Πανεπιστημιο Πατρων
Πολυτεχνικη Σχολη
Τμημα Μηχανικων Ηλεκτρονικων Υπολογιστων και Πληροφορικης

Μελέτη και Υλοποίηση Πλατφόρμας Ανάπτυξης Παιχνιδιών Περιρρέουσας Υπολογιστικής Νοημοσύνης με Χρήση Ασύρματων Δικτύων Αισθητήρων

Μάριος Λογαράς
Α.Μ. 3280

Επιβλέπων: Καθηγητής Παύλος Σπυράκης
Συνεπιβλέπων: Δρ. Ιωάννης Χατζηγιαννάκης

Πάτρα
Νοέμβριος 2010

(η σελίδα έχει αφεθεί σκοπίμως κενή)

Περιεχόμενα

Περίληψη

Στην παρούσα διπλωματική παρουσιάζεται ο σχεδιασμός, η μελέτη και υλοποίηση μιας πλατφόρμας ανάπτυξης παιχνιδιών περιρρέουσας Υπολογιστικής Νοημοσύνης με Χρήση Ασύρματων Δικτύων Αισθητήρων. Τέτοιου είδους παιχνίδια εξελίσσονται στο φυσικό χώρο, οι παίκτες –συνήθως σε μεγάλο αριθμό– αλληλεπιδρούν μεταξύ τους και με το περιβάλλον μέσω των κινήσεών τους και της παρουσίας τους ή οχι στο χώρο χρησιμοποιώντας συσκευές αισθητήρων. Η υλοποίηση του εν λόγω συστήματος κάνει χρήση Sun SPOTS για το σκοπό αυτό. Παρουσιάζεται ένα σύνολο αρχών που τέτοιου είδους παιχνίδια πρέπει να ακολουθούν καθώς και ένα σύνολο μηχανισμών που ικανοποιούν τις αρχές αυτές. Ένας αριθμός υλοποιημένων παιχνιδιών με τη χρήση της πλατφόρμας αυτής χρησιμοποιείται για την αξιολόγηση και πιστοποίηση της ορθής λειτουργίας του συστήματος.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον κ. Παύλο Σπυράκη, Καθηγητή του Τμήματος Ηλεκτρονικών Υπολογιστών και Πληροφορικής και επιβλέποντα καθηγητή μου για την εμπιστοσύνη που μου έδειξε και την επιστημονική καθοδήγηση που παρείχε σ' εμένα καθ' όλη τη διάρκεια της εκπόνησης της εργασίας.

Επίσης, θέλω να ευχαριστήσω τον Δρ. Ιωάννη Χατζηγιαννάκη που μου έδωσε την ευκαιρία να ανακαλύψω ενδιαφέροντες τομείς έρευνας και να ασχοληθώ με αυτούς. Η στήριξή του, η ενθάρρυνση και η φιλική καθοδήγηση δημιούργησαν τις απαραίτητες συνθήκες για την ολοκλήρωση της διπλωματικής αυτής εργασίας.

Τέλος, ευχαριστώ θερμά τον Ορέστη Ακριβόπουλο για τη φιλική του συμβολή.

Κεφάλαιο 1

Εισαγωγή

1.1 Κίνητρο και Σημασία

Τα τελευταία χρόνια παρατηρείται μια συνεχώς αυξανόμενη τάση για την συνένωση του φυσικού με τον ψηφιακό κόσμο. Ιδιαίτερα σημαντικό ρόλο στη διαδικασία αυτή έχει η ερεύνα στον τομέα των ασύρματων δικτύων αισθητήρων καθώς και η εισαγωγή αυτών σε ένα πλήθος διαφορετικών τομέων εφαρμογών.

Παρά την τάση αυτή στη χρήση των αισθητήρων σε όλο και περισσότερους τομείς της καθημερινής ζωής, οι εφαρμογές στον τομέα της ψυχαγωγίας παραμένουν σε μικρό αριθμό. Ο χώρος των Παιχνιδιών λίγη προσοχή έχει λάβει από την κοινότητα των Ασύρματων Δικτύων Αισθητήρων. Η έρευνα και η βιβλιογραφία σχετικά με διαδραστικά παιχνίδια πολλών παικτών όπου οι χρήστες έχουν μαζί τους συσκευές αισθητήρων προκειμένου να αλληλεπιδράσουν μεταξύ τους αλλά και με το περιβάλλον είναι ακόμα περιορισμένη.

Ο τομέας των παιχνιδιών γενικότερα κατείχε παραδοσιακά ένα μεγάλο μερίδιο στη βιομηχανία των υπολογιστών και αποτελεί μέσο και κίνητρο για την εξέλιξη των τεχνολογιών σε επίπεδο υλικού και λογισμικού. Μια νέα τάση στο χώρο περιλαμβάνει χρήση κινητών συσκευών με κάποιου είδους αισθητήρα εγκαταστημένο και εκμεταλλεύεται το σύνολο των τεχνολογιών που οι συσκευές αυτές εισάγουν. Τα κινητά τηλέφωνα με τέτοιες δυνατότητες αυξάνονται συνεχώς δημιουργώντας ένα νέο τομέα εφαρμογών και ένα ισχυρό κοινό για μια νέα γενιά παιχνιδιών.

Ένας ακόμα σημαντικός κλάδος που επηρεάστηκε από την εισαγωγή νέων διαδραστικών μέσων ήταν αυτός των ηλεκτρονικών παιχνιδιών. Η βιομηχανία των ηλεκτρονικών παιχνιδιών αποτελεί τα τελευταία 40 χρόνια ένα από τα πιο δυναμικά κομμάτια του χώρου της Πληροφορικής. Ανάμεσα στις απόπειρες να σπάσει το "κατεστημένο" και να οδηγηθούμε σε νέους τρόπους ψυχαγωγίας και αλληλεπίδρασης με τον υπολογιστή, βρίσκεται το Wii [26] της Nintendo, το οποίο μέχρι σήμερα έχει πουλήσει περίπου 71 εκατομμύρια μονάδες. Αυτό

που κατάφερε ήταν να εισάγει νέους τρόπους αλληλεπίδρασης με τον υπολογιστή, αντικαθιστώντας το πληκτρολόγιο και το ποντίκι με μια συσκευή που αναγνωρίζει τις κινήσεις μας στην πραγματική ζωή, π.χ., την κίνηση του μπαστουνιού σε ένα παιχνίδι γκολφ. Το παράδειγμα της της Nintendo ακολουθούν σήμερα η Sony και η Microsoft. Η Sony παρουσίασε το PlayStation Move μία συσκευή ανίχνευσης των κινήσεων του παίκτη για το PlayStation 3 - PS3, που αναπαράγει στην οθόνη τις κινήσεις του παίκτη και μπορεί να χρησιμοποιηθεί επίσης ως εικονικό σπαθί, ρακέτα, κτλ. Επιπλέον, το 2010 η Microsoft παρουσίασε το Kinect [27] για την παιχνιδομηχανή της Xbox 360. Το Kinect έχει ενσωματωμένες κάμερες που επιτρέπουν την ανίχνευση των κινήσεων του χρήστη στο χώρο, οι οποίες αντιστοιχίζονται σε κινήσεις του παιχνιδιού, χωρίς να χρειάζεται αυτός να κρατά κάποια συσκευή ανίχνευσης κίνησης. Η τεχνολογία στην οποία βασίζεται το Kinect της Microsoft, είναι αυτή της καταγραφής και ανάλυσης βίντεο σε πραγματικό χρόνο με τέτοιον τρόπο ώστε να καταγραφούν οι κινήσεις που κάνει ένας παίκτης ή να εντοπιστούν άλλα στοιχεία της εικόνας (πρότυπα), όπως π.χ., τα πρόσωπα που εμφανίζονται στο βίντεο, συγκεκριμένα αντικείμενα (ένα αυτοκίνητο), κ.α.

Ένας άλλος κλάδος που επηρεάστηκε από αυτή την τάση για νέα μέσα αλληλεπίδρασης ανθρώπου-υπολογιστή είναι αυτός της κινητής τηλεφωνίας. Το iPhone, το κινητό τηλέφωνο της Apple, βγήκε στην αγορά τον Ιούνιο του 2007 και ήταν η πρώτη συσκευή που έκανε ευρέως γνωστή την τεχνολογία των επιφανειών πολλαπλής αφής, αλλάζοντας ριζικά την εικόνα των κινητών τηλεφώνων που μέχρι τότε χρησιμοποιούσαν οθόνες μονής επαφής (single touch).

Οι αλλαγές και εξελίξεις αυτές έχουν οδηγήσει στην δημιουργία ενός νέου είδους παιχνιδιών. Οι όροι Pervasive Gaming και Ubiquitous Gaming (Παιχνίδια Περιρρέουσας Υπολογιστικής Νοημοσύνης) έχουν προκύψει για να περιγράψουν τους νέους τρόπους διεπαφής που βασίζονται σε φυσικές κινήσεις των παικτών, λαμβάνουν χώρα σχεδόν οπουδήποτε χώρο και φυσικά εμπεριέχουν όλες τις τεχνολογίες που αναφέρθηκαν.

1.2 Στόχοι Διπλωματικής Εργασίας

Στόχος της παρούσας διπλωματικής εργασίας είναι η μελέτη και ανάπτυξη μια πλατφόρμας που επιτρέπει την ανάπτυξη και προσφέρει την εμπειρία χρήσης παιχνίδιών περιρρέουσας υπολογιστικής νοημοσύνης μέσω ασύρματων δικτύων αισθητήρων ενώ προτείνει απαντήσεις σε σημαντικά θέματα τεχνολογίας και υλοποίησης. Με ποιόν τρόπο είναι δυνατή η συμμετοχή μεγάλου πλήθους παικτών σε ένα παιχνίδι; Πώς μπορούν να αξιοποιηθούν πολλαπλές και διαφορετικές μέθοδοι αλληλεπίδρασης; Είναι δυνατή η δημιουργία παιχνιδιών που δεν βασίζονται σε κάποια κεντρική υποδομή; Η ανάπτυξη και η αξιολόγηση της

πλατφόρμας Fun In Numbers (αποδίδεται ως: Η Διασκέδαση εν τη Ενώσει) προσπαθεί δώσει απαντήσεις και να προτείνει τρόπους αντιμετώπισης αυτών των ζητημάτων συνδυάζοντας την τεχνολογία και την τεχνογνωσία των Ασύρματων Δικτύων Αισθητήρων.

1.3 Δομή Διπλωματικής Εργασίας

Το κείμενο που ακολουθεί διαρθρώνεται ως εξής. Στο Κεφάλαιο 2.1 γίνεται μια γενική αναφορά στα Παιχνίδια Περιρρέουσας Υπολογιστικής Νοημοσύνης και παρουσιάζονται κάποιες ήση υπάρχουσες, ενδεικτικές υλοποιήσεις τέτοιων. Στο Κεφάλαιο 2.1.1 αναλύεται ένα σύνολο Αρχών οι οποίες διέπουν τα παιχνίδια αυτού του τύπου. Στη συνέχεια, και στο Κεφάλαιο 2.2 γίνεται μια παρουσίαση των Ασύρματων Δικτύων Αισθητήρων: αναφέρονται οι σημαντικότεροι αισθητήρες σαν συσκευές με τα χαρακτηριστικά τους (Κεφάλαιο 2.2.1), οι εφαρμογές για τις οποίες χρησιμοποιούνται παραδοσιακά τα Ασύρματα Δίκτυα Αισθητήρων (Κεφάλαιο 2.2.2) καθώς και γίνεται λόγος για τα παιχνίδια που κάνουν χρήση αισθητήρων γενικότερα (Κεφάλαιο 2.2.3).

Το Κεφάλαιο 3.1.3 αφορά στην πλατφόρμα Fun In Numbers και η Αρχιτεκτονική της αναλύεται στο Κεφάλαιο 3.2. Στο Κεφάλαιο 3.1 γίνεται η παρουσίαση των συσκευών Sun SPOT που χρησιμοποιήθηκαν ενώ στα κεφάλαια 3.3.1 μέχρι 3.3.6 παρουσιάζονται 6 παιχνίδια που υλοποιήθηκαν με την εν λόγω πλατφόρμα. Τα βασικότερα πρωτόκολλα που συνθέτουν την πλατφόρμα FinN αναλύονται στο Κεφάλαιο 4 παρουσιάζοντας το πρόβλημα που καλούνται να επιλύσουν και τον τρόπο με τον οποίο το κάνουν.

Τέλος, καταγράφονται τα συμπεράσματα που προκύπτουν από την Εργασία αυτή καθώς και οι στόχοι για το μέλλον (Κεφάλαιο 5).

Κεφάλαιο 2

Αισθητήρες και Παιχνίδια

2.1 Παιχνίδια Περιρρέουσας Υπολογιστικής Νοημοσύνης

Ο όρος της περιρρέουσας υπολογιστικής νοημοσύνης (Ubiquitous Computing) [1], [2] περιγράφει ένα μοντέλο διεπαφής ανθρώπου-υπολογιστή κατά το οποίο η επεξεργασία πληροφορίας λαμβάνει χώρα σε καθημερινές δραστηριότητες και αντικείμενα. Η εφαρμογή του μοντέλου αυτού περιλαμβάνει ταυτόχρονη λειτουργία πολλών υπολογιστικών συσκευών και συστημάτων ενώ οι χρήστες δεν γνωρίζουν κατ' ανάγκη την ύπαρξη τους.

Τα τελευταία χρόνια τα Παιχνίδια Περιρρέουσας Υπολογιστικής Νοημοσύνης (*ΠΠΥΝ*) γνωρίζουν ιδιαίτερη ανάπτυξη σαν ένα νέο είδος παιχνιδιών. Παραδείγματα αυτού του νέου είδους είναι τα γεωγραφικά παιχνίδια (geotagging games) , παιχνίδια επαυξημένης πραγματικότητας (mixed/augmented reality games) [4] , παιχνίδια πόλης (urban games) και πολλά άλλα. Τα ΠΠΥΝ βασίζονται σε μεγάλο βαθμό σε έννοιες όπως η κίνηση των παιχτών, η παρουσία τους ή όχι σε προκαθορισμένο χώρο καθώ και σε άλλες αισθητηριακές μορφές εισόδου (όπως θερμοκρασία, φωτεινότητα). Σε τέτοια παιχνίδια οι παίχτες αλληλεπιδρούν ο ένας με τον άλλον καθώς και με το περιβάλλον γύρω τους με το να κινούνται και να κάνουν φυσικές χειρονομίες. Οι φυσικές αυτές ενέργειες αντιστοιχούν σε ενέργειες του παιχνιδιών.

Η ερευνητική δραστηριότητα στο χώρο αυτό έχει αρχίσει να αναπτύσσεται τα τελευταία χρόνια. Το project IPerG [3] (Intergrated Project for Pervasive Games) εστίασε από το 2004 μέχρι το 2008 στη έρευνα και τη δημιουργία παιχνιδιών στενά συνυφασμένων με την καθημερινή ζωή, τα αντικείμενα και τους ανθρώπους που περιβάλλουν τον καθένα από εμάς. Ένα σύνολο από πρωτότυπες εφαρμογές που αναπτύχθηκαν στα πλαίσια του IPerG είχαν σαν σκοπό να φέρουν στην επιφάνεια ένα σύνολο ερευνητικών ζητημάτων που άπτονται της νέας αυτής

Σχήμα 2.1: Οι παίχτες δημιουργούν με τα σώματά τους ένα τρίγωνο γύρω από ένα εικονικό αντικείμενο

κατηγορίας παιχνιδιών. Παράλληλα, πλήθος άλλων ερευνητικών ομάδων έχουν παρουσιάσει αποτελέσματα της έρευνάς τους στον τομέα. Μια συλλογή γνωστών και αντιπροσωπευτικών ΠΠΥΝ παρατίθεται παρακάτω με σκοπό την περαιτέρω εξοικείωση του αναγνώστη με τον χώρο των παιχνιδιών αυτών.

CatchBob![5] Το CatchBob! είναι μια πειραματική πλατφόρμα με τη μορφή παιχνιδιού κινητής συσκευής και με απώτερο σκοπό την πραγματοποίηση ψυχολογικών πειραμάτων και τη μελέτη της ανθρώπινης συμπεριφοράς. Ο σχεδιασμός του αναπτύσσει την ανάγκη για συνεργατικότητα των συμμετεχόντων για την επίτευξη ενός κοινού σκοπού. Σχηματίζοντας ομάδες των 3 ατόμων οι παίχτες αναζητούν στο φυσικό χώρο εικονικά αντικείμενα γύρω από τα οποία σχηματίζουν ένα τρίγωνο (Σχήμα 2.1).

Οι παίκτες έχουν συνεχώς μαζί τους έναν μικρό υπολογιστή με δυνατότητες αναγνώρισης τοποθεσίας από τον οποίο λαμβάνουν και την ανάλογη πληροφορία σχετικά με τη θέση τους και τη θέση των συμπαικτών τους (Σχήμα 2.2).

Με τη χρήση αισθητήρων εγγύτητας παρέχεται στους συμμετέχοντες η πληροφορία για το πόσο κοντά βρίσκονται στο εικονικό αντικείμενο, χωρίς οι ίδιοι να γνωρίζουν την ακριβή θέση του. Επιπλέον, χρησιμοποιώντας τον υπολογιστή του, ο κάθε παίκτης μπορεί να επικοινωνήσει με τους συμπαίκτες του προτείνοντάς τους να κινηθούν σε ορισμένη κατεύθυνση ή να ανταλλάξει άλλα μηνύματα σχετικά με το παιχνίδι.

Uncle Roy all around you[6] Πρόκειται για ένα παιχνίδι που επιτρέπει τόσο τη φυσική συμμετοχή όσο και τη συμμετοχή μέσω διαδικτύου συνδυάζοντας τον φυσικό με τον εικονικό κόσμο. Οι παίκτες που βρίσκονται στους δρόμους ακολουθούν στοιχεία και ψάχνουν μέσα σε μια πόλη τον "Θείο Ρόυ" ενώ οι παίκτες

Σχήμα 2.2: Παίχτης του CatchBob! χρησιμοποιεί ένα iPAQ, TabletPC για να εντοπίσει του συμπαίκτες του

που συμμετέχουν μέσω διαδικτύου παρακολουθούν την πρόοδο των υπολοίπων σε ένα παράλληλο εικονικό περιβάλλον και προσπαθούν να τους βοηθήσουν. Και σ' αυτό το παιχνίδι οι παίκτες έχουν μαζί τους μικρούς φορητούς υπολογιστές όπου εμφανίζονται χάρτες και χρησιμοποιούνται στην ανταλλαγή μηνυμάτων. Χρησιμοποιούνται, επίσης, web κάμερες, ηχητικά και γραπτά μηνύματα για την επίτευξη του στόχου του παιχνιδιού.

Netattack[7] Το παιχνίδι διαδραματίζεται σε δύο χώρους παράλληλα. Κάποιοι παίκτες βρίσκονται μπροστά σ'ενα έναν ηλεκτρονικό υπολογιστή (Σχήμα 2.1) και υποστηρίζουν όσους βρίσκονται στο φυσικό περιβάλλον. Όσοι βρίσκονται

Σχήμα 2.3: Η ομάδα υποστήριξης του NetAttack

έξω προσπαθούν να βρουν πληροφορίες και κρυμμένα αντικείμενα σχετικά με το παιχνίδι φορώντας οθόνες προσαρμοσμένες στο κεφάλι τους (Head-mounted Displays) οι οποίες προσφέρουν την εμπειρία της επαυξημένης πραγματικότητας (augmented reality) (Σχήμα 2.4). Παράλληλα φέρουν συσκευές GPS, βιντεοκάμερες

Σχήμα 2.4: Ο παίκτης σε περιβάλλον επαυξημένης πραγματικότητας

και άλλον εξοπλισμό που καθιστά δυνατό τον εντοπισμό τους στο χώρο. Οι δύο ομάδες συνεργάζονται με σκοπό να καταστρέψουν τη βάση δεδομένων μιας εικονικής εταιρίας.

Human Pacman[8] Αποτελεί μια άκρως εμπλουτισμένη εκδοχή του γνωστού PacMan και πρόκειται για ένα διαδραστικό, διάχυτο και κινητό σύστημα ψυχαγωγίας βασισμένο στις έννοιες της τοποθεσίας και της οπτικής των παικτών. Ειδικές συσκευές επιτρέπουν στους παίκτες να βιώσουν μιαν επαυξημένη πραγματικότητα Σχήμα 2.5 κινούμενοι στο φυσικό περιβάλλον.

