 2” μόνο για κατανόηση του βάθους που υπάρχει στη δομή.



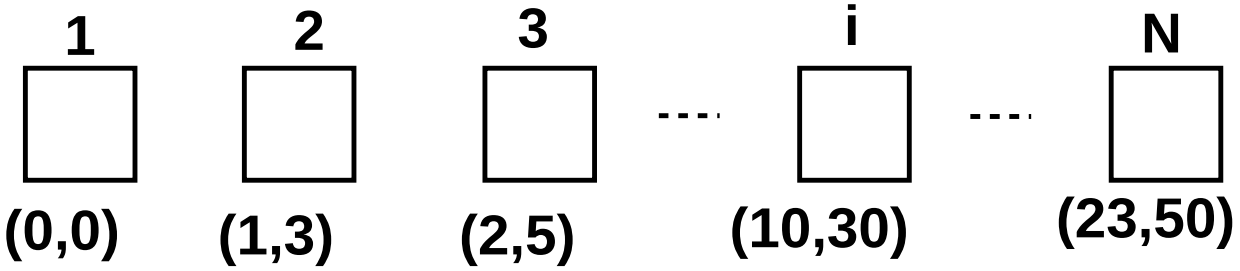
Αρχικά είχαμε το πολύ 2 γειτονικούς κόμβους. Έστω ότι όταν αυξήσουμε τις διαστάσεις θα υπάρχει τετραγωνική δομή, δηλαδή ένα κόμβος μπορεί να έχει γείτονες στις "ακμές" του τετραγώνου του, το οποίο φαίνεται λογικό, αφού γενικά οι μεταφορές γίνονται σε ορθογώνια παραλληλεπίπεδα σχήματα. Με αυτή τη λογική, αν πάμε στις δύο διαστάσεις, τότε κάθε κόμβος θα έχει μέχρι και 4 γειτονικούς κόμβους, ενώ αν πάμε στις τρεις διαστάσεις, θα έχει μέχρι και 6. Στο προηγούμενο σχήμα έχουμε τρεις διαστάσεις, και έχουμε αριθμήσει τα επίπεδα για ευκολία στη περιγραφή.

Παρατηρούμε ότι, στο προηγούμενο σχήμα, δεν είναι απαραίτητο να υπάρχουν και οι 6 συνδέσεις με όλους τους γείτονες, αλλά πρέπει να υπάρχουν αρκετές ώστε να υπάρχει μονοπάτι μεταξύ όλων των κόμβων. Έτσι, δημιουργείται μία λογική mesh δικτύου, όπου προκειμένου ένας κόμβος να μεταφέρει ένα πακέτο, βρίσκει σε ποιον κόμβο πρέπει να το στείλει ώστε (i) να υπάρχει μονοπάτι ανάμεσα στον αποστολέα και στον παραλήπτη κόμβο και (ii) να βρεθεί το μονοπάτι ελάχιστης απόστασης μέχρι τον κόμβο προορισμού (δηλαδή να υπάρξει ο μικρότερος αριθμός διαβάσεων ενδιάμεσων κόμβων).

8.3 Παράρτημα 3: Επέκταση κόμβων με περισσότερες δυνατότητες

Στην περίπτωση που είναι γραμμική η χωρική διάταξη των κόμβων, τότε μπορεί να σχεδιαστεί επέκταση της υλοποίησης που έγινε σε προηγούμενο κεφάλαιο, προσθέτοντας συσκευές εντοπισμού θέσης και ασύρματης επικοινωνίας μεγάλης εμβέλειας. Αυτές αναφέρονται σύντομα στα επόμενα.

Αρχικά, κάθε κόμβος μπορεί να έχει μία συσκευή GPS πάνω του. Αυτό θα έχει ως αποτέλεσμα να προκύπτει η αρίθμηση τους πιο εύκολα και άμεσα, δηλαδή αντί να γίνεται αυθαίρετα η αρίθμηση, να γίνεται σύμφωνα με τις συντεταγμένες που δίνει το κάθε GPS. Φαίνεται καλύτερα στο παρακάτω σχήμα.