Καλούνται να ακολουθήσουν προκαθορισμένες διαδρομές, να αποφύγουν ή να κυνηγήσουν συμπαίκτες τους Σχήμα 2.6 και να αλληλεπιδράσουν με φυσικά αντικείμενα. Οι ενέργειες τους επηρεάζουν άμεσα τους ήρωες που εκείνοι αντιπροσωπεύουν σε έναν αμιγώς εικονικό κόσμο ο οποίος παρουσιάζεται στην οθόνη ενός υπολογιστή.

Σχήμα 2.5: Το φυσικό περιβάλλον εμπλουτισμένο με στοιχεία του παιχνιδιού Human Pacman

Σχήμα 2.6: Παίχτες αλληλεπιδρούν πλησιάζοντας ο ένας τον άλλον

2.1.1 Αρχές

Κατά το σχεδιασμό και την ανάπτυξη ΠΠΥΝ με χρήση Ασύρματων Δικτύων Αισθητήρων, ένα σύνολο ερευνητικών προκλήσεων καλείται να απαντηθεί. Τα βασικά θέματα που προκύπτουν μπορούν να προσδιοριστούν ορίζοντας παραμέτρους που έχουν να κάνουν με την αλληλεπίδραση των παικτών, την ύπαρξη ή όχι κεντρικής υποδομής και την χρονική εξέλιξη των ίδιων των παιχνιδιών.

Κατ'αρχήν φαίνεται να εμφανίζονται οι εξής **κατηγορίες** παιχνιδιών:

Αυθόρμητα Τα παιχνίδια ξεκινούν αμέσως ανάμεσα στους παίκτες όπως για παράδειγμα στο διάλειμμα από τη δουλεία ή το σχολείο. Δεν υπάρχει ανάγκη για υποδομή ενώ για την επικοινωνία και την αισθητηριακή είσοδο αρκούν οι συσκευές που οι παίκτες έχουν πάνω τους. Τα αποτελέσματα και τα στατιστικά στοιχεία που αφορούν το παιχνίδι προωθούνται κεντρικά μετά το τέλος του παιχνιδιού. Σε αυτή την κατηγορία δεν υπεισέρχονται κανενός είδους χρονικοί και χωρικοί περιορισμοί στην εξέλιξη των παιχνιδιών.

Βασισμένα σε σενάριο Είναι απαραίτητη ως ένα βαθμό η ύπαρξη υποδομής ώστε να παρέχονται λειτουργίες όπως ο προσδιορισμός θέσης αλλά και να καθίσταται δυνατή η υποστήριξη ενός σεναρίου και συμβάντων σχετιζόμενα με το χρόνο εξέλιξης του. Μια κεντρική οντότητα μπορεί να παίζει το ρόλο του συντονιστή όπως απαιτείται από το εκάστοτε σενάριο. Παράδειγμα ενός τέτοιου παιχνιδιού θα μπορούσε να είναι μια διαδραστική εγκατάσταση (interactive installation) ψυχαγωγικού - εκπαιδευτικού χαρακτήρα στο χώρο ενός μουσείου. Οι παίκτες, δοθείσης μιας συγκεκριμένης πλοκής καλούνται να κινηθούν στους χώρους του μουσείου συλλέγοντας επί παραδείγματι στοιχεία για να προχωρήσουν στην εξέλιξη της ιστορίας κ.τ.λ. Η κεντρική υποδομή εποπτεύοντας τις κινήσεις των παικτών στο χώρο και τις χρονικές στιγμές που αυτές συμβαίνουν καθορίζει ως ένα βαθμό το αποτέλεσμα του παιχνιδιού.

Βασισμένα σε μια κοινότητα Πρόκειται για κλασσικά παιχνίδια –όπως το Κρυφτό ή το Κυνηγητό– εμπλουτισμένα με ιδιότητες που μπορούν να προσφέρουν πληροφοριακά συστήματα και οι συσκευές αισθητήρων: τρόπος διεξαγωγής, επιβολή των κανόνων, συλλογή και διατήρηση στατιστικών στοιχείων και άλλα. Μπορούν να βασίζονται τόσο σε κεντρική υποδομή, όσο και να μπορούν να παιχτούν χωρίς αυτή. Σε κάθε περίπτωση η ιδέα της κοινότητας έχει κυρίαρχο ρόλο. Υπάρχει μια κοινότητα που παίζει παιχνίδια διαφορετικών ειδών η διάρκεια των οποίων μπορεί να είναι εξαιρετικά μικρή μέχρι πολύ μεγάλη ενώ η αλληλεπίδραση μεταξύ των μελών αυτής της κοινότητας είναι έντονη.

Στη συνέχεια εντοπίζονται οι προκλήσεις που παρουσιάζονται στην ανάπτυξη λογισμικού παιχνιδιών βασισμένων σε ΑΔΕ.

Ταυτόχρονη συμμετοχή πολλών χρηστών Στα ΠΠΥΝ με χρήση Ασύρματων Δικτύων Αισθητήρων μπορεί δυνητικά να συμμετέχει μεγάλος αριθμός παιχτών. Επιπλέον, οι παίχτες θα βρίσκονται σε μικρή απόσταση ο ένας από τον άλλον τόσο σε εσωτερικούς όσο και σε εξωτερικούς χώρους ενώ θα καλούνται να αλληλεπιδράσουν είτε με τους συμπαίκτες τους είτε με την υποδομή. Ανάλογα με το είδος του παιχνιδιού οι παίχτες είτε ανταγωνίζονται ο ένας τον άλλον είτε συνεργάζονται, για την επίτευξη ενός κοινού στόχου, σε πραγματικό χρόνο. Όσον αφορά την υλοποίηση, γίνεται η υπόθεση πως υπάρχει ένας αξιόπιστος μηχανισμός ανεύρεσης γειτονιάς και εντοπισμού θέσης καθώς επίσης μηχανισμούς context-aware και location-aware που παρέχουν πληροφορία στους παίκτες αλλά και στο ίδιο το σύστημα. Τέτοιοι μηχανισμοί είναι απαραίτητοι για τη υποστήριξη αυξημένου πλήθους παικτών και διαφορετικών περιοχών εξέλιξης των παιχνιδιών.

Πολλαπλοί τύποι εισόδου Μια πληθώρα εισόδων που σχετίζονται με την παρουσία, την κίνηση και άλλες πληροφορίες αισθητηριακού περιεχομένου (sensory input) εμπλουτίζουν σε μεγάλο βαθμό την συνολική εμπειρία του παιχνιδιού. Αυτού του είδους οι πληροφορίες παρέχονται είτε απευθείας από τις κινητές συσκευές που οι παίκτες έχουν μαζί τους είτε από την κεντρική υποδομή. Για παράδειγμα, επισκέπτες ενός μουσείου φέρουν συσκευές που επιτρέπουν τον προσδιορισμό της θέσης τους (απόλυτης ή σχετικής με κάποιο προκαθορισμένο σημείο), αναγνωρίζουν τις κινήσεις τους (σε επίπεδο κίνησης του σώματός τους ή χειρονομιών) και προσδιορίζουν άλλα χαρακτηριστικά του περιβάλλοντος (όπως το εάν βρίσκονται σε φωτεινό-σκοτεινό ή ζεστό-κρύο χώρο). Για το λόγο αυτό η αρχιτεκτονική ενός συστήματος που παρέχει όλες αυτές τις υπηρεσίες απαιτείται να είναι επεκτάσιμη και να προσφέρει αξιόπιστους μηχανισμούς εντοπισμού κίνησης και αναγνώρισης χειρονομιών. Στην περίπτωση που γίνεται και η χρήση καμερών, απαιτείται η ανάπτυξη αντίστοιχων μηχανισμών επεξεργασίας video.

Μη συμβατικές μέθοδοι διεπαφών και χρήση actuators-haptics Οι παίκτες θα πρέπει να είναι σε θέση να έχουν πρόσβαση σε δεδομένα που αφορούν τους ίδιους αλλά και την ομάδα τους με τρόπο που συνάδει με σενάρια των παιχνιδιών και τη διαδραστική εμπειρία που αυτά προσφέρουν. Επιπλέον, οι διεπαφές που προσφέρει το σύστημα θα πρέπει να αντικατοπτρίζουν τα στοιχεία της τοποθεσίας και του περιβάλλοντος που επικρατούν. Η χρήση actuators για τον χειρισμό του φωτισμού και των γενικότερων συνθηκών που επικρατούν στο χώρο έχουν σαν αποτέλεσμα την εντονότερη εμπειρία κατά τη διάρκεια ενός παιχνιδιού ή μιας διαδραστικής εγκατάστασης.

Κατανεμημένη λειτουργία Η χρήση κόμβων αισθητήρων και οι δυνατότητες ad-hoc απαιτούν το εκάστοτε λογισμικού που εκτελείται στης κινητές συσκευές

να βασίζεται σε ελαφρούς μηχανισμούς. Περίπλοκες υλοποιήσεις της λογικής τους συστήματος χρειάζεται να εκτελούνται στην κεντρική υποδομή. Επιπλέον λειτουργίες ενδέχεται να βασίζονται σε ενέργειες συντονισμού που εκτελούνται σε πραγματικό χρόνο, απαιτούν πλήρη γνώση για την κατάσταση και τη θέση των παικτών, ενώ οι ίδιες οι λειτουργίες μπορεί να εκτελούντα σε αποσυνδεδεμένα μέρη του δικτύου. Δεδομένων των παραπάνω, το σύστημα οφείλει να προσδιορίζει καταστάσεις στις οποίες η επικοινωνία δεν είναι δυνατή και να ενεργοποιεί μηχανισμού ανεκτικούς στις καθυστερήσεις (delay-tolerant mechanisms). Με τον τρόπο αυτό διασφαλίζεται η σωστή λειτουργία καθώς και η αξιόπιστη επικοινωνία μεταξύ παικτών και υποδομής. Ως εκ τούτου είναι ζωτικής σημασίας η αρχιτεκτονική ενός τέτοιου συστήματος να είναι κατανεμημένη, να βασίζεται σε ανομοιογενή αυτόνομα στοιχεία (heterogeneity, modularity) και να παρουσιάζει προσαρμοστικότητα σε πραγματικές συνθήκες.

Ανάγκη για συγχρονισμό και συντονισμό των παικτών Η αλληλεπίδραση μεταξύ των χρηστών και μεταξύ των χρηστών και του συστήματος πρέπει να γίνεται με συγχρονισμένο τρόπο καθώς αυτοί συνεργάζονται για την επίτευξη ενός κοινού στόχου. Ο συγχρονισμός θα πρέπει να γίνεται σχετικά με την κατάσταση των παικτών και της κατάσταση του συστήματος. Οι κινήσεις και ενέργειες των χρηστών μέσα στο χώρο θα πρέπει επίσης να συντονίζονται όταν αυτό χρειάζεται. Μηχανισμοί συμφωνίας, αμοιβαίου αποκλεισμού, εκλογής αρχηγού μπορούν να χρησιμοποιηθούν για την εξασφάλιση της ορθής λειτουργίας του συστήματος.

Ανάγκη για αξιοπιστία Από τη στιγμή που τα παιχνίδια αυτού του τύπου πρόκειται να παίζονται με τη χρήση ακόμα και φθηνών συσκευών με σχεδιασμό που μπορεί να δημιουργήσει προβλήματα κατά την εξέλιξη του παιχνιδιού όπως η εσφαλμένη επανεκκίνηση της συσκευής κ.τ.λ. οι παίκτες χρειάζεται να είναι βέβαιοι ότι τα χαρακτηριστικά αυτά δεν θα αποτελούν εμπόδιο στην ομαλή εξέλιξη των παιχνιδιών. Η καταγραφή των ενεργειών και της κατάστασης των παικτών ίσως κρίνεται απαραίτητη σε κάποια παιχνίδια προκειμένου να διασφαλιστεί η εύρυθμη λειτουργία καθώς επίσης και η παροχή μηχανισμών ανοχής σε σφάλματα και ανάκαμψης από αυτά.

Ο κόσμος σαν ταμπλό παιχνιδιού Τα παιχνίδια θα πρέπει να είναι υλοποιημένα με τέτοιο τρόπο ώστε να παρέχουν την έννοια του κατανεμημένου συστήματος και τους παράγοντες τοπικής ενημερότητας (context-awareness) στους συμμετέχοντες. Αυτό αφορά βεβαίως την κεντρική ιδέα και την πρόκληση πίσω από τα ΠΠΥΝ γενικότερα. Ερωτήματα που γεννιούνται: πώς θα χειρίζεται ο μηχανισμός του παιχνιδιού την κεντρική υποδομή που συναντάται, τι θα συμβαίνει

τις περιόδους όπου ο παίκτης βρίσκεται αποκομμένος από την υποδομή ή κατα τις μετακινήσεις του;

Πολλαπλές πλατφόρμες Πέρα από τη χρήση πολλαπλών, διαφορετικών μορφών εισόδων, αυτού του είδους οι εφαρμογές μπορούν εναλλακτικά να βασίζονται σε ένα πλήθος από διαφορετικές πλατφόρμες ανάλογα με το περιβάλλον που εκτελούνται. Για παράδειγμα η έναρξη ενός παιχνιδιού στο χώρο εργασίας τις καθημερινές και στα σαββατοκύριακα στο σπίτι. Επίσης, θα πρέπει να προσαρμόζεται στη χρονική στιγμή και το χώρο ή το επίπεδο πολυπλοκότητας του περιβάλλοντος. Για παράδειγμα, μια εφαρμογή έχει άλλες εκφάνσεις όταν παρουσιάζεται σε μια κινητή συσκευή, άλλες όταν παρακολουθείται σε επίπεδο στατιστικών κ.τ.λ. Η ποικιλία των εισόδων, των διεπαφών του χρήστη και οι δυνατότητες που παρέχονται καθιστούν ιδιαίτερα δύσκολη την παροχή μια καθολικής εμπειρίας παιχνιδιού.

2.2 Ασύρματα Δίκτυα Αισθητήρων

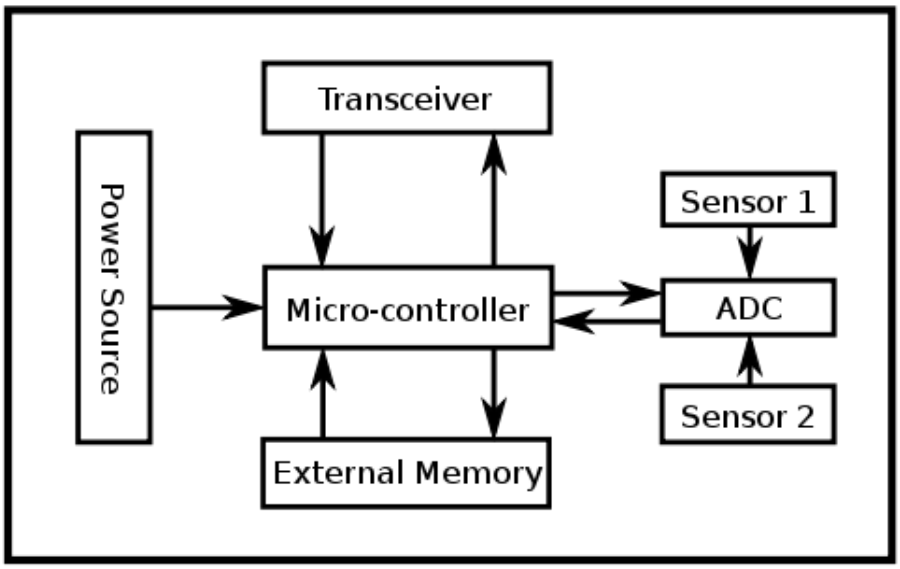
Ένα ασύρματο δίκτυο αισθητήρων αποτελείται –τυπικά– από ένα πλήθος αισθητήρων οι οποίοι βρίσκονται κατανεμημένοι στο χώρο και από κοινού καταγράφουν τις μεταβολές των συνθηκών και φυσικών ποσοτήτων που επικρατούν στην περιοχή που καλύπτουν. Οι ποσότητες αυτές μπορεί να είναι θερμοκρασία, υγρασία, φωτεινότητα, πίεση, κίνηση, επιτάχυνση και άλλες. Με αρχική χρήση στον στρατιωτικό τομέα, τα ασύρματα δίκτυα αισθητήρων βρήκαν σύντομα εφαρμογές σε ένα σύνολο άλλων διαφορετικών τομέων. Σήμερα, συναντά κανείς τέτοια δίκτυα ευρέως σε βιομηχανικές εγκαταστάσεις, σε τομείς υγείας, περιβαλλοντικών δραστηριοτήτων, συστήματα οικιακού αυτοματισμού ή και συστήματα διαχείρισης κυκλοφορίας.

2.2.1 Κόμβοι ασύρματων δικτύων αισθητήρων

Ένας Αισθητήρας στα Ασύρματα Δίκτυα Αισθητήρων πρόκειται για μια συσκευή ικανή να εκτελέσει ένα σύνολο υπολογισμών, να συλλέξει πληροφορίες που αφορούν το περιβάλλον στο οποίο βρίσκεται και να επικοινωνήσει με άλλους αισθητήρες συνδεδεμένους στο δίκτυο.

Η τυπική αρχιτεκτονική μιας τέτοιας συσκευής φαίνεται στο Σχήμα 2.7.

Ένας Αισθητήρας απαρτίζεται κυρίως από έναν μικροελεγκτή, έναν πομποδέκτη, μια εξωτερική μνήμη, την παροχή ρεύματος και ένα πλήθος αισθητήρων φυσικών ποσοτήτων. Το κόστος όπως και το μέγεθος των αισθητήρων ποικίλει και εξαρτάται από το μέγεθος της συσκευής, τις δυνατότητες που παρέχει και την υπολογιστική ισχύ με την οποία είναι εξοπλισμένος.



Σχήμα 2.7: Η τυπική αρχιτεκτονική ενός Αισθητήρα περιλαμβάνει την τροφοδοσία, τον πομποδέκτη, την εξωτερική μνήμη και έναν μικροελεγκτή μαζί με ένα πλήθος μονάδων αισθητήρων. Οι αισθητήρες αυτοί μπορεί να είναι θερμοκρασίας, υγρασίας, φωτεινότητας κ.τ.λ.

Παρακάτω παρουσιάζονται κάποιες από τις πιο σημαντικές και ευρέως χρησιμοποιούμενες συσκευές ασύρματων αισθητήρων.

Tmote [11]

- **CPU** 8MHz Texas Instruments MSP430 microcontroller
- **Memory** 10k RAM and 48k Flash
- **I/O** Integrated Humidity, Temperature, and Light sensors
- **Radio**250kbps 2.4GHz IEEE 802.15.4 Chipcon Wireless Transceiver

Mica2 [12]

- **CPU** Atmel ATmega128L
- **Memory** 4K RAM 128K Flash
- **I/O** Large expansion connector
- **Radio** 315, 433 or 868/916Mhz Multi-Channel transceiver with 38 Kbaud

MicaZ [13]

- **CPU** ATMEGA 128
- **Memory** 4K RAM 128K Flash
- **I/O** Large expansion connector
- **Radio** 802.15.4/ZigBee compliant RF transceiver

TelosB [14]

- **CPU** Texas Instruments MSP430 microcontroller
- **Memory** 10k RAM, 48k Flash
- **I/O** Digital I/O,I2C,SPI
- **Radio** 250 kbit/s 2.4 GHz IEEE 802.15.4 Chipcon Wireless Transceiver

MSB-A2 [9]

- **CPU** ARM7-TDMI
- **Memory** 512 KB ROM, 98 KB RAM
- **I/O** Temperature, Humidity,Coulomb counter
- **Radio** CC1100 802.15.4

CSIRO Fleck [10]

- **CPU** Atmega128L, 8MHz
- **Memory** 512K external memory
- **I/O** Sensors Temperature, Light, Screw terminal for 4X digital i/o and 2X analog
- **Radio** Nordic 903

iSense [15]

- **CPU** 32 Bit RISC Controller, 4-32MHz
- **Memory** 128kB RAM, 512kB Flash
- **I/O** Digital I/O,I2C,SPI
- **Radio** 802.15.4,250kbit/s, hardware AES encryption, ToF ranging engine

Sun SPOT [16]

- **CPU** 180 MHz 32 bit ARM920T core
- **Memory** 128kB RAM, 512kB Flash
- **I/O** accelerometer, Temperature, Light, 6 analog inputs, 5 general purpose I/O pins and 4 high current output pins
- **Radio** 2.4 GHz IEEE 802.15.4 radio with integrated antenna

Όπως παρατηρεί κανείς, οι αρχιτεκτονικές στα συστήματα αυτά ποικίλουν. Ξεκινούν από αρχιτεκτονικές μικροελεγκτών των 8 bit (CSIRO Flesh, Mica, TMote) αλλά φτάνουν και τις 32 bit (Sun SPOT, iSense). Αντίστοιχη διακύμανση παρατηρείται και στη μνήμη που οι συσκευές διαθέτουν, το πλήθος των αισθητήρων που διαθέτουν. Ώστότο το πρωτόκολλο IEEE 802.15.4 αποδεικνύεται ιδιαίτερα δημοφιλές καθώς η πλειοψηφία των συσκευών επικοινωνούν μέσω αυτού. Το γεγονός αυτό επιτρέπει την δημιουργία δικτύων στα οποία συμμετέχουν διαφορετικές συσκευές, με διαφορετικό σκοπό και δυνατότητες. Αξίζει να σημειωθεί επίσης πώς τα Sun SPOT και τα iSense παρέχουν την δυνατότητα διασύνδεσης επιπλέον δευτερευουσών πλακετών για επιπλέον χρήσεις (όπως γυροσκόπιο, μαγνητόμετρο, GPS module).

2.2.2 Δίκτυα αισθητήρων και εφαρμογές

Ένα δίκτυο αισθητήρων είναι της περισσότερες φορές ένα ασύρματο αδόμητο δίκτυο. Κάθε κόμβος υποστηρίζει ένα multi-hop αλγόριθμο δρομολόγησης ([19], [20]) και προωθεί πακέτα είτε σε άλλους κόμβους είτε σε κάποιο σταθμό κεντρικής υποδομής (base station).

Οι εφαρμογές σε τέτοιου είδους δίκτυα ποικίλουν περιλαμβάνουν κυρίως κάποιο είδους καταγραφής, παρακολούθησης και ελέγχου μεταβλητών του περιβάλλοντος στο οποίο βρίσκονται. Χαρακτηριστικοί τομείς εφαρμογής είναι ο έλεγχος οικιακού και βιομηχανικού περιβάλλοντος, η παρακολούθηση αντικειμένων, ο εντοπισμός πυρκαγιών, κατολισθήσεων και άλλων περιβαλλοντικών

καταστάσεων. Σε μια τυπική εφαρμογή, οι αισθητήρες διασκορπίζονται σε μια περιοχή και στη συνέχεια εκτελούν ένα σύνολο προκαθορισμένων λειτουργιών.

Παρακολούθηση μιας περιοχής Η παρακολούθηση μιας συγκεκριμένης περιοχής είναι μια συνήθης εφαρμογή στα Ασύρματα Δίκτυα Αισθητήρων. Σε τέτοιες εφαρμογές ένα δίκτυο εγκαθίσταται σε μια περιοχή όπου πιθανώς να λάβει χώρα κάποιο φαινόμενο. Όταν οι αισθητήρες αντιληφθούν το γεγονός μέσω της αλλαγής κάποιας ποσότητας (θερμοκρασία, πίεση, ήχος, ηλεκτρομαγνητικό πεδίο, δόνηση κ.τ.λ.) το γεγονός αναφέρεται στο base station το οποίο με τη σειρά του ενεργεί κατάλληλα (αποστολή μηνυμάτων μέσω Internet, ενεργοποίηση actuator κ.τ.λ.)

Περιβαλλοντική παρακολούθηση Ένα πλήθος Ασύρματων δικτύων αισθητήρων έχουν εγκατασταθεί κατά καιρούς με σκοπούς την περιβαλλοντική παρακολούθηση [17]. Ωστόσο πολλά από αυτά δεν κατάφεραν να αντέξουν στο χρόνο κυρίως λόγω των συνθηκών που επικρατούσαν στα πεδία εφαρμογής τους και της πειραματικής φύσης τους. Τυπικό παράδειγμα τέτοιων εφαρμογών είναι η παρακολούθηση θερμοκηπίων όπου αισθητήρες καταγράφουν τα επίπεδα υγρασίας και θερμοκρασίας και τα διατηρούν σε ιδανικά επίπεδα [21].

Βιομηχανική παρακολούθηση Ασύρματα δίκτυα αισθητήρων χρησιμοποιούνται και στην παρακολούθηση κατάστασης των μηχανημάτων σε βιομηχανικό περιβάλλον καθώς προσφέρουν σημαντική μείωση στο κόστος σε σχέση με τις ενσύρματες εγκαταστάσεις ενώ εισάγουν νέες λειτουργίες [18]. Η βιομηχανίες ύδρευσης / αποχέτευσης αποτελούν επίσης ένα χώρο στον οποίο προσφέρεται η χρήση Ασύρματων Δικτύων Αισθητήρων, εκεί όπου δίκτυα ηλεκτροδότησης και ενσύρματης επικοινωνίας δεν υπάρχουν, τα ΑΔΕ μπορούν να προσφέρουν λύσεις.

Η χρήση ΑΔΕ αυξάνεται επίσης στη γεωργική βιομηχανία. Τα επίπεδα νερού σε δεξαμενές μπορούν να παρακολουθούνται διαρκώς, οι αντλίες και τα συστήματα άρδευσης μπορούν να ελεγχθούν επίσης αυξάνοντας την αξιοποίηση των αποθεμάτων.

2.2.3 Παιχνίδια με τη χρήση αισθητήρων

Παρά την έντονη ερευνητική δραστηριότητα τα τελευταία χρόνια στο χώρο των Ασύρματων Δικτύων Αισθητήρων, η εύρευνα σε σχέση με τη χρήση τους στον τομέα των παιχνιδιών και της ψυχαγωγίας είναι περιορισμένη. Παρότι έχουν παρουσιαστεί πολλές εφαρμογές για Ασύρματα Δίκτυα Αισθητήρων, μόνο λίγες σχετίζονται με multi-player, mobile παιχνίδια όπου οι παίκτες φέρουν συ-

σκευές με δυνατότητες αντίληψης του περιβάλλοντός τους ως μέσω αλληλεπίδρασης με αυτό και τους συμπαίκτες τους [22], [23].

Η χρήση αισθητήρων γενικότερα ωστόσο φαίνεται να κερδίζει συνεχώς έδαφος στη βιομηχανία της ψυχαγωγίας. Τα παραδείγματα είναι πολλά και συνεχώς αυξάνονται.

Η πλατφόρμα Wii [26] έχει ήδη γνωρίσει μεγάλη επιτυχία εισάγοντας καινούργια χαρακτηριστικά στον εν λόγω χώρο. Το ιδιαίτερο χαρακτηριστικό της πλατφόρμας είναι το ασύρματο χειριστήριό της το οποίο μπορεί να χρησιμοποιηθεί ως συσκευή κατάδειξης ενώ αναγνωρίζει τις κινήσεις των παικτών στον τρισδιάστατο χώρο.

Την ίδια φιλοσοφία ακολουθεί και η πλατφόρμα Kinect [27] η οποία κάνει χρήση video καμερών και επεξεργασίας video προκειμένου να επιτρέψει στους χρήστες να αλληλεπιδρούν κάνοντας χειρονομίες, μιλώντας ή μετακινώντας φυσικά αντικείμενα.

Επιπλέον, είναι εμφανής η τάση για παιχνίδια σε κινητές συσκευές. Γεγονός που αποδεικνύεται από την επιτυχία που έχουν γνωρίσει αντίστοιχες συσκευές όπως το PSP , Nintendo DS και τα κινητά τηλέφωνα όπως το iPhone.

Κεφάλαιο 3

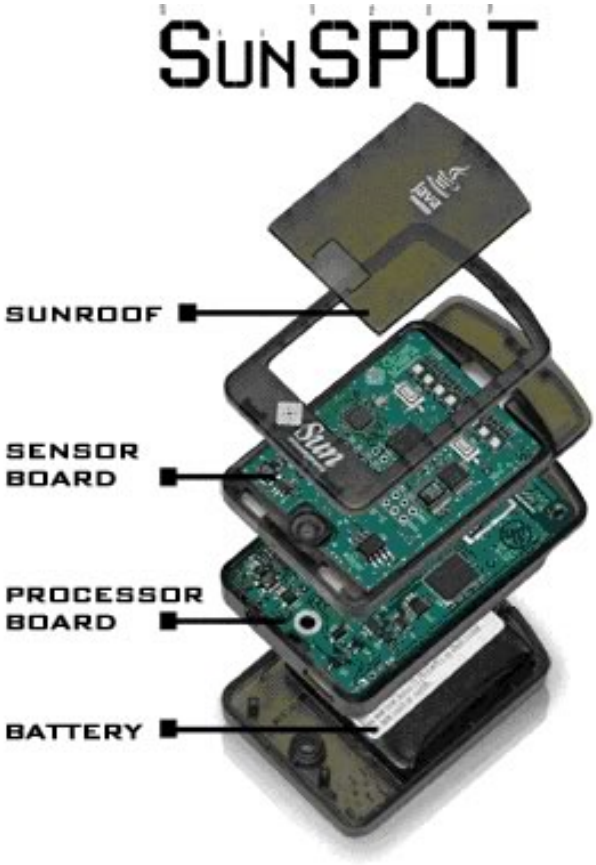
Η πλατφόρμα Fun In Numbers

3.1 SunSPOTs

Σχήμα 3.1: Ένας αισθητήρας Sun SPOT: μια μικρή, ασύρματη πειραματική πλατφόρμα.

Το Sun SPOT (βλ. Σχήμα 3.1) πρόκειται για μια μικρή, ασύρματη πειραματική πλατφόρμα(που εντάσσεται στην κατηγορία των αισθητήρων. Προγραμματίζεται εξολοκλήρου σε Java γεγονός που επιτρέπει την ανάπτυξη εφαρμογών χωρίς να απαιτείται εξειδικευμένη γνώση προγραμματισμού ενσωματωμένων συστημάτων. Η συσκευή περιλαμβάνει ένα πλήθος αισθητήρων καθώς και παρέχει τη δυνατότητα διασύνδεσης εξωτερικών συσκευών και επιπλέον πλακετών.

Η συσκευή είναι αποτελείται από –τουλάχιστον τέσσερα– διαφορετικά επίπεδα (βλ. Σχήμα 3.2) δύο εκ των οποίων και τα πιο σημαντικά: η βασική πλακέτα (eSpot board) και η πλακέτα των αισθητήρων (eDemo board).



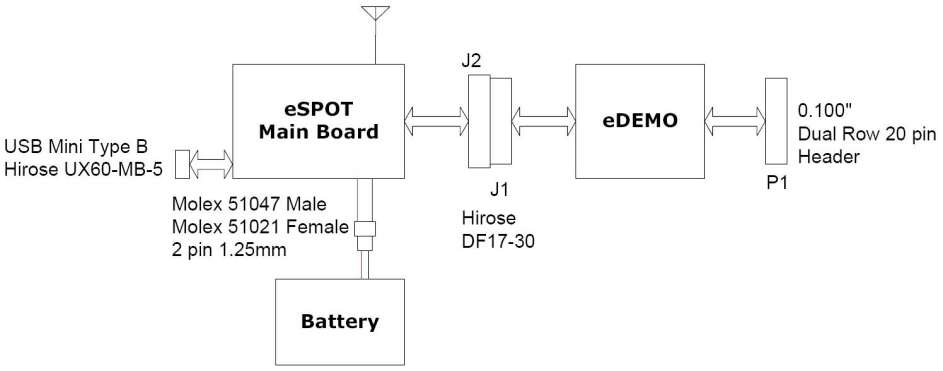
Σχήμα 3.2: Τα διαφορετικά επίπεδα υλικού που συνθέτουν ένα Sun SPOT. Από κάτω προς τα πάνω: μπαταρία, βασικά πλακέτα επεξεργαστή, πλακέτα αισθητήρων, καπάκι.

Το βασικό developement kit των SunSPOT περιλαμβάνει δύο ειδών συσκευές:

- **Basestation** Το Basestation αποτελείται μόνο από την βασική πλακέτα eSpot χωρίς μπαταρία ή επιπλέον πλακέτα. Τροφοδοτείται μέσω της USB σύνδεσης σε υπολογιστή ο οποίος χρησιμοποιείται και σαν host workstastion για τη συσκευή. Η κυριότερη λειτουργία του basestation είναι αυτή του gateway μεταξύ των Sun SPOT και γενικότερα συσκευών συμβατών με το IEEE 802.15.4 και του workstastion.

- **eSPOT** Η μονάδα περιλαμβάνει επιπλέον μια επανατροφοδοτούμενη μπαταρία και τη δευτερεύουσα πλακέτα eDemo board.

Μια σχηματική απεικόνιση των πλακετών και της διασύνδεσής τους φαίνεται στο Σχήμα 3.3.



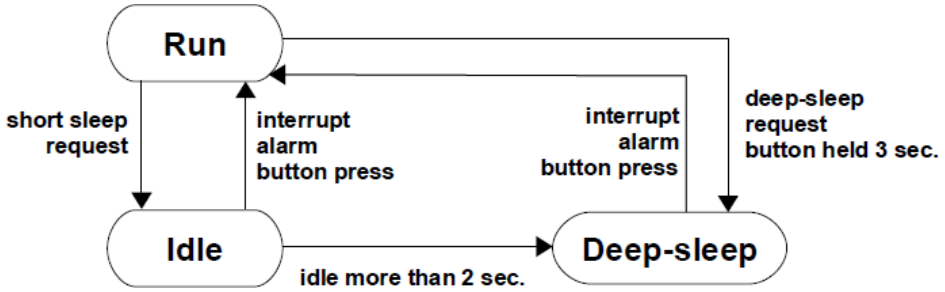
Σχήμα 3.3: Σχηματική αναπαράσταση των μερών ενός Sun SPOT και της μεταξύ τους διασύνδεσης.

3.1.1 Η πλακέτα eSpot Main Board

Η πλακέτα eSpot main board αποτελείται από:

- Κεντρικό επεξεργαστή
- Μνήμη
- Κύκλωμα διαχείρισης τροφοδοσίας
- πομποδέκτη 802.15.4 με ενσωματωμένη κεραία
- σύνδεση μπαταρίας
- σύνδεση δευτερεύουσας πλακέτας.

Ο **κεντρικός επεξεργαστής** είναι ένας Atmel ARM920T ARM Thumb processor ενσωματωμένος σ'ένα System On Chip (SOC) κύκλωμα, το AT91RM9200. Η μέγιστη συχνότητα λειτουργίας του φτάνει τα 180 Mhz ενώ σε κανονικές συνθήκες καταναλώνει 44mW. Διαθέτει 16 KB κρυφή μνήμη εντολών και 16 KB κρυφή μνήμη δεδομένων. Η **μνήμη** των Sun SPOT είναι μία μονή Spansion S71PL032J40 και αποτελείται από 4MB NOR Fash memory και 512KB pSRAM. Ο χρόνος πρόσβασης στη pSRAM είναι 70nsec και στην flash 65nsec. Τα περιεχόμενα της μνήμης διατηρούνται εφόσον υπάρχει τροφοδοσία. Η συσκευή **τροφοδοτείται** είτε από την μπαταρία είτε από μια εξωτερική πηγή μέσω της σύνδεσης USB. Το κύκλωμα ελέγχου τροφοδοσίας είναι υπεύθυνο για την φόρτιση της μπαταρίας, ρυθμίζει την τροφοδοσία των δευτερευουσών και της κεντρικής πλακέτας, παρέχει ρεύμα κατά την κατάσταση deep-sleep, διατηρεί των μετρητή του εσωτερικού ρολογιού 64 bit ενώ ελέγχει και παρακολουθεί τους διακόπτες και τα αντίστοιχα LED.



Σχήμα 3.4: Οι καταστάσεις λειτουργίας ενός Sun SPOT.

Τα Sun SPOT έχουν ένα firmware διατήρησης της ενέργειας που διακρίνει 3 καταστάσεις:

- **Run** Η βασική λειτουργία με τον επεξεργαστή και τον πομποδέκτη σε λειτουργία. Οι απαιτήσεις ισχύος σε αυτή την κατάσταση είναι μεταξύ 70mA και 120mA.
- **Idle** Το ρολόι του ARM9 καθώς και ο πομποδέκτης είναι κλειστά. Σε αυτή την κατάσταση η κατανάλωση είναι περίπου 24mA.
- **Deep-sleep** Όλοι οι ρυθμιστές είναι απενεργοποιημένοι εκτός από τον low-dropout regulator που περιέχεται στο κύκλωμα ελέγχου τροφοδοσίας, τον έλεγχο τροφοδοσίας Atmega και την pSRAM. Η κατανάλωση είναι 32μA. Ο χρόνος εκκίνησης από αυτή την κατάσταση κυμαίνεται από 2ms σε 10ms.

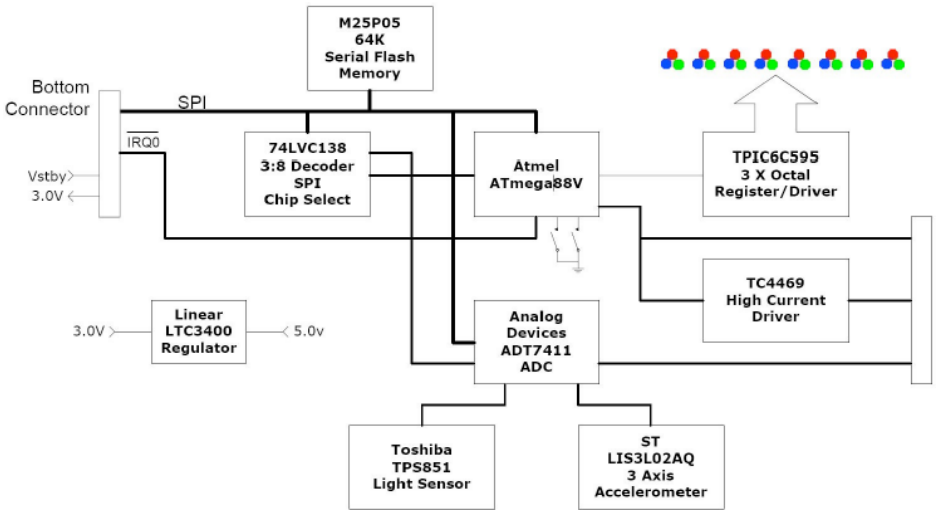
Το διάγραμμα καταστάσεων του Sun SPOT βρίσκεται στο Σχήμα 3.4.

Η **ασύρματη επικοινωνία** του Sun SPOT είναι δυνατή μέσω του TI CC2420 [31]. Το CC2420 είναι συμβατό με το πρωτόκολλο IEEE 802.15.4 και λειτουργεί στις συχνότητες από 2.4 Ghz μέχρι 24.835 GHz. Το ολοκληρωμένο περιέχει έναν πομποδέκτη 2.4Ghz RF με Digital Direct Sequence Spread Spectrum και υποστήριξη MAC. Επίπλεον λειτουργίες που υποστηρίζονται είναι διαφορετικές TX και RX ουρές FIFO των 128 bit, κωδικοποίηση AES, Received Signal Strength Indication (RSSI) με ευαισθησία 100dB και εκπεμπόμενη ισχύ μεταξύ -24dBm και 0dBm ενώ το effective bit rate είναι 250kpbs.

3.1.2 Η δευτερεύουσα πλακέτα eDemo Board

Η πλακέτα των αισθητήρων είναι εξοπλισμένη με τα εξής:

- Atmega88 επεξεργαστή
- μνήμη flash



Σχήμα 3.5: Οι καταστάσεις λειτουργίας ενός Sun SPOT.

- επιταχυνσιόμετρο τριών αξόνων με δύο ρυθμίσεις επιλογής εύρους (2G ή 6G)
- αισθητήρα θερμότητας
- αισθητήρα φωτός
- 8 RGB LEDs
- 6 αναλογικές εισόδους
- 2 διακόπτες
- 5 pins γενικής χρήσης
- 4 pins εξόδου υψηλού δυναμικού

Η σχηματική αναπαράσταση της πλακέτας φαίνεται στο Σχήμα 3.5.

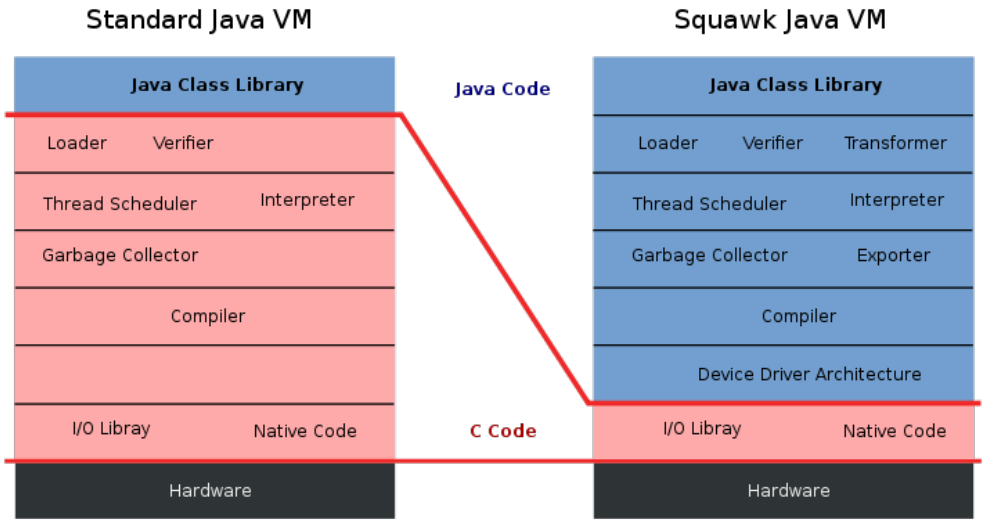
Ο μικροελεγκτής Atmega88 επικοινωνεί με την βασική πλακέτα μέσω ενός διαύλου SPI σαν slave συσκευή. Ο Atmega88 ελέγχει τα LED και τα pins εισόδου-εξόδου, στέλνει αιτήματα διακοπής στον ARM9, ελέγχει τα ψηφία ελέγχου του επιταχυσνιομέτρου καθώς και την αναλογική είσοδο.