Κάθε κόμβος έχει κάποιες συντεταγμένες. Αν θεωρήσουμε τις συντεταγμένες του κόμβου 1 ως αναφορά (οπότε μπορούμε να τις θέσουμε ίσες με (0,0)), τότε προκύπτουν και οι υπόλοιπες, απλά αφαιρώντας από αυτές τις συντεταγμένες του 1. Έτσι, έχουμε τώρα τις συντεταγμένες του κάθε κόμβου σε σχέση με τον κόμβο 1, και επακόλουθα γνωρίζουμε τις σχετικές θέσεις μεταξύ όλων. Σύμφωνα με αυτές τις συντεταγμένες, μπορεί να υπολογιστεί μία νόρμα, για παράδειγμα η πιο εύκολη θα είναι απλά η Ευκλείδεια νόρμα της απόστασης, για κάθε ζεύγος συντεταγμένων και να γίνεται μία σύγκριση μεταξύ των νορμών που προκύπτουν. Σύμφωνα με τις συγκρίσεις θα προκύπτει η τελική αρίθμηση των κόμβων. Στο σχήμα φαίνεται ότι ο κόμβος 2 έχει ευκλείδεια νόρμα ίση με περίπου 3.16 ενώ ο κόμβος 3 έχει περίπου 5.39. Αυτός είναι και ο λόγος που τους δόθηκαν οι αριθμήσεις 2 και 3. Ο κόμβος i έχει ακόμα μεγαλύτερη νόρμα και ο κόμβος N θα έχει τη μέγιστη νόρμα.

Αυτή η οργάνωση προσφέρει περισσότερες δυνατότητες στο δίκτυο, όπως αν μπαίνει νέος κόμβος ανάμεσα σε άλλους δύο, αυτό θα φαίνεται άμεσα χρησιμοποιώντας τις συντεταγμένες, οπότε θα προσαρμόζεται και το υπόλοιπο δίκτυο, ή αν κάποιου κόμβου οι συντεταγμένες αλλάξουν πολύ σε σχέση με τους γειτονικούς του, σημαίνει ότι συνέβη κάποιο μη προκαθορισμένο γεγονός, με αποτέλεσμα να έχει αποσυνδεθεί από το δίκτυο.

Άλλη συσκευή που θα ήταν χρήσιμη να υπάρχει σε κάθε κόμβο είναι συσκευή GSM. Με αυτήν, βαγόνια που έχουν πιο αυστηρές ανάγκες συντήρησης των περιεχομένων τους και ενημέρωσής τους, όπως για παράδειγμα βαγόνι-ψυγείο, θα μπορούν άμεσα να στέλνουν μήνυμα σε κάποιον υπεύθυνο αν τυχόν υπάρξει βλάβη ή ακραία τιμή όσον αφορά τη θερμοκρασία, την υγρασία κλπ.

8.4 Παράρτημα 4: Κώδικας για αυτοοργάνωση τοπικού δικτύου μεταφοράς που υλοποιήθηκε

Παρατίθεται, σε αυτό το σημείο, ο κώδικας που γράφηκε για την υλοποίηση του δικτύου που εξηγείται στο κεφάλαιο 5. Περιέχει όλα τα απαραίτητα σχόλια για την κατανόησή του. Έχει γραφεί σε γλώσσα προγραμματισμού C και χρησιμοποιήθηκαν βιβλιοθήκες σχετικές με αυτήν και με το Arduino και η βιβλιοθήκη για τον πομποδέκτη nRF24L01 συνδεδεμένο στο Arduino, η οποία περιγράφηκε σε προηγούμενο υποκεφάλαιο. Τα σχόλια γράφηκαν σε αγγλική γλώσσα, με σκοπό να είναι εύκολη, αν είναι επιθυμητή, η αντιγραφή του κώδικα σε ένα περιβάλλον ανάπτυξης προγραμμάτων, χωρίς να υπάρχουν προβλήματα μεταφοράς του κώδικα αυτού στο περιβάλλον λόγω χρήσης ελληνικών χαρακτήρων.

```
#include <SPI.h>
#include <RF24.h>
#include <string.h>
#include <printf.h>

//Type for packets to be sent and received
struct packetType{
  char type[8];
  //addresses at nRF24L01 have 40bits, so we make a vector of 5 8-bit indexes
  uint8_t senderNode[5];
  uint8_t receiverNode[5];
  //data is made to be a 8-index vector, just for testing purposes
```

```
  int dataSize;
  int data[8];
  int dataSizeEnd;  //dataSizeEnd = dataSize
};


//Initialize transceiver
RF24 radio(7,8);


//Global variables used

//Addresses:
//40bits addresses, only the first 8 bits really matter, so the [0] of the vectors.
//the rest is balanced 0s and 1s for sending to be successful.
uint8_t selfad[5] = {0x01, 0x0F, 0x0F, 0x0F, 0x0F};//self address
uint8_t broadcastad[5] = {0xFE, 0x0F, 0x0F, 0x0F, 0x0F};//address for broadcasts
uint8_t sendtoad[5] = {0xFF, 0x0F, 0x0F, 0x0F, 0x0F};//address for where to send the
packet


int inNW = 0;//if it's =1, then the node is in the network.
char serialRead;//used to keep the value from reading from the serial (keyboard).
int NWcounter = 0; //how many nodes there are
int printtemp = 0, printtemp1 = 0; //used so only printed once
unsigned long int lastRefreshedTime = 0; //used for time intervals for broadcasting invites
//general packet used to store and process received packets.


//Packets used:
//this is where the received packets will be stored
struct packetType packet = {"0000000", 0x00, 0x0F, 0x0F, 0x0F, 0x0F, 0x00, 0x0F,
0x0F, 0x0F, 0x0F, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0};
//packet used to send invitations at broadcastad address.
struct packetType broadPacket = {"invite", 0x01, 0x0F, 0x0F, 0x0F, 0x0F, 0x01, 0x0F,
0x0F, 0x0F, 0x0F, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1};


//FUNCTIONS
//-------------------
//Used to send packet
int sendPacket(struct packetType sPacket){
  //printPacket(sPacket);//print packet for checking
  int a;//this is the value to be returned

  //check where to send the packet
  if(sPacket.receiverNode[0] > selfad[0]) sendtoad[0] = selfad[0] + 1;
  else if(sPacket.receiverNode[0] < selfad[0]) sendtoad[0] = selfad[0] - 1;
  else sendtoad[0] = 0xFF;

  //send packet to sendtoad
  if(sendtoad[0] != 0xFF){
    radio.stopListening();
```

```
   radio.openWritingPipe(sendtoad);
   a = radio.write(&sPacket, sizeof(sPacket));
   a = radio.write(&sPacket, sizeof(sPacket));
   //radio.printDetails();
   delay(5);
   radio.closeReadingPipe(0);
   radio.startListening();
  }
  else a = -1;//if reached destination, a=-1


  if(a==0){//if write failed, assume sendtoad node left
   Serial.print(sendtoad[0]);
   Serial.println(" failed");


   //if it was to be sent to the right
   if(sendtoad[0] > selfad[0]){
    int oldNWcounter = NWcounter; //keep NWcounter to know how many consecutive
nodes left
    //temporarily change selfad
    selfad[0]++;
    //make leave packet and send it
    struct packetType leavepacket = {"leave", 0x00, 0x0F, 0x0F, 0x0F, 0x0F, 0x00, 0x0F,
    0x0F, 0x0F, 0x0F, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0};
    leavepacket.receiverNode[0] = NWcounter;
    leavepacket.senderNode[0] = selfad[0];
    leavepacket.dataSize=1;
    sendPacket(leavepacket);
    //go back to original selfad, to update left nodes
    selfad[0]--;
    NWcounter--;
    //only update nodes if sent message isn't leave, otherwise it's
    //consecutive leaves, and the last one will handle it all.
    if(strcmp(sPacket.type , "leave") != 0) updateNodes(0x01);
    //match destination with new receiver address.
    sPacket.receiverNode[0] = sPacket.receiverNode[0] - (oldNWcounter-NWcounter);
    a = sendPacket(sPacket);
   }


   //if it was to be sent to the left
   else if(sendtoad[0] < selfad[0]){
    //make leave packet and send it.