3.1.3 Λογισμικό

Squawk VM

Τα Sun SPOT χρησιμοποιούν μια έκδοση της Java με τις δυνατότητες της Java Microedition και λέγεται Squawk [33],[34] . Το Squawk υποστηρίζει τα πακέτα

CLDC 1.1 και MIDP 1.0 και παρέχει τα βασικά ενός Λειτουργικού Συστήματος. Η Virtual Machine εκτελείται απευθείας από την μνήμη Flash. Ένα διάγραμμα σύγκρισης της Squawk με την Standard VM εμφανίζεται στο Σχήμα 3.6.



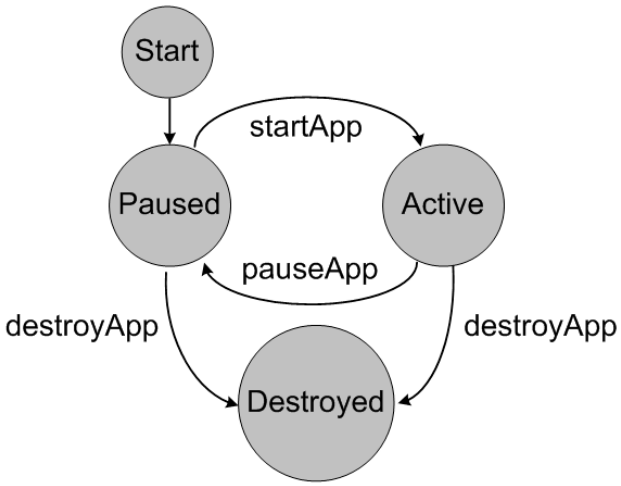
Σχήμα 3.6: Η Squawk VM σε σύγκριση με την Standar Java VM. Ο σχεδιασμός της πρώτης είναι ιδανικός για εκτέλεση σε ενσωματωμένα συστήματα και μικρές συσκευές. Σε αντίθεση με τις περισσότερες Virtual Machines για Java ο πυρήνας των οποίων είναι γραμμένος σε C/C++, ο πυρήνας της Squawk είναι γραμμένος σε Java.

Οι εφαρμογές που χτίζονται για το Sqauwk και γενικότερα ακολουθούν τις προδιαγραφές του MIDP ονομάζονται MIDlets. Για την εκτέλεση των MIDlet απαραίτητη είναι και η εκτέλεση ενός συνόλου εφαρμογών που τρέχουν σε χαμηλότερο επίπεδο καθώς απουσιάζει πλήρως το λειτουργικό σύστημα.

Τα MIDlet ελέγχονται από έναν application manager ο οποίος μπορεί να αναστείλει και να συνεχίσει την εκτέλεση τους. Οι καταστάσεις στις οποίες μπορεί να βρεθεί ένα MIDlet είναι τρείς:

- **paused** - Το στιγμιότυπο του MIDlet έχει δημιουργηθεί αλλά δεν εκτελείται.
- **active** - Το MIDlet εκτελείται.
- **destroyed** - Το MIDlet έχει τερματιστεί το αντικείμενο μπορεί να συλλεχθεί από τον garbage collector.

Ο κύκλος ζωής ενός MIDlet φαίνεται στο Σχήμα 3.7. Η μετάβαση σε κάθε μία κατάσταση γίνεται με την κλήση αντίστοιχων μεθόδων.



Σχήμα 3.7: Το διάγραμμα καταστάσεων του κύκλου ζωής ενός MIDlet.

Για να θεώρηθεί μια κλάση MIDlet πρέπει να κληρονομεί την κλάση `javax.microedition.midlet.MIDlet` και να υλοποιεί τις αντίστοιχες μεθόδους μετάβασης καταστάσεων όπως φαίνεται παρακάτω.

```
public class Application extends MIDlet {
   public Application() { }

   // Called when the MIDlet is created or re-started
   public void startApp() { }

   // Called to pause the MIDlet
   public void pauseApp() { }

   // Called to terminate the MIDlet
   public void destroyApp(boolean unconditional) { }
}
```

Sun SPOT SDK Για τον προγραμματισμό των Sun SPOT χρησιμοποιείται το Sun SPOT SDK που περιλαμβάνει ένα σύνολο βιβλιοθηκών για το χειρισμό των συστημάτων της συσκευής. Η έκδοση 5.0 του SDK περιέχει τρία ξεχωριστά πακέτα βιβλιοθηκών:

1. SPOT and Sensorboard libraries - Μεταξύ άλλων δίνει πρόσβαση στη μνήμη, στις περιφερειακές συσκευές του main board, στο wireless radio, στα υποστηριζόμενα routing πρωτόκολλα, επιτρέπει τη χρήση του sensorboard καθώς και παρέχει πακέτα για την εκτέλεση δοκιμαστικών εφαρμογών.

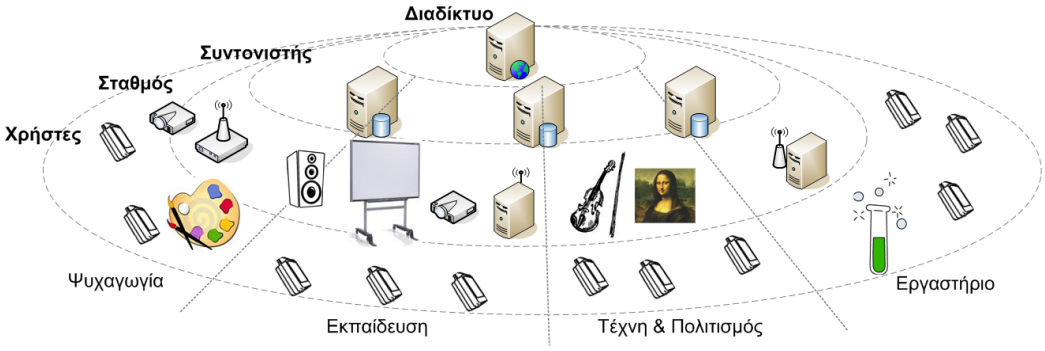
2. SPOT Generic Connection Framework - Δίνει προσβαση σε όλους τους συσκευές εισόδου/εξόδου του Sun SPOT.

3. Squawk Java ME library - Περιέχει όλες της βιβλιοθήκες για τον έλεγχο της Squawk VM.

3.2 Αρχιτεκτονική

Η πλατφόρμα Fun In Numbers έχει σχεδιαστεί και υλοποιηθεί στοχεύοντας σε εφαρμογές που συναντούν ένα σύνολο προδιαγραφών και χαρακτηριστικών. Οι εφαρμογές ή τα παιχνίδια υποστηρίζουν σχετικά μεγάλο πλήθος συμμετεχόντων (από 20 παίκτες και πάνω). Η ίδια η φιλοσοφία της πλατφόρμας (Fun in Numbers – Διασκέδαση εν τη ενώσει) υποδηλώνει πώς τα παιχνίδια είναι πιο διασκεδαστικά όταν παίζονται από πολλούς. Κάθε παίκτης έχει μαζί του μια κινητή συσκευή που διαθέτει κάποιου είδους αισθητήρων. Αυτή μπορεί να είναι ένα κινητό τηλέφωνο, ένα smartphone ή –στην περίπτωση που εξετάζεται– ένας κόμβος ασύρματου δικτύου αισθητήρων. Οι χρήστες των εφαρμογών έχουν τη δυνατότητα να συμμετέχουν σε διαφορετικά παιχνίδια και είδη παιχνιδιών με διαφορετικό περιεχόμενο. Οι παίκτες μεταπηδούν από εκπαιδευτικά παιχνίδια σε διαδραστικές εγκαταστάσεις ή σενάρια κρυμμένου θησαυρού χωρίς να συμμετέχουν συνειδητά σε κάποια διαδικασία αλλαγής. Τα παιχνίδια που λαμβάνουν χώρα μπορούν να συμβαίνουν ταυτόχρονα αλλά και σε διαφορετικές χρονικές στιγμές. Υπάρχει, δηλαδή, μια χρονική συνέχεια. Επιπλέον, οι παίκτες μπορούν να αλληλεπιδρούν με όσους βρίσκονται στον ίδιο χώρο αλλά και με άλλους που βρίσκονται σε διαφορετικό φυσικό χώρο. Η όλη διαδικασία των παιχνιδιών και των εφαρμογών έχει τη δυνατότητα υποστήριξης από μια κεντρική υποδομή η οποία παρέχει ένα σύνολο από υπηρεσίες. Τέτοιες μπορούν να σχετίζονται με πρόσβαση σε βάσεις δεδομένων, πραγματοποίηση εντοπισμού στο χώρο (localization) ή αντίληψη του περιβάλλοντος (context awareness) [28]. Η ύπαρξη μιας κεντρικής υποδομής πρέπει να επιτρέπει επίσης των συντονισμό όλων των παιχνιδιών, των αλληλεπιδράσεων και γενικότερα των γεγονότων που λαμβάνουν χώρα σε κάποιου είδους σενάριο. Ωστόσο, εφαρμογές και παιχνίδια μπορούν να εκτυλίσσονται χωρίς να προϋποθέτουν μια τέτοια υποδομή είτε να συνεχίζουν να τη σωστή λειτουργία τους όντας απομακρυσμένα από αυτή.

Η αρχιτεκτονική της πλατφόρμας βασίζεται σ'αυτές τις αρχές και υλοποιείται από μια ιεραρχία επιπέδων (βλ.Σχήμα 3.8). Το κάθε ένα από αυτά τα επίπεδα έχει έναν διακριτό ρόλο και καλείται να λύσει συγκεκριμένα προβλήματα.



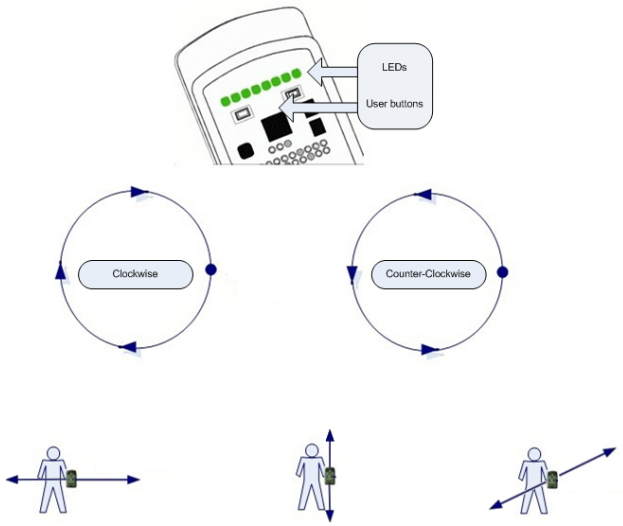
Σχήμα 3.8: Η αρχιτεκτονική της πλατφόρμας FinN. Διακρίνονται τα 4 διακριτά επίπεδα και το πλήθος των διαφορετικών εφαρμογών που μπορούν να υποστηρίξουν – Ψυχαγωγία, Εκπαίδευση, Τέχνη και Πολιτισμός, Εργαστηριακές Εφαρμογές

Επίπεδο χρηστών Το επίπεδο αποτελείται από τις συσκευές (Guardians) που φέρουν οι χρήστης κατά τη διάρκεια των παιχνιδιών FinN. Ο Guardian αποτελεί το λογισμικό που εκτελείται στη συσκευή και χρησιμοποιεί τις δυνατότητες της συσκευής σε επίπεδο επικοινωνίας, διεπαφής χρήστη, αποθήκευσης και άλλων λειτουργιών.

Πρωτόκολλα ανακάλυψης γειτόνων και επικοινωνίας με την κεντρική υποδομή αλλά και με γειτονικούς Guardians παρέχονται σ'αυτό το επίπεδο. Όταν δύο Guardians συναντηθούν οι παίκτες έχουν την δυνατότητα να αλληλεπιδράσουν μεταξύ τους κάνοντας χρήση των διεπαφών που προσφέρουν οι συσκευές. Ο τρόπος της αλληλεπίδρασης αυτής εμπλέκει τη χρήση των αισθητήρων, των επιταχυνσιόμετρων και εν γένει τις φυσικές κινήσεις των παικτών.

Σχετικά με την παρακολούθηση της εξέλιξης του παιχνιδιού, κάθε ενέργεια που λαμβάνει χώρα σχετίζεται με τη δημιουργία ενός Συμβάντος (Event). Το Event περιέχει πληροφορίες όπως το παιχνίδι στο οποίο εντάσσεται, τη χρονική στιγμή που η αλληλεπίδραση ή η ενέργεια συνέβη, την φυσική τοποθεσία και άλλα στοιχεία που μπορούν να προστεθούν ή να αφαιρεθούν κατά περίσταση. Μετά τη δημιουργία τους, τα Event αποστέλλονται στην κεντρική υποδομή ώστε αυτή να είναι ενήμερη. Ωστόσο, υπάρχουν φορές που η αποστολή δεν είναι δυνατή καθώς οι αλληλεπιδράσεις πραγματοποιούνται όταν οι Guardians είναι αποσυνδεδεμένοι από την κεντρική υποδομή. Σε αυτή την περίπτωση, τα δεδομένα αποθηκεύονται στις συσκευές για μεγάλη χρονική περίοδο και μέχρι την αποκατάσταση της σύνδεσης με την κεντρική υποδομή. Το επίπεδο των Guardians με αυτό τον τρόπο επιτρέπει την λειτουργία με ανοχή σε καθυστερήσεις (Delay Tolerance) επηρεάζοντας όσο το δυνατόν λιγότερο την εύρυθμη λειτουργία του συστήματος. Η επικοινωνία των συσκευών των χρηστών στο επίπεδο αυτό

γίνεται μέσω του πρωτοκόλλου IEEE 802.15.4.



Σχήμα 3.9: Η διεπαφή χρήστη βασίζεται στα 8 LED και 2 κουμπιά της συσκευής Sun SPOT ενώ μέσω του επιταχυνσιόμετρου αναγνωρίζονται 6 βασικές κινήσεις: δεξιόστροφος και αριστερόστροφος κύκλος, κίνηση στον άξονα X, κίνηση στον άξονα Y και κίνηση στον άξονα Z.

Διεπαφή Χρήστη Το μέρος της διεπαφής χρήστη είναι υπεύθυνο για την αλληλεπίδραση των παικτών μεταξύ τους αλλά και με το ίδιο το σύστημα. Καθώς οι συσκευές που χρησιμοποιούν οι παίκτες κυρίως είναι τα Sun SPOTS, οι μηχανισμοί διεπαφής των χρηστών εκμεταλλεύονται όσο το δυνατόν περισσότερο τα χαρακτηριστικά τους. Τα βασικά στοιχεία στα οποία βασίζεται το User Interface είναι τα τα 8 LEDs, τα 2 κουμπιά και το επιταχυνσιόμετρο της συσκευής. Κάθε LED μπορεί θεωρητικά να παράξει 256 διαφορετικά χρώματα. Στην πράξη όμως ο αριθμός των χρωμάτων που μπορεί να διακριθεί είναι πολύ μικρότερος. Αυτό δεν λειτουργεί ανασταλτικά καθώς η φύση των παιχνιδιών και των εφαρμογών βασίζονται σε ένα πολύ μικρό ποσοστό στο οπτικό ερέθισμα που παρέχει η ίδια η συσκευή. Μέσω των LEDS οι παίκτες μπορούν να πληροφορούνται στατιστικά που τους αφορούν, την ομάδα που ανήκουν, το πλήθος των συμπαικτών του και άλλα.

Η αναγνώριση των χειρονομιών (Gesture recognition) χρησιμοποιεί το επιταχυνσιόμετρο τριών αξόνων για τον προσδιορισμό των κινήσεων του παίκτη. Ο αλγόριθμος που χρησιμοποιείται δίνει τη δυνατότητα αναγνώρισης των εξής κινήσεων (βλ. Σχήμα 3.9):

- Δεξιόστροφος κύκλος

- Αριστερόστροφος κύκλος
- Κίνηση εμπρός-πίσω
- Κίνηση πάνω-κάτω
- Κίνηση δεξιά-αριστερά

Ανάλογα με τους κανόνες τους παιχνιδιού, η αναγνώριση των κινήσεων ενεργοποιεί άλλες υπηρεσίες και πρωτόκολλα του συστήματος. Όπως για παράδειγμα το Action Protocol στην περίπτωση που πρόκειται για αλληλεπίδραση μεταξύ παικτών ή το Delay Tolerance Service στην περίπτωση δημιουργίας κάποιου Event.

Μόνιμη Αποθήκευση Μέσω της υπηρεσίας του Storage είναι δυνατή η αποθήκευση ενός συνόλου πληροφοριών στη συσκευή του παίκτη. Αυτό επιτυγχάνεται με τη χρήση Recordstore. Ενα Recordstore αποτελεί μια συλλογή εγγραφών που διατηρούνται ανεξάρτητα από τις εκτελέσεις του MIDlet της εφαρμογής. Η συσκευή στην οποία εκτελείται η εφαρμογή καλείται να διατηρήσει τα recordstores αξιόπιστα καθ' όλες τις φάσεις λειτουργίας της συμπεριλαμβανομένων επανεκκινήσεων, αλλαγής μπαταρίας κ.τ.λ.

Το Storage Service παρέχει ένα σύνολο μεθόδων και interfaces επιτρέποντας την αποθήκευση και ανάκτηση κλάσεων που χρησιμοποιούνται σ'ένα παιχνίδι. Οι κλάσεις αυτές αφορούν Ενέργειες, Συμβάντα, τα στοιχεία του παίκτη που έχει τη συσκευή και ένα σύνολο από σημεία ενδιαφέροντος που σχετίζονται με το παιχνίδι. Για κάθε τέτοιο σύνολο αποθηκευμένων κλάσεων, το Storage Service δίνει μεθόδους για την προσθήκη νέων εγγραφών, απαρίθμηση των ήδη υπαρχουσών, διαγραφή και ανάκτηση αποθηκευμένων στοιχείων. Για την αποθήκευση χρησιμοποιείται η μνήμη Flash του Sun SPOT.

Επίπεδο Σταθμών Το επίπεδο αυτό αποτελείται από τα Station (βλ. Σχήμα 3.10) και υλοποιεί την κεντρική υποδομή του συστήματος. Η ύπαρξή της είναι σημαντική αλλά όχι απαραίτητη για όλα τα παιχνίδια και τις εφαρμογές. Παρέχει ένα σύνολο υπηρεσιών όπως localization και context awareness. Μέσω των υπολογιστών τους οποίους εκτελείται το επίπεδο αυτό (για παράδειγμα μικρές υπολογιστικές μηχανές όπως τα Alix boards [24]) είναι δυνατή η συλλογή των δεδομένων που παράγονται από το κατώτερο επίπεδο των Guardians (κυρίως μέσω των Events) με σκοπό τον συντονισμό και τη μόνιμη αποθήκευση δεδομένων. Παράλληλα, και τα ανώτερα επίπεδα μπορούν να έχουν πρόσβαση σε αυτό των χρηστών.

Λόγω της ασύρματης φύσης της υποδομής, κάθε Station σχετίζεται με μια περιοχή γύρω του –την εμβέλειά του– την οποία και ελέγχει. Οι Σταθμοί

Σχήμα 3.10: Μια τυπική εγκατάσταση ενός Σταθμού αποτελούμενη από ένα Alix board με δυνατότητα σύνδεσης σε WiFi και ένα 802.15.4 gateway.

της υποδομής επικοινωνούν με τις συσκευές των χρηστών είτε μέσω αδόμητων δικτύων (ad-hoc) είτε μέσω personal area non-IP δικτύων [29]. Τα Station Λειτουργούν σαν πύλες (gateways) επιτρέποντας την αμφίδρομη επικοινωνία μεταξύ του επιπέδου των Συντονιστών (Engines) και των συσκευών των παικτών.