    struct packetType leavepacket = {"leave", 0x00, 0x0F, 0x0F, 0x0F, 0x0F, 0x00, 0x0F,
    0x0F, 0x0F, 0x0F, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0};
    leavepacket.receiverNode[0] = NWcounter;
    leavepacket.senderNode[0] = selfad[0];
    leavepacket.dataSize=1;
    sendPacket(leavepacket);
    //reduce self address by 1 to keep network correct.
    selfad[0]--;
    radio.stopListening();
```

```
      radio.openReadingPipe(1,selfad);
      radio.startListening();
      NWcounter--;
      updateNodes(0x01);//update the nodes to the left with the new NWcounter.
      sPacket.senderNode[0]--;//match destination with new receiver address.
      //if, after the network is fixed, the receiver node is higher than the
      //NWcounter, then set it to NWcounter.
      if(sPacket.receiverNode[0] > NWcounter) sPacket.receiverNode[0] = NWcounter;
      //if about to be sent packet is update packet, don't send it,
      //because above updateNodes gave the most recent and correct info.
      if(strcmp(sPacket.type , "update") != 0) a = sendPacket(sPacket);
    }
   }
   return a;
}
//------------------------------


//Used to inform every node from the node who sends it
//to the node with address recNode about the new NWcounter.
int updateNodes(uint8_t recNode){
  //make update packet
  struct packetType updatePacket = {"update", 0x00, 0x0F, 0x0F, 0x0F, 0x0F, 0x01, 0x0F,
  0x0F, 0x0F, 0x0F, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1};
  updatePacket.senderNode[0] = selfad[0];
  updatePacket.receiverNode[0] = recNode;
  updatePacket.data[0] = NWcounter;
  printPacket(updatePacket);
  //send it to recNode
  int a = sendPacket(updatePacket);
  return a;
}
//------------------------------


//Used to print packet's content.
void printPacket(struct packetType pPacket){
  Serial.println("PRINT PACKET:");
  Serial.print("Packet type: ");
  Serial.println(pPacket.type);
  Serial.print("Sender node: ");
  Serial.println(pPacket.senderNode[0]);
  Serial.print("Receiver node: ");
  Serial.println(pPacket.receiverNode[0]);
  Serial.print("Data size: ");
  Serial.println(pPacket.dataSize);
  Serial.print("Data: ");
  for(int i=0; i<pPacket.dataSize; i++){
    Serial.print(pPacket.data[i]);
    Serial.print(" ");
  }
  Serial.println("");
}
```

```
//----------------------------


//Main program starts here.
void setup() {
  //Begin serial connection through USB
  Serial.begin(9600);
  printf_begin();

  Serial.println("Type a letter to begin:");
  //wait for command (just a random key from the keyboard) to start
  while(!Serial.available());
  Serial.read();
  Serial.println("Turned ON. Listening for invitations for 5 sec...");
  radio.begin();
  //set power to low, since we are testing
  //we don't need any higher that tires the transceiver.
  radio.setPALevel(RF24_PA_LOW);
  //start listening for broadcasts
  radio.openReadingPipe(1, broadcastad);
  radio.startListening();
  unsigned long int miltimer = millis();


  while(millis()-miltimer<5000 && inNW==0){//listen for 5 seconds for broadcasts.
    if(radio.available()){
      delay(5);
      Serial.println("Heard broadcast. Trying to join...");
      radio.read(&packet, sizeof(packet));
      radio.read(&packet, sizeof(packet));

      //If it's an actual invitation packet, do all the procedures needed.
      //Has to be type "invite" to be an actual invitation.
      if(strcmp(packet.type, "invite")==0)
      {
        Serial.print("Sender node = ");
        Serial.println(packet.senderNode[0]);
        selfad[0] = packet.senderNode[0] + 1;
        Serial.print("Selfad = ");
        Serial.println(selfad[0]);
        //Inform every node for new node in NW.
        NWcounter = selfad[0];
        Serial.println("Joined. Updating info on all nodes.");
        radio.stopListening();
        radio.openReadingPipe(1,selfad);
        //radio.closeReadingPipe(2);
        radio.startListening();
        int b = updateNodes(0x01);
        packet.dataSize=0;
        Serial.print("Update sends success = ");
        Serial.println(b);
        lastRefreshedTime = millis();
```

```
      inNW = 1;
     }

   }
  }

  //If reached this line with inNW==0,
  //it means they didn't receive anything and 5 seconds passed
  //so they assume they are the 1st node in the network.
  if(inNW==0){
   Serial.println("1st node in NW, initializing...");
   selfad[0] = 0x01;
   radio.stopListening();