Περισσότεροι από έναν Σταθμοί μπορούν να συνδεθούν σε ένα Engine με σκοπό να μεγιστοποιηθεί η περιοχή κάλυψης ή να εισαχθούν στο σενάριο του παιχνιδιού περισσότερα σημεία ενδιαφέροντος. Κατά την αρχικοποίηση του παιχνιδιού τα Station επικοινωνούν με τα Engine με σκοπό να ανακτήσουν δεδομένα όπως το πλήθος τον χρηστών που έχουν δηλώσει συμμετοχή στο παιχνίδι, το προφίλ των παικτών αυτών, πληροφορίες για τις συσκευές τους, το σύνολο των σημείων ενδιαφέροντος καθώς και άλλες πληροφορίες απαραίτητες κατά περίσταση. Τα Station είναι επίσης υπεύθυνα για την αρχικοποίηση των συσκευών μεταφέροντας όλα τα απαραίτητα στοιχεία που έχουν ανακτηθεί από τα ανώτερα επίπεδα, στο επίπεδο των Guardians.

Υπάρχει επίσης η δυνατότητα λειτουργίας του Κινητού Σταθμού (mobile Station). Κατά τη διάρκεια ενός παιχνιδιού, ένας τέτοιος Σταθμός χρησιμοποιείται με μερικώς διαφορετικό ρόλο σε σχέση με τους σταθερούς. Η χρήση τους εντοπίζεται κυρίως στον προσδιορισμό των ορίων μιας περιοχής ενδιαφέροντος ή στην πραγματοποίηση εντοπισμού των παικτών στο χώρο. Όσον αφορά την

ανταλλαγή δεδομένων διαμέσου των επιπέδων χρησιμοποιώντας τους σταθμούς αυτού του είδους, αυτή μπορεί είτε να απενεργοποιηθεί πλήρως είτε να ολοκληρωθεί σε μελλοντικό χρόνο. Στην πρώτη περίπτωση, το χαμηλότερο επίπεδο των Guardian πληροφορείται σχετικά με την αδυναμία προώθησης των μηνυμάτων σε ανώτερο επίπεδο ενώ στη δεύτερη, η παράδοση των μηνυμάτων πραγματοποιείται μέσω μεθόδων ανοχής σε καθυστερήσεις (delay tolerance) [30].

Η αμφίδρομη επικοινωνία μεταξύ των Σταθμών και του ανώτερου επιπέδου των Συντονιστών γίνεται μέσω της τεχνολογίας Remote Method Invocation με την οποία είναι δυνατή η ανταλλαγή αντικειμένων και η διατήρηση του αντικειμενοστραφούς μοντέλου.

Επίπεδο Συντονιστών Κάθε στιγμιότυπο παιχνιδιού συνδέεται και με έναν Συντονιστή (Engine). Ένα Engine συντονίζει τα κατώτερα επίπεδα ενώ ανακτά δεδομένα από τα επίπεδα που βρίσκονται πάνω από αυτό και τα αποθηκεύει τοπικά κατά τη διάρκεια του παιχνιδιού. Με σκοπό να μειωθεί ο επικοινωνιακός και υπολογιστικός φόρτος, η ανταλλαγή των δεδομένων μεταξύ των Engine και των ανώτερων επιπέδων πραγματοποιείται ανά τακτά χρονικά διαστήματα. Συνεπώς, η επεξεργασία και η αποθήκευση των Events που φτάνον στο Engine γίνεται –σε πρώτη φάση τουλάχιστον– τοπικά. Επιπρόσθετα, ο Συντονιστής αποτελεί έναν ελεγκτικό μηχανισμό που παρέχει υπηρεσίες άμεσα συσχετιζόμενες με το παιχνίδι και συντελλούν στην εξέλιξη του. Η επικοινωνία μεταξύ του Engine και του Station πραγματοποιείται μέσω ενσύρματου ή ασύρματου IP δικτύου.

Στο επίπεδο των Engine, η μόνιμη αποθήκευση δεδομένων είναι επίσης απαραίτητη. Για την ικανοποίηση των σχεδιαστικών προδιαγραφών, το σύστημα ανά πάσα στιγμή χρειάζεται να γνωρίζει πληροφορίες όπως τα στοιχεία των παικτών, το πλήθος που συμμετέχει σ' ένα παιχνίδι ή ποίοι Σταθμοί αποτελούν την υποδομή του. Παράλληλα, για λόγους διατήρησης ιστορικού και παρακολούθησης στατιστικών τα Συμβάντα, η Ενέργειες και οι κινήσεις των παικτών πρέπει να καταγράφονται. Παράλληλα, λόγω της κατανεμημένη φύσης της πλατφόρμας, τα δεδομένα αυτά χρειάζεται να είναι διαθέσιμα σε διαφορετικά σημεία την ίδια χρονική στιγμή.

Μια αποδοτική λύση στο πρόβλημα αυτό αποτελεί η χρήση της τεχνολογίας Hibernate [39]. Η Hibernate είναι μια βιβλιοθήκη open source για την Java η οποία πραγματοποιεί μια αντιστοίχηση μεταξύ του οντοκεντρικού μοντέλου της γλώσσας σε ένα μοντέλο σχεσιακής βάσης δεδομένων. Αποτελεί μία ολοκληρωμένη λύση στο πρόβλημα της διαχείρισης δεδομένων καθώς εκτελεί όλες τις ενέργειες μεταξύ των εφαρμογών και των σχεσιακών βάσεων δεδομένων απαλλάσσοντας τον προγραμματιστή από επιπρόσθετο κόπο και τον κίνδυνο αστοχιών στην αποτύπωση σχέσεων μεταξύ κλάσεων και σχήματος βάσης.

Ακολουθώντας αυτή την προσέγγιση, όλα τα δεδομένα προς αποθήκευση εισάγονται ή ενημερώνονται από το Engine κάθε παιχνιδιού σε μια κεντρική

βάση δεδομένων. Αντίστοιχα, όταν το Engine απαιτεί πρόσβαση, τα δεδομένα ανακτώνται από τη βάση. Μέσω της Hibernate, κάθε αντικείμενο της πλατφόρμας αντιστοιχίζεται σε έναν πίνακα βάσης δεδομένων ενώ οι μεταβλητές της κλάσεις στα πεδία του πίνακα αυτού. Αντίστοιχα, οι σχέσεις κληρονομικότητας των αντικειμένων αυτών συνθέτουν το σχεσιακό μοντέλο της βάσης. Υπεύθυνοι για την μετατροπή αυτή είναι οι Entity Managers. Σε κάθε αντικείμενο που που προορίζεται για αποθήκευση αντιστοιχεί και ένας Manager ο οποίος περιγράφει τον τρόπο που οι βασικές μέθοδοι αποθήκευσης και ανάκτησης του αντικειμένου υλοποιούνται.

Ιδιαίτερα σημαντικός είναι ο ρόλος του επιπέδου αυτού σε σχέση με την παροχή οπτικοακουστικού feedback στους παίκτες. Ανάλογα με τη φύση του παιχνιδιού, της εφαρμογής ή της διαδραστικής εγκατάστασης που υλοποιείται, τα Events που φτάνουν στον Συντονιστή έχουν ως αποτέλεσμα την πυροδότηση ήχων, αλλαγή εικόνων ή κάποιο άλλο αντίκτυπο στο φυσικό χώρο –όπως αλλαγή του φωτισμού.

Επίπεδο Κόσμου Πρόκειται για το υψηλότερο επίπεδο της ιεραρχίας. Μέσω του επιπέδου αυτού είναι δυνατός ο έλεγχος και η παρακολούθηση πολλαπλών παιχνιδιών ταυτόχρονα, των φυσικών χώρων που συμβαίνουν καθώς και των παικτών που συμμετέχουν σε αυτά. Το επίπεδο επιτρέπει την διαχείριση των υπηρεσιών του μέσω ενός Web Portal ή της οντότητας του HyperEngine, όπου συγκεντρώνονται στατιστικά στοιχεία, προβάλλονται οι θέσεις και η δραστηριότητα των παικτών αλλά και επιτρέπει την εξατομικευμένη παρουσίαση όλων αυτών.

3.3 Παιχνίδια

Στο κεφάλαιο αυτό παρουσιάζονται 6 παιχνίδια και εγκαταστάσεις που έχουν υλοποιηθεί με τη χρήση της πλατφόρμας Fun In Numbers. Σκοπός των εφαρμογών είναι να αναδείξουν τις διαφορετικές δυνατότητες της πλατφόρμας και να παράσχουν όσο το δυνατόν το εύρος των χρήσεων της. Παράλληλα, αποτελούν ένα μέσο για την αξιολόγηση και πιστοποίηση όλων των υπηρεσιών που αυτή παρέχει.

3.3.1 Hot Potato

Οι παίκτες συγκεντρώνονται κρατώντας ο καθένας τη συσκευή του και δημιουργώντας παράλληλα ένα αδόμητο ασύρματο δίκτυο. Για κάθε συσκευή υπάρχει η πιθανότητα να δημιουργηθεί μια "Καυτή Πατάτα". Η πιθανότητα, ωστόσο, εξαρτάται από το πλήθος των παικτών που βρίσκονται συγκεντρωμένοι: όσο μεγαλύτερο είναι αυτό, τόσο μικρότερη η πιθανότητα και αντίστροφα.

Σχήμα 3.11: Ένα στιμιότυπο από παιχνίδι του Hot Potato. Οι παίκτες έχουν στα χέρια τους μια συσκευή Sun SPOT και καμία άλλη υποδομή δεν είναι απαραίτητη.

Μια "Καυτή Πατάτα" είναι στην ουσία ένας μετρητής που μετρά αντίστροφα το χρόνο ενώ αναβοσβήνει τα LED της συσκευής του παίκτη. Κάθε παίκτης έχει τη δυνατότητα να να "πετάξει" την πατάτα του σ'έναν γειτονικό του παίκτη κάνοντας μια φυσική χειρονομία πετάγματος κρατώντας τη συσκευή του. Ο μετρητής τότε συνεχίζει να μειώνεται στα χέρια του επόμενου παίκτη και το παιχνίδι συνεχίζεται. Τελικά ο μετρητής θα φτάσει το μηδέν και η πατάτα θα εκραγεί, ενώ ο τελευταίος που την είχε στα χέρια του θα αποκλειστεί από το παιχνίδι. Ο επόμενος γύρος συνεχίζεται με έναν παίκτη λιγότερο.

Καθώς η πιθανότητα δημιουργίας μιας Καυτής Πατάτας εξαρτάται από το πλήθος των παικτών που βρίσκονται σε επικοινωνία μεταξύ τους, όταν οι συμμετέχοντες δημιουργούν πολλές απομονωμένες "νησίδες" περισσότερες Καυτές Πατάτες δημιουργούνται. Κατά συνέπεια, όταν ένας παίκτης που έχει ήδη μια Καυτή Πατάτα λάβει μια άλλη πατάτα, αυτή με το μεγαλύτερο μετρητή απενεργοποιείται και το παιχνίδι συνεχίζεται με την παλαιότερη. Νικητής είναι ο τελευταίος παίκτης που παραμένει ζωντανός.

Το παιχνίδι αυτό είναι χαρακτηριστικό παράδειγμα "αυθόρμητου" παιχνιδιού που δεν προϋποθέτει την ύπαρξη κεντρική υποδομής για την εξέλιξή του. Οι παίκτες κινούνται ελεύθερα δίχως χωρικούς περιορισμού και αλληλεπιδρούν μεταξύ τους πραγματοποιώντας φυσικές κινήσεις (Σχήμα 3.11). Ωστόσο, η ύπαρξη κεντρικής υποδομής δεν αποκλείεται και μπορεί να έχει βοηθητικό ρόλο.

Σχήμα 3.12: Οι δύο ομάδες ελέγχουν διαφορετικές προβαλλόμενες χρωματικές περιοχές στο παιχνίδι Tug of War.

3.3.2 Casanova

Ένα δεύτερο παράδειγμα αυθόρμητου και αδόμητου παιχνιδιού είναι ο Καζανόβα.

Και πάλι οι παίκτες συγκεντρώνονται κρατώντας ο καθένας τη συσκευή του. Μετά από διαδικασία επιλογής, ένας από όλους τους παίκτες επιλέγεται ως Καζανόβα. Σκοπός του παιχνιδιού είναι ο Καζανόβα να καταφέρει να φύγει εκτός εμβέλειας από όλους τους υπόλοιπους παίκτες ενώ οι υπόλοιποι παίκτες πρέπει να βρίσκονται συνεχώς κοντά σ'αυτόν. Κάθε φορά που εκείνος μένει ακίνητος το ίδιο πρέπει να κάνουν και όλοι οι άλλοι αλλιώς χάνουν τους αρχικούς τους πόντους.

3.3.3 Tug of War

Οι συμμετέχοντες συγκεντρώνονται μπροστά από μια επιφάνεια προβολής και χωρίζονται αυτόματα σε δύο ομάδες ώστε αυτές να είναι ισάριθμες (βλ. Σχήμα 3.12). Σε κάθε μια ομάδα αντιστοιχεί χρωματική μια περιοχή στην επιφάνεια. Σκοπός της κάθε ομάδας είναι να επεκτείνει την περιοχή της όσο το δυνατόν περισσότερο μειώνοντας την περιοχή της αντίπαλης ομάδας. Κάτι τέτοιο επιτυγχάνεται όταν οι παίκτες εκτελούν τις χειρονομίες που υποδεικνύονται με σωστό και γρήγορο τρόπο.

Βασικά χαρακτηριστικά του παιχνιδιού είναι η έντονη παρουσία οπτικών ερεθισμάτων, η ύπαρξη κεντρικής υποδομής, η εκτέλεση χειρονομιών από τους παίκτες το μεγάλο πλήθος τους (βλ. Σχήμα 3.13) .

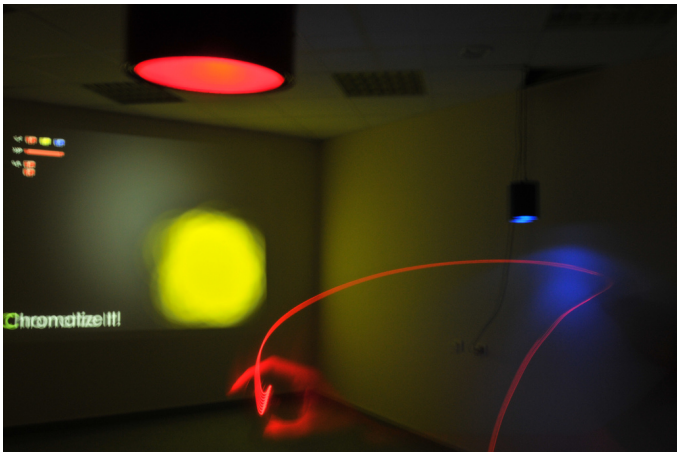
Σχήμα 3.13: Το Tug of War είναι ένα παράδειγμα παιχνιδιού που υποστηρίζει μεγάλο πλήθος παικτών.

Σχήμα 3.14: Οι παίκτες βουτάνε τις συσκευές σε κουβάδες για να "μαζέψουν" χρώμα και στη συνέχεια το "πετούν" προκειμένου να ζωγραφίσουν μια εικόνα.

3.3.4 Chromatize Images

Οι παίκτες βυθίζουν το χέρι τους σ'έναν κουβά με υποτιθέμενη μπογιά κρατώντας τη συσκευή στο χέρι τους. Η συσκευή "γεμίζει" με το αντίστοιχο χρώμα και στη συνέχεια το πετούν στην προβαλλόμενη επιφάνεια εκτελώντας μια φυσική κίνηση πετάγματος (βλ. Σχήμα 3.14). Ο σκοπός τους είναι να χρωματίσουν έναν λευκό καμβά σχηματίζοντας σταδιακά ένα διάσημο -ή και όχι- έργο ζωγραφικής. Περισσότεροι από έναν παίκτες μπορούν να συνδυάσουν χρώματα πετώντας τα ταυτόχρονα στον καμβά. Κάθε φορά που μια εικόνα συμπληρώνεται σωστά, ένα μουσικό θέμα σχετικό με αυτή ακούγεται ανταμείβοντας τους παίκτες.

Επιπλέον χαρακτηριστικά της εγκατάστασης είναι η ύπαρξη "έξυπνων αντικειμένων".



Σχήμα 3.15: Στο Chromatize it οι παίκτες πλησιάζουν φωτεινές πηγές για να πάρουν ένα από τα τρία βασικά χρώματα προκειμένου να και χρωματίζουν ένα λευκό σύννεφο.

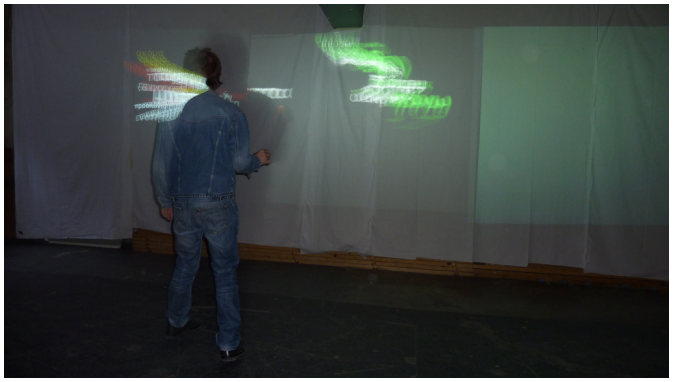
3.3.5 Chromatize It

Σ'αυτή την εφαρμογή-εγκατάσταση οι συμμετέχοντες πειραματίζονται με την ανάμειξη των βασικών χρωμάτων. Οι συμμετέχοντες κινούνται μεταξύ φωτιζόμενων με τα τρία βασικά χρώματα χώρων: κόκκινο, μπλέ και κίτρινο. Όταν εισέρχονται στις περιοχές διαφορετικού χρώματος η συσκευή τους "μαζεύει" το αντίστοιχο χρώμα (Σχήμα 3.15). Τα χρώματα αυτά χρησιμοποιούνται στη συνέχεια για τον χρωματισμό μιας αρχικά λευκής μάζας που αμφιταλαντεύεται σε μια προβαλλόμενη επιφάνεια. Οι παίκτες πλησιάζουν τη μάζα, και "πετούν" το χρώμα σ'αυτή. Σταδιακά η μάζα αλλάζει χρωματικές αποχρώσεις ακολουθώντας τη φυσική ανάμειξη των χρωμάτων (βλ. Σχήμα 3.15). Κάθε φορά που ο ζητούμενος συνδυασμός επιτυγχάνεται, η χρωματική πολυπλοκότητα αυξάνεται.

Χαρακτηριστικό της εγκατάστασης είναι η αντίληψη του περιβάλλοντος των παικτών.

3.3.6 Magnetize Words

Η εγκατάσταση αυτή κάνει χρήση της κίνησης πολλών ανθρώπων σ'ένα κοινό χώρο, όπως για παράδειγμα ενός διαδρόμου προβολών. Με την είσοδό τους στο χώρο αυτό ένα ποίημα σχηματίζει ένα σύννεφο από λέξεις το οποίο ακολουθεί την κίνηση του κάθε παίκτη ξεχωριστά (Σχήμα 3.16). Τα σύννεφα μπλέκονται το ένα με το άλλο καθώς πλησιάζουν έλκοντας ή απωθώντας λέξεις ανάλογα με τις έννοιές τους. Οι λέξεις συνεχώς αναδιατάσσονται σύμφωνα με την κίνηση των συμμετεχόντων αλλάζοντας συνεχώς νόημα και μορφή.



Σχήμα 3.16: Ένα σύννεφο από λέξεις ακολουθεί τα βήματα και τις κινήσεις των παικτών στην εγκατάσταση του Magnetize Words

Η εγκατάσταση Magnetize Words κάνει έντονη χρήση τεχνικών εντοπισμού θέσης στο χώρο.