   radio.closeReadingPipe(1);
   radio.openReadingPipe(1,selfad);
   radio.startListening();
   lastRefreshedTime = millis();
   inNW = 1;
   NWcounter = 1;
  }
  //radio.printDetails();
  Serial.print("In the network: ");
  Serial.println(selfad[0]);
}



//When in here, the node is in the network
void loop() {

  //If last node, send invitation every 3 sec.
  if(millis()-lastRefreshedTime > 3000 && selfad[0] == NWcounter)
  {
   //make invite packet
   broadPacket.senderNode[0] = selfad[0];
   Serial.println(broadPacket.type);
   Serial.print("Selfad = ");
   Serial.println(selfad[0]);
   //broadcast it
   radio.stopListening();
   //radio.setAutoAck(0);
   radio.openWritingPipe(broadcastad);
   radio.write(&broadPacket, sizeof(broadPacket));
   radio.write(&broadPacket, sizeof(broadPacket));
   delay(5);
   radio.closeReadingPipe(0);
   //radio.setAutoAck(1);
   radio.startListening();
   lastRefreshedTime = millis();
  }


  //If a packet was just received, process it and forward it, if needed.
```

```
if(packet.dataSize!=0)
{

  //case "update" packet
  if(strcmp(packet.type, "update")==0)
  {
    NWcounter = packet.data[0];
    Serial.println(packet.type);
    Serial.print("No. of nodes = ");
    Serial.println(NWcounter);
    int a = sendPacket(packet);
    Serial.print("Send success: ");
    Serial.println(a);
  }

  //case "leave" packet
  else if(strcmp(packet.type, "leave")==0)
  {
    sendPacket(packet);
    NWcounter--;
    selfad[0]--;
    radio.stopListening();
    radio.openReadingPipe(1,selfad);
    radio.startListening();
  }

  //case "transit" packet
  else if(strcmp(packet.type, "transit")==0)
  {
    if(sendPacket(packet) == -1){
      Serial.println("Transit reached destination.");
      printPacket(packet);
    }
    else{
      Serial.println("Transit forwarded.");
      printPacket(packet);
    }
  }
  packet.dataSize=0;//Clear data. Just needed to set dataSize=0.
}

//If there is no packet or serial, print that you are waiting.
else{
  if(printtemp1==0){
    Serial.print("Waiting: ");
    Serial.println(selfad[0]);
    printtemp1=1;
  }

  //If there is nothing to listen to, try to read from serial.
  if(!radio.available()){

    //If there is something in serial, print NWcounter and self address,
```

```
//then proccess it if needed
if(Serial.available()){
  serialRead = Serial.read();
  printtemp1=0;
  Serial.print("NWcounter = ");
  Serial.println(NWcounter);
  Serial.print("Selfad = ");
  Serial.println(selfad[0]);


  if(serialRead == 'L'){//case "leave"
    Serial.print("Leaving the network, selfad = ");
    Serial.println(selfad[0]);
    serialRead=0;
    //make leave packet and send to highest node
    strcpy(packet.type, "leave");
    packet.senderNode[0] = selfad[0];
    packet.receiverNode[0] = NWcounter;
    packet.dataSize=1;
    sendPacket(packet);
    printPacket(packet);
    NWcounter--;
    //update the lower nodes
    updateNodes(0x01);
    packet.dataSize=0;
    //turn off
    radio.powerDown();
    while(1);//stay here forever
  }


  else if(serialRead == 'T'){//case send "transit"
    //read packet data, make it and send it.
    Serial.println("MAKING TRANSIT PACKET");
    strcpy(packet.type, "transit");
    Serial.print("Selfad = ");
    Serial.println(selfad[0]);
    Serial.print("NWcounter = ");
    Serial.println(NWcounter);
    packet.senderNode[0] = selfad[0];
    Serial.print("Give Receiver Ad: ");
    while(!Serial.available());
    packet.receiverNode[0] = Serial.read() - '0';
    Serial.println(packet.receiverNode[0]);
    Serial.print("Give data size (up to 8): ");
    while(!Serial.available());
    packet.dataSize = Serial.read() - '0';
    Serial.println("Give data:");
    for(int i=0; i<packet.dataSize; i++){
      Serial.print(i);
      Serial.print(": ");
      while(!Serial.available());
      packet.data[i] = Serial.read() - '0';
```

```
          Serial.println(packet.data[i]);
        }
        Serial.println("TRANSIT PACKET MADE, INFO");
        printPacket(packet);
        sendPacket(packet);
        packet.dataSize = 0;
      }


      else if(serialRead == 'P'){//case "print details"
        radio.printDetails();
      }
    }
  }

  //If heard something from the selfad pipe
  else{
    delay(5);
    printtemp1=0;
    Serial.println("Heard something");
    radio.read(&packet, sizeof(packet));
    radio.read(&packet, sizeof(packet));
    Serial.println(packet.type);
  }
 }
}
```