Κεφάλαιο 4

Πρωτόκολλα

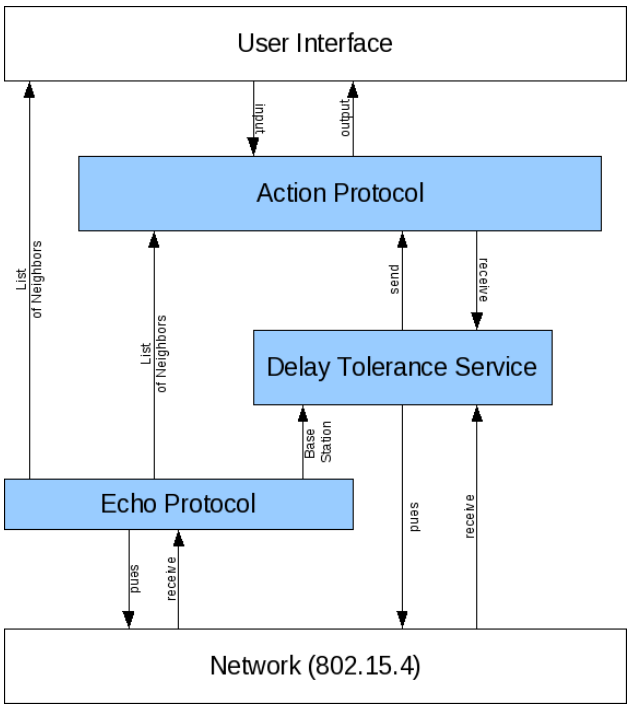
Στο κεφάλαιο περιγράφονται τα πρωτόκολλα που σχετίζονται με το χαμηλότερο επίπεδο στην αρχιτεκτονική του FinN. Πρόκειται για τους μηχανισμούς εκείνους που καθιστούν δυνατή τη λειτουργία των Guardians παρέχοντας υπηρεσίες όπως αυτές του προσδιορισμού του περιβάλλοντος των χρηστών, της αξιόπιστης επικοινωνίας και του προσδιορισμού της θέσης τους στο χώρο (βλ. Σχήμα 4.1).

4.1 Πρωτόκολλο γειτνίασης

Το Πρόβλημα Ο προσδιορισμός των συσκευών που βρίσκονται στην εμβέλεια επικοινωνίας μια συγκεκριμένης συσκευής αποτελεί ένα από τα σημαντικότερα θέματα στο χώρο των δικτύων και ιδιαίτερα σε αυτά των ασύρματων δικτύων αισθητήρων. Όταν η επικοινωνία μεταξύ μεγάλου αριθμού συσκευών γίνεται με ασύρματο τρόπο, ο προκαθορισμός των μονοπατιών σ'ένα δίκτυο και ο έλεγχός τους από έναν κεντρικό μηχανισμό καθίστανται εξαιρετικά δύσκολες λειτουργίες. Επιπρόσθετα, η επικοινωνία αυτή ενδέχεται να είναι μη συμμετρική. Σε τέτοιες περιπτώσεις, κάθε κόμβος πρέπει να είναι σε θέση να γνωρίζει ο ίδιος τους γείτονές τους, το είδος των γειτόνων του και το ρόλο του ίδιου στο δίκτυο αυτό.

Εισάγοντας έναν υψηλό βαθμό κινητικότητας σ' ένα τέτοιο δίκτυο τότε το εγχείρημα της ανακάλυψης των γειτόνων γίνεται ακόμα πιο δύσκολο καθώς η γειτονιά αλλάζει συνεχώς.

Η υλοποίηση παιχνιδιών που βασίζονται σε ασύρματα δίκτυα αισθητήρων συναντά το ίδιο ακριβώς πρόβλημα: πολλοί παίκτες που φέρουν μια συσκευή κινούνται διαρκώς με μη προκαθορισμένο τρόπο σ'ένα περιβάλλον που περιέχει άλλες συσκευές, ο ρόλος των οποίων μπορεί να είναι διαφορετικός.



Σχήμα 4.1: Τα σημαντικότερα πρωτόκολλα και υπηρεσίες στο επίπεδο των Χρηστών: Echo Protocol, Action Protocol, Delay Tolerance Service και UI.

Η Λύση Το Echo Protocol ,που είναι σχεδιασμένο να εκτελείται σε συσκευές με περιορισμένους πόρους, αποτελεί μια λύση στο παραπάνω πρόβλημα και αποτελεί τον πυρήνα της πλατφόρμας FinN.

Το πρωτόκολλο προσφέρει ένα σύνολο δυνατοτήτων που επιτρέπει στις συσκευές των παικτών (Guardians) αλλά και σ'αυτές της υποδομής να αναγνωρίζουν τις συσκευές που βρίσκονται σε εμβέλεια επικοινωνίας, να τις διαχωρίζουν ανάλογα με το ρόλο τους (Guardian, Station, Mobile Station) και να προσδιορίζουν την κατάστασή τους σε σχέση με παραμέτρους του παιχνιδιού. Επιπλέον, το Echo Protocol είναι σε θέση να αντιλαμβάνεται τις δυναμικές μεταβολές στη δομή του τοπικού δικτύου με πολύ χαμηλό χρόνο απόκρισης ενώ μπορεί να προσδιορίσει εάν η επικοινωνία με γειτονικές συσκευές είναι αμφίδρομη ή οχι (εντοπισμός asymetric links) [35].

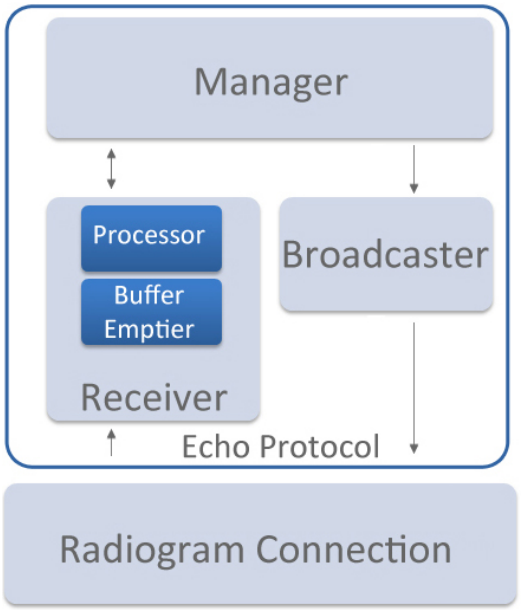
Κάθε συσκευή περιέχει δύο πίνακες σχετικούς με τις λειτουργίες του Echo Protocol:

- τον πίνακα των Γειτόνων που περιέχει τις διευθύνσεις των συσκευών με τις οποίες η αμφίδρομη επικοινωνία είναι εφικτή
- τον πίνακα των Μονόπλευρων Γειτόνων που περιέχει τις διευθύνσεις των συσκευών εκείνων από τις οποίες λαμβάνονται μηνύματα αλλά δεν μπορούν να σταλούν

Κατά τη λειτουργία το πρωτοκόλλου, κάθε συσκευή εκπέμπει περιοδικά (ανά διαστήματα `BEACON_INTERVAL`) ένα beacon μήνυμα το οποίο περιέχει τα ακόλουθα στοιχεία:

- role identifier: Guardian, Station,
- station-address: διεύθυνση του Station στο οποίο η συσκευή είναι συνδεδεμένη,
- id: το id της συκευής,
- led-id: πενταψήφιος χρωματικός κωδικός της συσκευής,
- init-phase: την φάση αρχικοποίησης της συσκευής,
- πλήθος Μονόπλευρων Γειτόνων,
- λίστα διευθύνσεων των Μονόπλευρων Γειτόνων

Όταν μια συσκευή λαμβάνει ένα beacon από μια άλλη συσκευή B, κατηγοριοποιεί τη σύμφωνα με το ρόλο της (Guardian, Station ή Mobile Station). Στη συνέχεια, η A ελέγχει εάν η δική της διεύθυνση βρίσκεται στη λίστα των



Σχήμα 4.2: Η εσωτερική αρχιτεκτονική του Echo Protocol, του πρωτοκόλλου γειτνίασης. Τα τρία βασικά αντικείμενα υπεύθυνα για τη λειτουργία του είναι ο Receiver, ο Broadcaster και ο Manager.

Μονόπλευρων Γειτόνων της συσκευής . Στην περίπτωση που κάτι τέτοιο ισχύει, η διεύθυνση της B προστίθεται στον πίνακα Γειτόνων της A αλλιώς προστίθεται μόνο στον πίνακα Μονόπλευρων Γειτόνων. Επίσης, και στους δύο πίνακες διατηρείται η χρονική στιγμή λήψης του κάθε μηνύματος (timestamp). Εάν η διεύθυνση της B τυχαίνει να υπάρχει ήδη στους πίνακες, τότε το αντίστοιχο timestamp ανανεώνεται. Εαν μετά την πάροδο συγκεκριμένου χρονικού παραθύρου, η A δεν έχει λάβει beacon από την B τότε η τελευταία δεν θεωρείται πλέον γειτονική συσκευή και διαγράφεται από τους πίνακες.

Επιπλέον, μειώνοντας την τιμή του `BEACON_INTERVAL`, είναι δυνατό η απόκριση του πρωτοκόλλου στις αλλαγές του δικτύου να πλησιάσει πραγματικό χρόνο. Αυτό έχει βέβαια σαν αποτέλεσμα να επιβαρύνεται το δίκτυο λόγω της πολύ συχνής ανταλλαγής μηνυμάτων.

Στην περίπτωση όπου η συσκευή ανακαλύψει ένα Station με το οποίο η αμφίδρομη επικοινωνία είναι δυνατή, η συνδέεται στο εφόσον δεν είναι ήδη συνδεδεμένη σε κάποιο άλλο Station.

Η Υλοποίηση Στην υλοποίησή του, το πρωτόκολλο αποτελείται από τρία βασικά στοιχεία:

1. Broadcaster

2. Receiver

3. EchoProtocolManager

Ο τρόπος που αυτά συνδέονται και η εσωτερική αρχιτεκτονική παρουσιάζονται στην Εικόνα 4.2.

Ο Broadcaster είναι ένα Νήμα της Java (Java Thread) υπεύθυνο για την δημιουργία και αποστολή των beacon μηνυμάτων με τη χρήση μιας σύνδεσης τύπου datagram σε έναν συγκεκριμένο port ECHO_PORT. Τα πεδία που συνθέτουν το μήνυμα είναι primitive Java τύποι δεδομένων και τοποθετούνται ξεχωριστά μέσα στο datagram.

Ακολουθεί μέρος του κώδικα του Broadcaster.

```
package guardian.communication.echoprotocol;
public class Broadcaster extends Thread {
//...
public final void run() {
// Setup the datagram connection
setupConnection();
while (true) {
    try {
        // Clean the Datagram
        datagram.reset();
        // Write device identifier: station or guardian
        datagram.writeInt(deviceIdentifier);
        // Get the semi-neighbours MAC addresses.
        final Enumeration semiNeighEnum =
        EchoProtocolManager.getInstance().getSemiNeighbours().
            getKeys();
        // Define the number of one-way neighbours
        final int totSemiNeighs =
        EchoProtocolManager.getInstance().getSemiNeighbours().
            size();
        // Write the number of neighbours
        datagram.writeInt(totSemiNeighs);
        // Write the corresponding MAC addresses
        while (semiNeighEnum.hasMoreElements()) {
            tmpSemiNeighMAC = (String) semiNeighEnum.nextElement
                ();
            datagram.writeUTF(tmpSemiNeighMAC);

        }
        // Write the connected station's address
        datagram.writeLong(stationAddress.longValue());
        // Write the guardianID
        datagram.writeInt(guardianID);
        // Write the Led ID
        datagram.writeInt(ledId);
```

```
        // Write the Initialization phase.
        datagram.writeInt(initPhase);
        // Send the Datagram
        dgConnection.send(datagram);
    } catch (Exception ex) {
        Logger.getInstance().debug("Could not send radiogram",
            ex);
    }
    // Wait
    Utils.sleep(EchoProtocolManager.getInstance().
        getBeaconInterval());
}
}
}
```

Ο Receiver είναι επίσης ένα Java Thread το οποίο με τη χρήση μιας server σύνδεσης ακούει εισερχόμενα μηνύματα στο ίδιο ECHO_PORT που εκπέμπει ο Broadcaster. Τη στιγμή που γίνεται λήψη ενός datagram ένα αντικείμενο τύπου Guardian ή Station δημιουργείται ανάλογα με το ρόλο πού έχει ο αποστολέας. Τα δεδομένα που εξάγονται από το ληφθέν datagram (το οποίο θα είναι και beacon) χρησιμοποιούνται για τον καθορισμό των ιδιοτήτων των αντικειμένων αυτών. Οι πίνακες Γειτόνων υλοποιούνται χρησιμοποιώντας δομές Hash Maps.

Όσον αφορά τους Μονόπλευρους Γείτονες, οι διευθύνσεις τους εισάγονται στον αντίστοιχο πίνακα (βλ. Πίνακα 4.1) μαζί με τη χρονική στιγμή λήψης του μηνύματος (ts – (**t**ime**s**tamp). Βάσει το χρόνου που έχει παρέλθει από τη χρονική στιγμή αυτή γίνεται γίνεται και ο καθαρισμός του πίνακα αυτού.

Για να καθοριστεί εάν ένας Γείτονας B είναι αμφίπλευρος (σε σχέση με τη συσκευή) γίνεται έλεγχος του πεδίου *value* που αντιστοιχεί στο κλειδί MAC_B του Πίνακα 4.1. Εάν το επιστρεφόμενο Hash Map περιέχει την διεύθυνση της τότε η θεωρεί τον ως Αμφίπλευρο Γείτονα και η τοποθετείται στο αντίστοιχο HasH Map.

Ακολουθεί μέρος του κώδικα του Receiver.

```
public class Receiver extends Thread { //NOPMD

public final void run() { //NOPMD

//Indicates if the device is a {guardian|station}
int deviceIdentifier;

// A guardian is connected to a <MAC address> or to "no station"
// A station is connected to "no station" (proper station)
//   or to "mobile station" (not a infrastucture station)
Long connectedTo;

setupConnection();
```

value	key	
MAC_A	value	key
	ts	date
	MAC_1	null
	⋮	⋮
MAC_B	ts	date
	MAC_1	null
	⋮	⋮
⋮	⋮	

Πίνακας 4.1: Ο πίνακας μονόπλευρων γειτόνων του EchoProtocol: μια συσκευή με διεύθυνση MAC_B έχει τη συσκευή με διεύθυνση MAC_1 σαν μονόπλευρο γείτονα.

```
while (true) {
try {
// Clean the Datagram
datagram.reset();
// Receive from Datagram
dgConnection.receive(datagram);
// Read device identifier: station or guardian
deviceIdentifier = datagram.readInt();

// Read the number of one-way neighbours of the sender
final int totSemiNeighs = datagram.readInt();

// Create the hash map for one-way neighbours
final HashMap semisHash = new HashMap();
semisHash.put("ts", "" + new Date().getTime());

for (int neighIndex = 0; neighIndex < totSemiNeighs; neighIndex
    ++) {
     semisHash.put(datagram.readUTF(), null);
}
        // A guardian message
        if (deviceIdentifier == Broadcaster.TOKEN_GUARDIAN) {

                final Long stationAddr = new Long(datagram.
                    readLong());
                final int id = datagram.readInt();
                final int ledId = datagram.readInt();
                final int initPhase = datagram.readInt();
```

```
            final Guardian tmpNeighbour = new Guardian();
            tmpNeighbour.setAddress(new Long(IEEEAddress.
                toLong(datagram.getAddress())));
            tmpNeighbour.setStation(stationAddr.toString());
            tmpNeighbour.setID(id);
            tmpNeighbour.setLedId(ledId);
            tmpNeighbour.setInitPhase(initPhase);


            // This is not a station one-way Neighbour
            EchoProtocolManager.getInstance().setSemisOfSemi
                (datagram.getAddress(), semisHash);

            // Set the Neighbours last alive to "now"
            tmpNeighbour.setLastAlive();

            // EchoProtocolManager should handle this
                message properly
            EchoProtocolManager.getInstance().
                updateNeighbour(tmpNeighbour);
        }
        // A station message
        } else if (deviceIdentifier == Broadcaster.TOKEN_STATION
            && !iAmStation) {
            final Station tmpStation = new Station();

            // Used to distiquence an infrastructure station
                 from a mobile one
            connectedTo = new Long(datagram.readLong());

            // This is a mobile station.
            if (!((int) connectedTo.longValue() ==
                Broadcaster.TOKEN_NOSTATION)) {
                 // Make clear that this is a mobile station
                 tmpStation.setMobileStation(true);
            }

            tmpStation.setAddress(datagram.getAddress());

            tmpStation.setStationId(datagram.readInt());
            tmpStation.setLEDId(datagram.readInt());
            datagram.readInt();

            // This is a station semi-neighbour
            EchoProtocolManager.getInstance().setSemisOfSemi
                ("" + tmpStation.getLEDId(),
                     semisHash);

            // Set the Neighbours last alive to "now"
```

```
                tmpStation.setLastAlive();

                // EchoProtocolManager should handle this
                    message properly
                EchoProtocolManager.getInstance().updateStation(
                    tmpStation);
        }
}
} catch (IOException e) {
Logger.getInstance().debug("Nothing received", e);
}
}}
```

Ο Echo Protocol Manager είναι υπεύθυνος για την διατήρηση των πινάκων σε ενημερωμένη κατάσταση είτε βάσει των beacons που έχουν ληφθεί είτε διαγράφοντας παλιές καταχωρήσεις στους πίνακες. Επιπλέον, επιτρέπει την πρόσβαση στους πίνακες Γειτόνων σε άλλα αντικείμενα και επιπλέον πληροφορίες για την συνδεσιμότητα της συσκευής τη συγκεκριμένη χρονική στιγμή ή το πλήθος και την κατάσταση των γειτονικών συσκευών.

Για να εξασφαλίσουμε ότι τα μόνο τα τελευταία beacon μηνύματα μιας συσκευής υφίστανται επεξεργασία (και όχι παλαιότερα) εισάγουμε τον Buffer Emptier στη αρχιτεκτονική του πρωτοκόλλου. Με τον τρόπο αυτό, τα ληφθέντα datagram σώζονται σε μια ενδιάμεση δομή (για παράδειγμα ταξινομημένη λίστα) έχοντας το παλαιότερο μήνυμα στην αρχή της και η επεξεργασία τους γίνεται στη συνέχεια.

Το πρωτόκολλο έχει υλοποιηθεί με τη χρήση του design pattern `Observable/Obserer` [36]. Αυτό σημαίνει πως κάθε φορά που μια αλλαγή λαμβάνει χώρα σχετικά με την κατάσταση του πρωτοκόλλου (για παράδειγμα, την ανακάλυψη ενός γείτονα ή την δημιουργία μιας σύνδεσης σ'ενα Station), οι κλάσεις οι οποίες παρακολουθούν (κάνουν δηλαδή observe) το Echo Protocol, ενημερώνονται και λαμβάνουν το αντικείμενο που έχει αλλάξει ώστε να εντοπίσουν την αλλαγή. Με το τρόπο αυτό, μειώνεται ο αριθμός των νήματος που εκτελούνται καθώς μέθοδοι άλλων αντικειμένων καλούνται μόνο όταν αυτό είναι απαραίτητο.

Ανάλογα με τις απαιτήσεις της εκάστοτε εφαρμογής, οι λειτουργίες του πρωτοκόλλου μπορούν να περιοριστούν, ώστε να γίνει οικονομία πόρων ή να επεκταθούν για να καλύψουν σε μεγαλύτερο βαθμό τις ανάγκες της. Για παράδειγμα, το περιεχόμενο των beacon μηνυμάτων μπορεί να μεταβληθεί για να την μεταφορά επιπλέον πληροφορίας, ο χρόνος εκπομπής να αυξηθεί ή να μειωθεί καθώς επίσης και ο διαχωρισμός των Μονόπλευρων από τους Αμφίπλευρος Γείτονες μπορεί να απενεργοποιηθεί.

Ακολουθεί μέρος του κώδικα του Echo Protocol Manager.

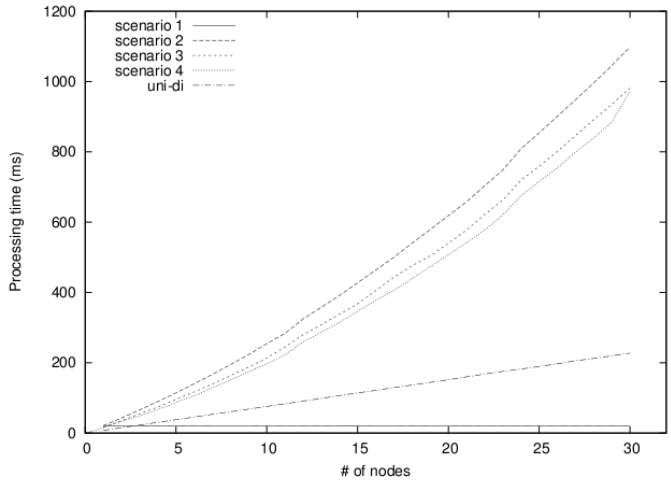
```
public final class EchoProtocolManager extends Observable {
```

```
/**
 * Either updates neighbours's timestamp
 * or requests neighbours addition to the neighbours Vector.
 *
 * @param neighbour the Guardian instance
 */
public void updateNeighbour(final Guardian neighbour) {
    // Get the semi-neighbours of this device
    final HashMap semisHash = (HashMap) semiNeighbours.get(
        neighbour.getAddress());

    // By default use MAC address to do the lookup
    Long key = myAddress;

    /*
     * Check if this neighbor is connected to some station
     */

    // A bi-directional neighbour
    if (semisHash.containsKey(key)) {

        // This is an already known neighbour
        if (neighbours.containsKey(neighbour.getAddress()))
            {
            //final Guardian thisGuardian = (Guardian)
                neighbours.get(neighbour.getAddress());
            // Update timestamp
            neighbour.setLastAlive();

            // Update the init phase of the device
            //thisGuardian.setInitPhase(neighbour.
                getInitPhase());

            // Update neighbour
            neighbours.put(neighbour.getAddress(), neighbour
                );

            // Notify observers if init phase of the
                neighbor guardians has changed
            final Guardian guardian = (Guardian) neighbours.
                get(neighbour.getAddress());
            if ((guardian.getInitPhase() != neighbour.
                getInitPhase()) || (forceNotifyObservers)) {
                this.setChanged();
                notifyObservers(neighbour);
            }
```

```
                // A new bi-directional neighbour!
                // Add her to the neighbours hashmap
            } else {
                neighbours.put(neighbour.getAddress(), neighbour
                    );
                neighbour.setAlive(true);
                this.setChanged();
                notifyObservers(neighbour);
            }

            // This device is not a bi-directional neighbour
                anymore.
            // Set neighbour to be inactive
        } else if (neighbours.containsKey(neighbour.getAddress()
            )) {
            ((Guardian) neighbours.get(neighbour.getAddress())).
                setAlive(false);

            //Remove the inactive neighbour
            neighbours.remove(neighbour.getAddress());

            // Notify Observers
            this.setChanged();
            notifyObservers(neighbour);
        }
    }
}
```

Αξιολόγηση Η αξιολόγηση του Echo Protocol που παρουσιάζεται αφορά στο χρόνο επεξεργασίας των μηνυμάτων από τις συσκευές και πως επηρεάζεται από την πυκνότητα του δικτύου. Ο χρόνος αυτός είναι εξαιρετικά σημαντικός καθώς καθορίζει την αποδοτικότητα και την ταχύτητα με την οποία οι συσκευές των παικτών αντιλαμβάνονται το περιβάλλον τους και αντιδρούν στις αλλαγές που συμβαίνουν σε αυτό. Στις παρακάτω μετρήσεις θεωρούμε ότι υπάρχουν δύο κόμβοι, ο Receiver και ο Broadcaster και εξετάζεται ο χρόνος που απαιτεί ο Receiver να επεξεργαστεί το beacon μήνυμα του Broadcaster. Έστω N_R και N_B ο αριθμός των γειτόνων του Receiver και Broadcaster αντίστοιχα, t_O ο χρόνος κατά το οποίο ο Receiver είναι απασχολημένος επεξεργαζόμενος ένα εισερχόμενο μηνύμα και t_{net} ο χρόνος που χρειάζεται ο Receiver να επεξεργαστεί όλα τα μηνύματα. Σημειώνεται πώς η λειτουργία του Echo Protocol να εντοπίζει αμφίδρομους γείτονες είναι ενεργοποιημένη εκτός αν αναφέρεται διαφορετικά.

Στα Σχήματα 4.3 και 4.4 φαίνεται η σύγκριση των διαφορετικών χρόνων που παρατηρούνται σε διαφορετικά σενάρια.



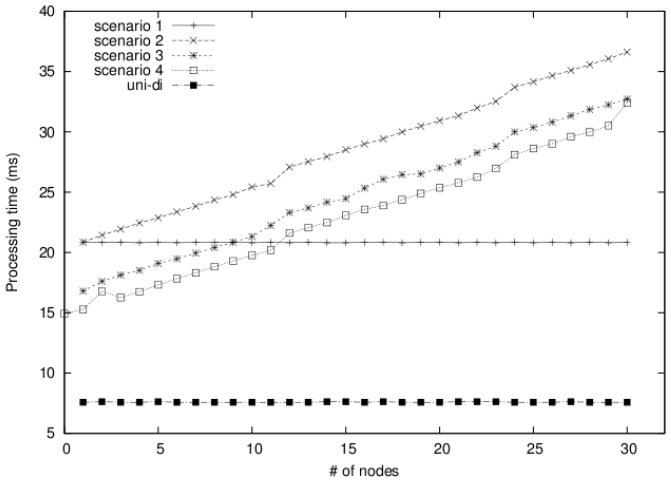
Σχήμα 4.3: Ο χρόνος t_{net} κατά τον οποίο ο Receiver του EchoProtool είναι απασχολημένος επεξεργαζόμενος ένα μήνυμα που μόλις έλαβε για διαφορετικά πειραματικά σενάρια.

Σενάριο 1 Η γειτονιά του Broadcaster παραμένει σταθερή ($N_B = 1$) ενώ η γειτονιά του Receiver αυξάνεται ($N_R = 1, 2, 3, ..., 30$). Σαν αποτέλεσμα, ο χρόνος που απαιτείται από τον Receiver να επεξεργαστεί το beacon του Broadcaster, t_O, παραμένει σταθερός. Από την άλλη πλευρά, ο χρόνος επεξεργασίας όλων των μηνυμάτων από την συνολική γειτονιά του Receiver αυξάνεται εκθετικά σε σχέση με τον αριθμό των γειτόνων N_B.

Σενάριο 2 Ο Receiver και Broadcaster αποτελούν κόμβους του ίδιου πλήρους συνδεδεμένου δικτύου με τον αριθμό των κόμβων να αυξάνεται ($N_B = N_R = 1, 2, 3..., 30$). Σε αυτή την περίπτωση το χρόνος t_O αυξάνεται, καθώς τα δεδομένα που περιέχει το beacon του Broadcaster περιλαμβάνουν περισσότερες MAC διευθύνσεις.

Σενάριο 3 Ο Broadcaster εισέρχεται σ'ένα πλήρως συνδεδεμένο δίκτυο, στο οποίο ανήκει και ο Receiver, για πρώτη φορά. Σε αυτή την περίπτωση, ο Receiver λαμβάνει ένα beacon από μη γνωστό γείτονα. Το μήνυμα από αυτόν τον γείτονα περιέχει μεταξύ άλλων την MAC διεύθυνση του Receiver. Η διαφορά εδώ, είναι πως εκτελείται η λειτουργία του Hash Map `put` σε αντίθεση με την απλή ενημέρωση στον πίνακα γειτόνων. Παρατηρείται πως η λειτουργία `put` είναι ταχύτερη.

Σενάριο 4 Πρόκειται για μία ελαφριά παραλλαγή του Σεναρίου 3. Ο Receiver λαμβάνει ένα beacon από μονόπλευρο γείτονα ($N_R = k, N_B = k - 1$). Η επε-



Σχήμα 4.4: Ο χρόνος t πρόκειται για το χρόνο που απαιτείται προκειμένου ο Receiver να επεξεργαστεί μηνύματα όλης της γειτονιάς για διαφορετικά πειραματικά σενάρια.

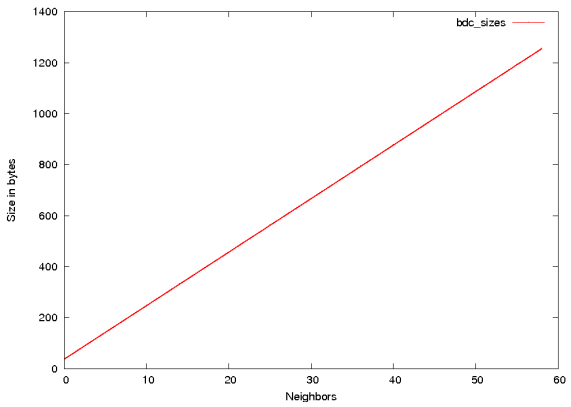
ξεργασία του μηνύματος δεν περνάει από όλα τα στάδια καθώς ο γείτονας δεν προστίθεται σε όλους τους πίνακες. Ως εκ τούτου, η χρόνος t_O αναμένεται να είναι ακόμα μικρότερος.

Unidirectional Δεν πραγματοποιείται κανένας διαχωρισμούς στο είδος των γειτόνων. Όλοι οι μονόπλευροι γείτονες προστίθενται στον πίνακα των γειτόνων ενώ δεν μεταδίδονται διευθύνσεις μονόπλευρων γειτόνων στα μηνύματα. Όπως είναι φυσικό, ο χρόνος επεξεργασίας ενός μηνύματος είναι σταθερός και πολύ χαμηλός.

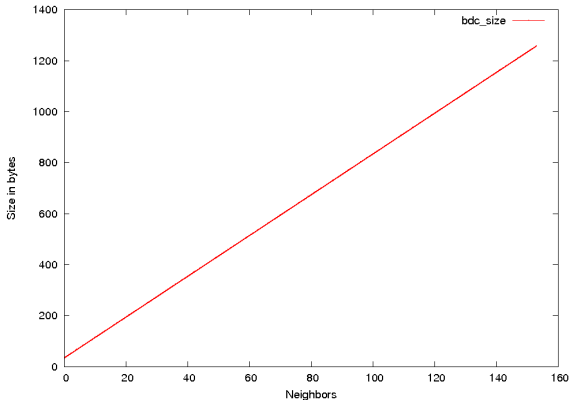
Με την τεχνική αναγνώρισης μονόπλευρων και αμφίπλευρων γειτόνων, είναι σαφές ότι επιβαρύνεται ο φόρτος του δικτύου καθώς το μέγεθος των μηνυμάτων που ανταλλάσσονται αυξάνεται κι αυτός. Στα παρακάτω διαγράμματα (Σχήματα 4.5 και 4.6) φαίνεται το μέγιστο πλήθος διευθύνσεων που μπορούν να τοποθετηθούν σε ένα datagram. Θεωρητικά αυτό είναι και το μέγιστο πλήθος γειτόνων που υποστηρίζεται. Καθώς όμως ο χρόνος εκπομπής θα είναι μεγαλύτερος όσο το datagram μεγαλώνει, οι πιθανότητες για collission αυξάνονται κι αυτές.

4.2 Αλληλεπίδραση παικτών

Το Πρόβλημα Στη γενική περίπτωση, οι παίκτες αλληλεπιδρούν μεταξύ τους είτε σε ζευγάρια είτε σε ομάδες εκτελώντας Ενέργειες (Actions). Για κάθε παιχνίδι

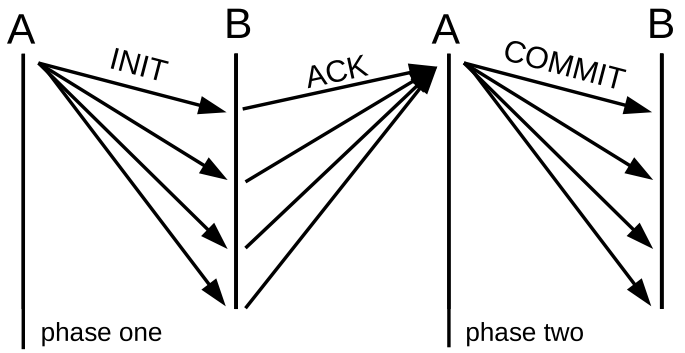


Σχήμα 4.5: Το μέγεθος ενός datagram σε bytes αναλογικά με τον αριθμό των διευθύνσεων που περιέχει. Οι διευθύνσεις είναι τύπου String.



Σχήμα 4.6: Το μέγεθος ενός datagram σε bytes αναλογικά με τον αριθμό των διευθύνσεων που περιέχει. Οι διευθύνσεις είναι τύπου long.

υπάρχει ένα σαφώς ορισμένο σύνολο Ενεργειών που μπορούν να εκτελέσουν οι παίκτες. Για την ορθή εξέλιξη ενός παιχνιδιού, είναι εξαιρετικά σημαντικό οι Ενέργειες αυτές να μην καθυστερούν στην εκτέλεσή τους – ει δυνατό να επιτυγχάνεται εκτέλεση σε πραγματικό χρόνο. Παράλληλα, είναι απαραίτητο να παρέχονται μηχανισμοί ικανοί να εγγυηθούν την αποδοτική εκτέλεσή του και την ανοχή σε σφάλματα, τα οποία είναι ιδιαίτερα συχνά σε ασύρματα δίκτυα λόγω αποτυχιών στην διαδικασία επικοινωνίας. Επιπλέον, εξίσου σημαντικό είναι ο μηχανισμός αυτός να είναι αξιόπιστος ώστε να εμποδίζει τις κακόβουλες ενέργειες κατά τη διάρκεια του παιχνιδιού και να πιστοποιεί την εφαρμογή των κανόνων.



Σχήμα 4.7: Η διαδικασία two-phase commit που ακολουθείται από το Action Protocol. Μια Ενέργεια είτε θα πραγματοποιθεί σωστά από όλους τους εμπλεκόμενους παίκτες είτε δε θα πραγματοποιηθεί καθόλου.

Η Λύση Για την επίλυση των θεμάτων που περιγράφηκαν, σχεδιάστηκε και υλοποιήθηκε το αντίστοιχο πρωτόκολλο, Action Protocol. Πρόκειται για έναν μηχανισμό δέσμευσης (commit) που υποστηρίζει διαφορετικές Ενέργειες, επιτρέπει σε περισσότερους από δύο παίκτες να αλληλεπιδράσουν ταυτόχρονα και μειώνει την πιθανότητα απόκλισης από τους κανόνες του παιχνιδιού. Ο μηχανισμός αποτελεί εφαρμογή του αλγορίθμου two phase commit [25].

Για την ενεργοποίηση του μηχανισμού, ένας παίκτης I πυροδοτεί μια συγκεκριμένη Ενέργεια A. Οι αντίπαλοι –έστω $_{1,2...n}$ ενημερώνονται πως ο πρόκειται να εκτελέσει την A με κάποιο αντίκτυπο σε αυτούς. Έπειτα από μια σειρά ελέγχων που αφορούν την κατάσταση του T σε σχέση με τον I για το αν η Ενέργεια μπορεί να εκτελεστεί, οι συμφωνούν με την ενέργεια και ενημερώνουν τον . Ο με τη σειρά του εφαρμόζει τα αποτελέσματα της στον ίδιο και ενημερώνει τους να κάνουν το ίδιο. Εάν κάποιος από τους T δεν συμφωνήσει ή δεν απαντήσει στον I τότε η Ενέργεια ακυρώνεται και οι παίκτες επιστρέφουν στην κατάσταση πριν την πυροδότησή της. Μια σχηματική απεικόνιση με τις φάσεις εκτέλεσης της διαδικασίας φαίνεται στο Σχήμα 4.7.

Η Υλοποίηση Με την έναρξη κάθε παιχνιδιού, στις συσκευές των παικτών είναι αποθηκευμένες όλες οι δυνατές ενέργειες που σχετίζονται με το παιχνίδι. Η αποθήκευση αυτή γίνεται σε ένα HashMap με κλειδί έναν αναγνωριστικό αριθμό της Ενέργειας. Κάθε Ενέργεια αποτελεί ένα αντικείμενο της Java με τα εξής σημαντικότερα χαρακτηριστικά να είναι τα εξής

- Μεταβλητή `ActionType`
  Ένας ακέραιος αριθμός μοναδικός για κάθε Ενέργεια σε κάθε παιχνίδι.

- Μεταβλητή `Targets`
  Περιέχει όλους τους αντιπάλους που αφορά η Ενέργεια σε ένα διάνυσμα.

- Μεταβλητή `Params`
  Περιέχει σε μορφή συμβολοσειράς παραμέτρους που αφορούν την εκτέλεση της Ενέργειας.

- Μέθοδος `checkAction`
  Πραγματοποιεί τον αρχικό έλεγχο για τον αν η Ενέργεια μπορεί να εκτελεστεί βάσει των τοπικών παραμέτρων. Για παράδειγμα πιστοποιείται ότι ο χρήστης έχει το δικαίωμα να πυροδοτήσει την εκτέλεση και πως υπάρχει τουλάχιστον ένας αντίπαλος τον οποίο αφορά η Ενέργεια.

- Μέθοδος `doPartA`
  Εκτελείται στη συσκευή του παίκτη που πυροδότησε την Ενέργεια εφόσον όλοι οι αντίπαλοι έχουν συμφωνήσει στην εκτέλεση. Η μέθοδος εφαρμόζει τις επιπτώσεις της Ενέργειας στον ίδιο τον παίκτη.

- Μέθοδος `doPartA`
  Εκτελείται σε κάθε μια συσκευή αντιπάλου μετά την επιβεβαίωση που στέλνεται από τον παίκτη . Η μέθοδος εφαρμόζει τις επιπτώσεις της Ενέργειας στους υπόλοιπους παίκτες.

Η σειρά εκτέλεσης των `doPartA` και `doPartA` μπορεί να διαφέρει κατά περίσταση.

Το πρωτόκολλο αποτελείται από τρείς κλάσεις:

- Receiver
- Sender
- ActionProtocolManager

Ο Sender είναι υπεύθυνος για την αποστολή το μηνύματος Ενέργειας σε όλους τους αντιπάλους. Το μήνυμα τύπου datagram περιέχει τα εξής στοιχεία:

- `ActionID`
  Πρόκειται για έναν ακέραιο που αυξάνεται κατά 1 έπειτα από κάθε Ενέργεια και συντελεί στον διαχωρισμό διαφορετικών Ενεργειών που συμβαίνουν στο ίδιο παιχνίδι.

- `ActionType`
  Ο ακέραιος που προσδιορίζει μοναδικά το είδος της Ενέργειας σε κάθε παιχνίδι. Παραμένει σταθερός και ίδιος σε κάθε στιγμιότυπο παιχνιδιού.

- PhaseNumber
  Ο ακέραιος που καθορίζει την τρέχουσα φάση εκτέλεσης του πρωτοκόλλου. Αυτή μπορεί να είναι μία από τις:
  - Φάση αρχικοποίησης
  - Φάση αναγνώρισης
  - Φάση συμφωνίας
- Parameters
  Περιέχει τη συμβολοσειρά που περιγράφει τις παραμέτρους που αφορούν την Ενέργεια.

Το μήνυμα αυτό αποστέλλεται σειριακά στο διάνυσμα Targets που περιέχει η Ενέργεια.

Ο Receiver είναι ένα Java Thread που ακούει για εισερχόμενα μηνύματα σε μια datagram σύνδεση τύπου server. Με τη λήψη κάθε μηνύματος, εκτελείται η αντίστοιχη μέθοδος ανάλογα με τη φάση εκτέλεσης. Όταν λαμβάνεται μήνυμα αρχικοποίησης ο παραλήπτης $_i$, με $i = 1, 2, ..., n$ απαντάει με μήνυμα αναγνώρισης στον εφόσον οι απαραίτητοι έλεγχοι είναι αληθείς. Τα μηνύματα αναγνώρισης λαμβάνονται από τον . Εφόσον ληφθούν και τα n μηνύματα, ο περνά στη φάση συμφωνίας και καλεί την doPartA της Ενέργειας. Η λήψη ενός μηνύματος συμφωνίας έχει σαν αποτέλεσμα την κλήση της μεθόδου doPartB εκ μέρους του αντίπαλου παίκτη.

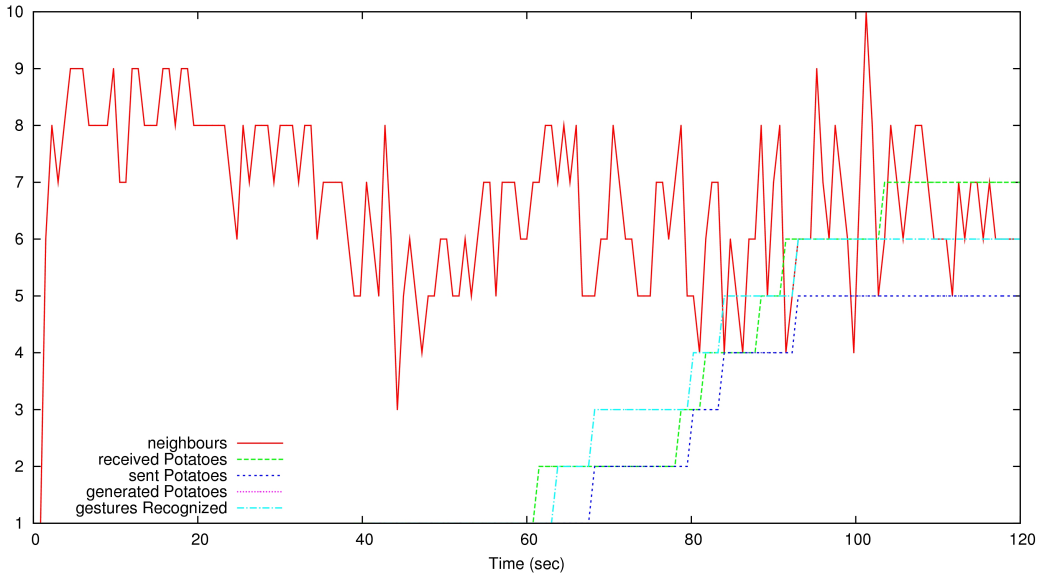
Ο ActionProtocolManager παρέχει πρόσβαση σε όλες τις γνωστές Ενέργειες του εκάστοτε παιχνιδιού και διατηρεί ενημερωμένο το τρέχον ActionID που υπάρχει στο δίκτυο.