9 Αναφορές

[1] Elias Koukoutsis, Constantin Papaodysseus, Nikolaos V. Karadimas, Athanasios Ballis, Designing a flexible, highly adaptable and gracefully expandable information system for the implementation of complex decision support systems, National Technical University of Athens, 9 Heroon Polytechneiou, Zografou Campus, 157 80 Athens, Greece
[2] Τσακαλίδου Παναγιώτα, Κουκούτσης Ηλίας, Μπαλλής Θεοχάρης, Μπαλλής Αθανάσιος, Φωτόπουλος Ευάγγελος, Μελέτη και ανάπτυξη δυναμικών δομών περιγραφής της υποδομής μεταφορών για την αποτελεσματική εφαρμογή υπολογιστικών μοντέλων
[3] A. Ballis, E. Koukoutsis, I. Lagou, S. Zannos, F. Giannopoulos, Research towards the finalization of European Transport Information System (ETIS)
[4] ΓΑ. Στέκος, Μια νέα μεθοδολογία για τη διαχείριση πολύπλοκων, πολυθεματικών και μεγάλης κλίμακας συστημάτων γεωσυσχετισμένων πληροφοριών
[5] Elias Koukoutsis, Athanasios Ballis, Yiannis Koukoutsis, Anastasios Koutoumanos, The european transport-policy information system (ETIS): Overview and assessment of the GIS component, National Technical University of Athens, 9 Heroon Polytechniou, Zografou Campus, 157 80 Athens, Greece
[6] Lt. Col. A. Stekos, Cpt G. Tsavdaris, Cpt Th. Papageorgiou, Asst. Prof. Dr. E. Koukoutsis, Achieving Interperability in Large Collections of Georeference Data Sets used in Crises Management Systems and in Systems Supporting the Design, Simulation and Control of Coplex Operational Procedures
[7] ATmega328/P Datasheet Complete 2016 November http://www.atmel.com/Images/Atmel-42735-8-bit-AVR-Microcontroller-ATmega328-328P_Datasheet.pdf
[8] Arduino Uno https://www.farnell.com/datasheets/1682209.pdf
[9] Geekcreit UNO R3 ATmega328P Development Board For Arduino, https://www.banggood.com/UNO-R3-ATmega328P-Development-Board-For-Arduino-No-Cable-p-964163.html
[10] ARDUINO UNO REV3, https://store.arduino.cc/arduino-genuino/arduino-genuino-boards-modules/arduino-uno-rev3
[11] Datasheet Raspberry Pi Compute Module 2016 October https://www.raspberrypi.org/documentation/hardware/computemodule/RPI-CM-DATASHEET-V1_0.pdf
[12] Raspberry Pi 3 Model B ARM Cortex-A53 CPU 1.2GHz 64-Bit Quad-Core 1GB RAM 10 Times B+, http://www.banggood.com/Raspberry-Pi-3-Model-B-ARM-Cortex-A53-CPU-1_2GHz-64-Bit-Quad-Core-1GB-RAM-10-Times-B-p-1041862.html
[13] Raspberry pi zero, https://shop.pimoroni.com/products/raspberry-pi-zero
[14] BeagleBone Black, https://www.sparkfun.com/products/12857
[15] pcDuino8 Uno, https://www.amazon.com/pcDuino8-Uno-performance-Single-Computer/dp/B016B4EIFQ
[16] freetronics Goldilocks, https://www.freetronics.com.au/products/goldilocks-arduino-compatible-with-atmega1284p-mcu#.WPiET4h96t8
[17] Teensy 3.6, https://www.sparkfun.com/products/14057
[18] Intel Edison, https://www.adafruit.com/product/2180
[19] Geekcreit DUE R3 32 Bit ARM With USB Cable Arduino Compatible, https://www.banggood.com/Arduino-Compatible-DUE-R3-32-Bit-ARM-With-USB-Cable-p-906466.html
[20] Arduino Due, https://store.arduino.cc/arduino-due

[21] Geekcreit ATmega328P Arduino Compatible Nano V3 Improved Version, https://www.banggood.com/ATmega328P-Arduino-Compatible-Nano-V3-Improved-Version-With-USB-Cable-p-933647.html
[22] Arduino Nano, https://store.arduino.cc/arduino-nano
[23] NRF24L01L Long Range Wireless Module R2 (1.1KM, PA+SMA), http://www.electrodragon.com/product/1000-meters-nrf24l01-long-distance-module/
[24] Optimized High Speed NRF24L01+ Driver Class Documenation: Optimized High Speed Driver for nRF24L01(+) 2.4GHz Wireless Transceiver, http://tmrh20.github.io/RF24/index.html
[25] NRF24L01 2.4GHz Wireless Transceiver Module, https://www.banggood.com/Wholesale-Perfect-High-Quality-New-NRF24L01-2_4GHz-Wireless-Transceiver-Module-Arduino-p-41612.html?rmmds=search
[26] UM801 passthrough (UART) SX1276 wireless module LoRa, http://www.sunhokey.com/product/60370293327-219301501/UM801_passthrough_UART_SX1276_ultra_long_range_6km_862_1020M_wireless_module_LoRa.html
[27] RFM95 wireless transceiver module/LoRa spread spectrum communication 868/915mhz SX1276, https://www.aliexpress.com/store/product/RFM95-wireless-transceiver-module-LoRa-spread-spectrum-communication-868-915mhz-SX1276-16-16mm/1361444_32706857688.html
[28] RFM95/96/97/98 Low power Long range transceiver module, https://github.com/SeeedDocument/RFM95-98_LoRa_Module/blob/master/RFM95_96_97_98_DataSheet.pdf
[29] Build LoRa/LoRaWAN gateway, https://www.youtube.com/watch?v=ZFVA6cQyheY
[30] LoRa Node with Arduino and Dragino Shield connected to TTN LoRaWAN, https://www.youtube.com/watch?v=duwUwXt-hs8
[31] Adafruit feather 32u4 RFM95 LoRa radio, https://www.adafruit.com/product/3078
[32] Digi XBee-PRO s2c ZigBee, https://www.digikey.com/product-detail/en/digi-international/XBP24CZ7WIT-004/602-1564-ND/5322378?WT.z_cid=ref_neda_dkc_buynow_digiintl
[33] Digi XBee-PRO s2c ZigBee specifications, https://www.digi.com/products/digi-xbee-rf-solutions/embedded-rf-modules-modems/digi-xbee-zigbee#specifications
[34] Xbee ZigBee s2c, https://www.digikey.com/product-detail/en/digi-international/XB24CZ7WITB003/602-1858-ND/5322880?WT.z_cid=ref_neda_dkc_buynow_digiintl
[35] RFM12B, http://www.tme.eu/en/details/rfm12b-868s2p/rf-communication-modules/hope-microelectronics/
[36] RFM12B Datasheet https://www.tme.eu/en/Document/e38d805f9a476a416522af89296a370e/RFM12B-433DP.pdf
[37] RFM12B Library, https://github.com/LowPowerLab/RFM12B
[38] RFM69HW, http://uk.rs-online.com/web/p/lower-power-rf-modules/7931992/
[39] RFM69HW ISM TRANSCEIVER MODULE V1.3 2006 http://www.hoperf.com/upload/rf/RFM69HW-V1.3.pdf
[40] RFM69 Library, https://github.com/LowPowerLab/RFM69
[41] LoRa Architecture, http://www.3glteinfo.com/lora/lora-architecture/
[42] N. Sornin (Semtech), M. Luis (Semtech), T. Eirich (IBM), T. Kramp (IBM), O.Hersent (Actility) LoRaWAN Specification 2015 January https://www.rs-online.com/designspark/rel-assets/ds-assets/uploads/knowledge-items/application-notes-for-the-internet-of-things/LoRaWAN%20Specification%201R0.pdf
[43] ALOHAnet, https://en.wikipedia.org/wiki/ALOHAnet
[44] Spread spectrum, https://en.wikipedia.org/wiki/Spread_spectrum
[45] Chirp spread spectrum, https://en.wikipedia.org/wiki/Chirp_spread_spectrum

[46] Phase-shift keying, https://en.wikipedia.org/wiki/Phase-shift_keying#Quadrature_phase-shift_keying_.28QPSK.29
[47] Frequency-shift keying, https://en.wikipedia.org/wiki/Frequency-shift_keying
[48] Zigbee® Wireless Standard - Digi International, https://www.digi.com/resources/standards-and-technologies/rfmodems/zigbee-wireless-standard
[49] Minimum-shift keying, https://en.wikipedia.org/wiki/Minimum-shift_keying
[50] On-off keying, https://en.wikipedia.org/wiki/On-off_keying
[51] Understanding Wireless Range Calculations, http://www.electronicdesign.com/communications/understanding-wireless-range-calculations
[52] Friis transmission equation, https://en.wikipedia.org/wiki/Friis_transmission_equation
[53] DBm, https://en.wikipedia.org/wiki/DBm
[54] Decibel, https://en.wikipedia.org/wiki/Decibel#Antenna_measurements
[55] Receiver sensitivity, http://www.radio-electronics.com/info/rf-technology-design/rf-noise-sensitivity/receiver-sensitivity-performance-tutorial.php
[56] Antenna gain, https://en.wikipedia.org/wiki/Antenna_gain
[57] Isotropic radiator, https://en.wikipedia.org/wiki/Isotropic_radiator
[58] Link budget, https://en.wikipedia.org/wiki/Link_budget
[59] Understanding Asynchronous Dual-Port RAMs http://www.cypress.com/file/46721/download
[60] IDT Asynchronous Dual-Port RAMs, https://www.idt.com/products/memory-logic/multi-port-memory/asynchronous-dual-port-rams/?field-bus-width-bits=18%2C36&field-architecture=Dual-Port&method-field-architecture=OR
[61] Digi Asynchronous Dual-Port SRAM, https://www.digikey.com/catalog/en/partgroup/asynchronous-dual-port-sram/1515?mpart=7028L15PF&vendor=800
[62] Cypress Dual-Port SRAMs, http://www.cypress.com/search/psg/1243#/?_facetShow=fs_pdensity_kb_,ss_porganization_x_x_y_,fs_pspeed_ns_,ss_pfrequency_mhz_,fs_pmin_operating_voltage_v_,fs_pmax_operating_voltage_v_,ss_ppackage,fs_pmin_operating_vccq_v_,fs_pmax_operating_vccq_v_,ss_ptemp_classification,fs_part_price
[63] IDT7028 high speed 64Kx16 dual-port static ram, https://www.digikey.com/product-detail/en/7028L15PF/800-1368-ND/1915669?curr=usd&WT.z_cid=ref_octopart_dkc_buynow&site=us
[64] IDT7130LA35PDG high speed 1Kx8 dual-port static sram, https://www.digikey.com/products/en?mpart=7130LA35PDG&v=800
[65] IDT7130LA100PDG, https://www.digikey.com/products/en?mpart=7130LA100PDG&v=800
[66] IDT7132LA100PDG, https://www.digikey.com/products/en?mpart=7132LA100PDG&v=800
[67] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, Hari Balakrishnan, Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications 2001 University of California, Berkley
[68] Παρουσίαση του Chord, https://github.com/KostasKoutrou/Diplwmatikh/blob/master/ChordPresentationKoutroumpouchos.pdf
[69] Freenet, https://en.wikipedia.org/wiki/Freenet
[70] Loren P. Clare(a), Gregory J. Pottie(b) and Jonathan R. Agrea, Self-Organizing Distributed Sensor Networks (a)Rockwell Science Center Thousand Oaks, CA 91360 USA (b)Electrical Engineering Department, University of California Los Angeles, Los Angeles, CA 90095 USA
[71] Katayoun Sohrabi, Jay Gao, Vishal Ailawadhi and Gregory J Pottie, Protocols for Self-Organization of a Wireless Sensor Network 1999 University of California, Los Angeles

[72] Mobile ad hoc network, https://en.wikipedia.org/wiki/Mobile_ad_hoc_network
[73] Ανάγκες ειδικού χειρισμού γεωσυσχετισμένης πληροφορίας σε Συστήματα Διαχείρισης Κρίσεων και παρακολούθησης και ελέγχου επιχειρησιακών χειρισμών, Γεώργιος Τσαβδαρίδης
[74] Κουκούτσης Ηλίας, Μπαμπαλής Χαράλαμπος, Μπαλλής Αθανάσιος, Ανάπτυξη συστήματος διαχείρισης στόλου οχημάτων σε περιβάλλον GIS, 25η ΠΑΝΕΛΛΗΝΙΑ ΣΥΝΑΝΤΗΣΗ ΧΡΗΣΤΩΝ ArcGIS, Μάιος 2017
[75] Single chip 2.4 GHz Transceiver nRF24L01 2006 March https://www.sparkfun.com/datasheets/Components/nRF24L01_prelim_prod_spec_1_2.pdf
[76] Serial Peripheral Interface Bus, https://en.wikipedia.org/w/index.php?title=Serial_Peripheral_Interface_Bus&oldid=805780714
[77] Κώδικας της υλοποίησης δικτύου αυτοοργάνωσης, https://github.com/KostasKoutrou/Diplwmatikh/blob/master/SelfOrganizeNRF24_6thIteration/SelfOrganizeNRF24_6thIteration.ino