Ακολουθεί μέρος του κώδικα του Action της Καυτής Πατάτας.

```
public class PotatoAction extends AbstractGuardianAction { //
    NOPMD

/**
* Checks if this action will be performed or not based on local
    rules.
*
* @return True if the action must be performed
*/
protected final boolean checkAction() {
// Ready to send the Potato.
if (!potman.getActiveNeighbours().isEmpty()
        && potman.isActive()
        && potman.getPenaltyTimer() == 0) {

    findTarget();
```

```
    return true;
    // Perfoming gesture without the Potato.
} else if (!potman.isActive()) {
    return false;
    // No neighbours
} else {
    return false;
}
}

/**
* Finds a random target from the neighbouring players.
*/
protected final void findTarget() {
// Get a random target
final Vector possibleTargets = PotatoManager.getInstance().
   getActiveNeighbours();
// No neighbours
if (possibleTargets.isEmpty()) {
        return;

} else {
    // Choose a random neighbour and add him as target
    final int targetIndex = RANDGEN.nextInt(possibleTargets.size
       ());
    final Guardian receiverGuardian = (Guardian) possibleTargets
       .elementAt(targetIndex);
    this.addTarget(receiverGuardian);
}
}

/**
* Local results of the PotatoAction.
* Upon success Potato passage, the local Potato set to be
   inactive
*/
protected final void doPartA() {
// Deactivate potato
potman.setActive(false);

// Initialize the Generation timer to delay
// Potato generation
potman.initGenTimer();

// Get Target of potato
final Guardian target = (Guardian) getTargets().elementAt(0);
//FinnLogger.getInstance().potatoSent();

// Create passing event
```

```
final Event event = new Event();

event.setGuardianID(GuardianInfo.getInstance().getGuardian().
    getID());
event.setType("HotPotato");
event.setTimeStamp(new Date());
event.setDescription("Potato Sent to " + target.getID() + " [" +
     target.getAddress() + "]");

// Send the event to some Base Station
DTSManager.getInstance().sendEvent(event);
    }

/**
* Remote result of the PotatoAction.
* Invoked when a Potato is received
*/
protected final void doPartB() {

// Set the correct boomTimer value
final int receivedBoomTimer = Integer.parseInt(this.getParams())
    ;
logger.debug("PotatoTimer " + receivedBoomTimer);

// Create Event for this actions
final Event event = new Event();

event.setGuardianID(GuardianInfo.getInstance().getGuardian().
    getID());
event.setType("HotPotato");
event.setTimeStamp(new Date());

// There is an active potato on the system.
if (receivedBoomTimer > potman.getBoomTimer() && potman.isActive
    ()) {
    //FinnLogger.getInstance().mergedPotato();
    // Create Merging event
    event.setDescription("Merged Potatos");

    // Send the event to some Base Station
    DTSManager.getInstance().sendEvent(event);

    logger.debug("You already have and older potato!");

    // Receive the Potato.
} else {
    //FinnLogger.getInstance().potatoReceived();
    // Create Receiving Potato event
    event.setDescription("Received Potato");
```

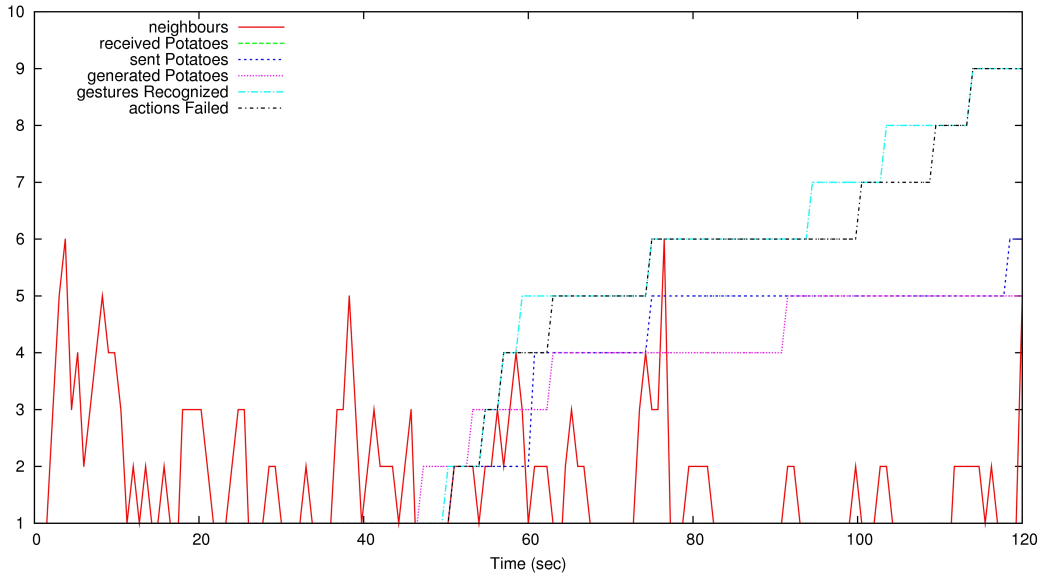


Σχήμα 4.8: Οι παίκτες, οι Καυτές Πατάτες και οι αλληλεπιδράσεις

```
    // Send the event to some Base Station
    DTSManager.getInstance().sendEvent(event);

    potman.setActive(true);
    potman.setBoomTimer(receivedBoomTimer);
}

}
}
```

Αξιολόγηση Προκειμένου να αξιολογηθεί η αποδοτικότητα και η σωστή λειτουργία του Action Protocol σε συνδυασμό και με τα υπόλοιπα πρωτόκολλα και υπηρεσίες της πλατφόρμας, καταγράφηκε ένα σύνολο δεδομένων κατά την εξέλιξη ενός παιχνιδιού Hot Potato. Για να γίνει δυνατή αυτή η καταγραφή χρησιμοποιήθηκε για μια ακόμη φορά το software design pattern των Observable/Observer αντικειμένων. Για παράδειγμα, κάθε φορά που μια χειρονομία γινόταν αντιληπτή από τη συσκευή ή μια συμφωνία του action protocol αρχικοποιείτο, το γεγονός καταγραφόταν από τα αντικείμενα Observer του πειράματος. Τα δεδομένα καταγράφονται στη συσκευή και μετά το τέλος του παιχνιδιού αποστέλλονται σε ένα Station για περαιτέρω επεξεργασία.

Στοχεύοντας στην αξιολόγηση της υλοποίησης αλλά και την τρόπο διεξαγωγής του παιχνιδιού, έγινε λεπτομερής καταγραφή των εξής παραμέτρων:



Σχήμα 4.9: Οι παίκτες, οι Καυτές Πατάτες και οι αλληλεπιδράσεις

- **Γείτονες** – Neighbors
  Ο αριθμός των αμφίπλευρων γειτόνων που έχει η κάθε συσκευή σε κάθε χρονική στιγμή.

- **Πατάτες που λήφθησαν και εστάλησαν** – Potatos sent/received
  Ο αριθμός από "Καυτές Πατάτες" που έχει λάβει ή στείλει κάθε παίκτης.

- **Πατάτες που δημιουργήθηκαν** – Potatoes generated
  Ο αριθμός από "Καυτές Πατάτες" που δημιουργήθηκαν στην συσκευή κάθε παίκτη.

- **Χειρονομίες που εντοπίστηκαν** – Gestures recognized
  Ο αριθμός των χειρονομιών που αναγνωρίστηκαν επιτυχώς από τη συσκευή.

- **Αποτυχημένες Ενέργειες** – Failed actions
  Οι φορές που οι παίκτες προσπάθησαν να "πετάξουν" την Καυτή Πατάτα άλλα η ενέργεια δεν ολοκληρώθηκε επιτυχώς.

- **Χρήση CPU** – CPU usage
  Εκτίμηση της χρήσης του επεξεργαστή κατά τη διάρκεια του παιχνιδιού.

- **Χρήση Μνήμης** – Memory usage
  Το ποσό της μνήμης που καταλαμβάνουν τα αντικείμενα του παιχνιδιού.

Για τις εν λόγω γραφικές παραστάσεις, έχει επιλεχθεί και εξετάζεται ενδεικτικά η δραστηριότητα ενός παίκτη κάθε φορά.

Στο Σχήμα 4.8 παρουσιάζεται η δυναμικότητα και οι αλλαγές που υπήρξαν στο δίκτυο σε ένα παιχνίδι που διάρκειας 2 λεπτών. Υπάρχουν σημεία όπου οι γειτονιά του παίκτη άλλαξε από 3 σε 9 γείτονες σε διάστημα μικρότερο των τριών δευτερολέπτων. Το γεγονός επιβεβαιώνει την υψηλή δυναμικότητα που παρουσιάζεται στα παιχνίδια αυτού του είδους ενώ ενισχύει την επιλογή του IEEE 802.15.4 σαν κατάλληλο πρωτόκολλο επικοινωνίας καθώς μπορεί να αντιληφθεί αλλαγές τέτοια ταχύτητας στο δίκτυο.

Σχετικά με τις στρατηγικές που ακολουθεί ο παίκτης, έντονη δραστηριότητα παρατηρείται για τουλάχιστον ένα λεπτό όπου και παίκτης δεν έχει καμία Καυτή Πατάτα. Μετά από αυτό το σημείο ο παίκτης λαμβάνει μια πατάτα από κάποιον συμπαίκτη του. Δύο δευτερόλεπτα έπειτα αντιλαμβάνεται το γεγονός και εκτελεί τη χειρονομία προκειμένου να την στείλει σε άλλο συμπαίκτη του. Η κίνηση αναγνωρίζεται επιτυχώς από τη συσκευή και ενεργοποιείται ο μηχανισμός του Action Protocol. Απαιτούνται περίπου 3 δευτερόλεπτα για την εκτέλεση όλων των φάσεων του πρωτοκόλλου. Συνολικά η Καυτή Πατάτα αλλάζει παίκτη τέσσερις φορές πριν καταλήξει να "εκραγεί".

Διαφορετική στρατηγική και πλήθος αλληλεπιδράσεων έχει επιλέξει ο παίκτης τα στατιστικά του οποίου παρουσιάζονται στο Σχήμα 4.9. Ο παίκτης αυτός παραμένει απομακρυσμένος χωρίς να συναντά παρά λίγους συμπαίκτες του. Δεν λαμβάνει καμία πατάτα κατά τη διάρκεια του παιχνιδιού μέχρι που τελικά, μετά από 40 δευτερόλεπτα και λόγω της απομόνωσής του η συσκευή του δημιουργεί μια Καυτή Πατάτα. Καθώς ήταν απομακρυσμένος από τους υπόλοιπους παίκτες δεν κατάφερε να διώξει την Καυτή Πατάτα παρά το ότι εκτέλεσε σωστά την αντίστοιχη κίνηση. Λόγω της δυναμικότητας και της αδρότητας του δικτύου το Action Protocol απέτυχε.

4.3 Delay Tolerance Service

Το Πρόβλημα Κατά τη διάρκεια του παιχνιδιού, οι παίκτες μπορεί –είτε ηθελημένα είτε όχι– να απομακρυνθούν από την περιοχή κάλυψης της κεντρικής υποδομής. Μάλιστα, κάτι τέτοιο ενδέχεται να εμπεριέχεται και στο σχεδιασμού του ίδιου του παιχνιδιού, όπως συμβαίνει σε αυθόρμητα και αδόμητα παιχνίδια. Σε αυτή την περίπτωση οτιδήποτε συμβαίνει στο επίπεδο των Παικτών παραμένει άγνωστο στα ανώτερα επίπεδα.

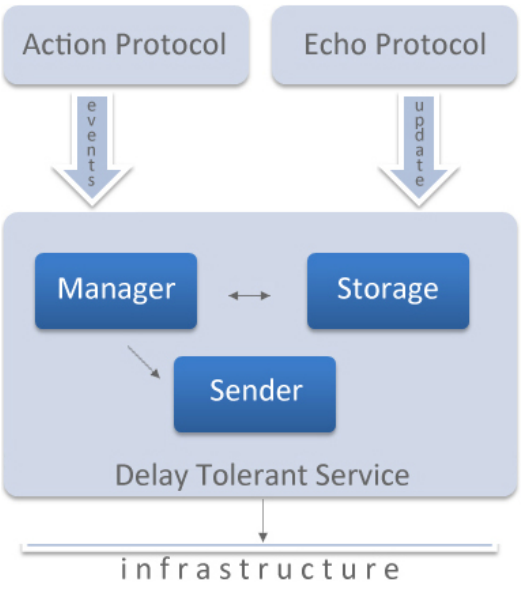
Κάτι τέτοιο, ωστόσο, δεν θα πρέπει να επηρεάζει την εξέλιξη του παιχνιδιού ή να απασχολεί τους παίκτες. Η εφαρμογή απαιτείται να να είναι σε θέση να συνεχίζει να εκτελείται χωρίς να παρουσιάζονται σφάλματα στο σύστημα. Τα δεδομένα που έχουν δημιουργηθεί δε θα πρέπει, όμως, να χάνονται αλλά να

αποστέλλονται στην κεντρική υποδομή σε μελλοντικό χρόνο.

Η Λύση Για να διασφαλιστεί η εύρυθμη λειτουργία του συστήματος όταν οι παίκτες ενεργούν αποσυνδεδεμένοι από την κεντρική υποδομή χρειάζεται να δημιουργηθεί ένας μηχανισμός ο οποίος αφενός θα αποθηκεύει τα δεδομένα όταν η αποστολή τους δεν είναι δυνατή και αφετέρου θα εξασφαλίζει την προώθησή τους όταν κάτι τέτοιο είναι εφικτό. Η όλη διαδικασία θα πρέπει να είναι διαφανής τόσο για τον χρήστη όσο και για τον προγραμματιστή.

Η Υλοποίηση Για την επίλυση του ζητήματος της ανοχής σε καθυστερήσεις σχεδιάστηκε και υλοποιήθηκε το Delay Tolerance Service – DTS.

Η υπηρεσία αυτή παρεμβάλλεται μεταξύ του οποιουδήποτε μηχανισμού παραγωγής δεδομένων και αυτόν της αποστολής τους.



Σχήμα 4.10: Η εσωτερική αρχιτεκτονική του Delay Tolerance Service εγγυάται την ενημέρωση της κεντρικής υποδομής με τις ενέργειες του παίκτη όποτε αυτό είναι δυνατό.

Οι λειτουργίες της υπηρεσίας εκτελούνται από δύο αντικείμενα:

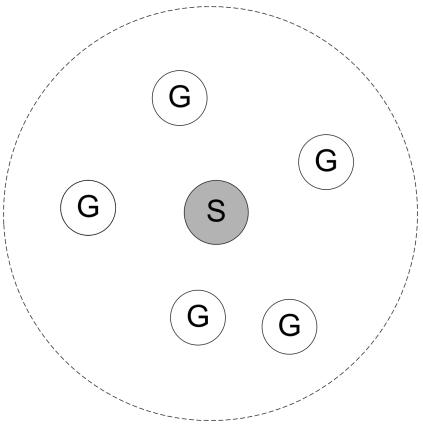
- DTSManager
- EventSender

Οι συνδέσεις μεταξύ των αντικειμένων αυτών καθώς με τις άλλες υπηρεσίες του συστήματος φαίνονται στο Σχήμα 4.10.

Ο DTSManager παραλαμβάνει τα μηνύματα που πρόκειται να σταλούν και εκτελεί έναν έλεγχο της συνδεσιμότητας της συσκευής με την κεντρική υποδομή. Ο έλεγχος αυτός μπορεί να επιστρέψει τρεις διαφορετικές καταστάσεις:

1. σε Σύνδεση
2. σε έμμεση Σύνδεση
3. σε Αποσύνδεση

Στην πρώτη περίπτωση (Σχήμα 4.11), η συσκευή βρίσκεται στην εμβέλεια της κεντρικής υποδομής και τα δεδομένα προωθούνται αμέσως σε αυτή. Δεν εισάγεται καμία καθυστέρηση στην ενημέρωση των ανώτερων επιπέδων.

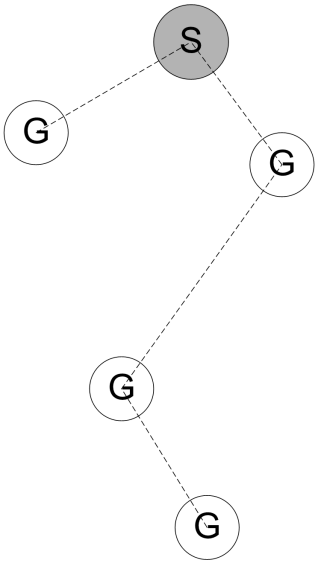


Σχήμα 4.11: Περίπτωση όπου όλοι οι παίκτες (G) είναι συνδεδεμένοι στην κεντρική υποδομή (S). Τα μηνύματα παραδίδονται απευθείας.

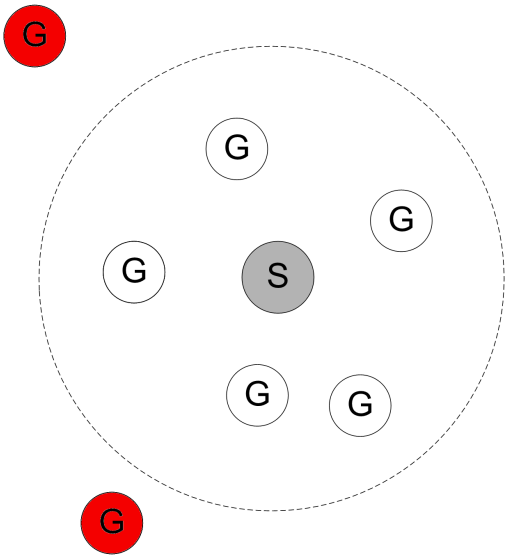
Η δεύτερη περίπτωση υποδηλώνει πως κάποιος γείτονας βρίσκεται σε ένα σύνδεση με την υποδομή. Αυτό σημαίνει πως είναι δυνατή η δημιουργία ενός μονοπατιού με σκοπό την παράδοση των δεδομένων (βλ. Σχήμα 4.12). Σε μια τέτοια κατάσταση, οι παίκτες λειτουργούν ως ενδιάμεσοι κόμβοι προωθώντας τα ληφθέντα μηνύματα στον επόμενο κόμβο που βρίσκεται πιο κοντά στην υποδομή. Το μέγιστο μήκος του μονοπατιού που σχηματίζεται μπορεί να οριστεί κάθε φορά.

Στην τρίτη περίπτωση η επικοινωνία με την υποδομή είναι αδύνατη (βλ. Σχήμα 4.13). Τότε τα δεδομένα αποθηκεύονται στη μόνιμη μνήμη της συσκευής κάθε παίκτη.

Προκειμένου ο DTSManager ενημερωθεί όταν η σύνδεση με την υποδομή αποκατασταθεί ακολουθείται το design pattern των Observable/Observer αντικειμένων με τον DTSManager να παρακολουθεί το EchoProtocol και την κατάσταση της



Σχήμα 4.12: Δημιουργία μονοπατιού μεταξύ παικτών και κεντρικής υποδομής. Τα μηνύματα προωθούνται σε άλλους παίκτες προκειμένου να παραδοθούν.



Σχήμα 4.13: Οι κόκκινοι παίκτες βρίσκονται εκτός εμβέλειας. Τα μηνύματα αποθηκεύονται και αποστέλλονται σε κάποια μελλοντική χρονική στιγμή.

συνδεσιμότητας της συσκευής. Τη στιγμή εγκαθίδρυσης σύνδεσης ο DTSManager

δημιουργεί ένα Java Threas, τον Sender.

Ο Sender είναι υπεύθυνος για την ανάκτηση των αποθηκευμένων δεδομένων από τη μνήμη της συσκευής, τη δημιουργία μηνυμάτων που να τα περιέχουν και την τελική αποστολή τους είτε απευθείας στην υποδομή, είτε σε κάποιον ενδιάμεσο κόμβο. Καθώς η διαδικασία αυτή ενδέχεται να διαρκέσει, αυτός είναι και ο λόγος που δημιουργείται ξεχωριστό Thread, αφοσιωμένο στην αποστολή των δεδομένων. Εάν για οποιοδήποτε λόγω η αποστολή δεν μπορέσει να ολοκληρωθεί, τα εναπομείναντα δεδομένα αποθηκεύονται και πάλι στη μόνιμη μνήμη.

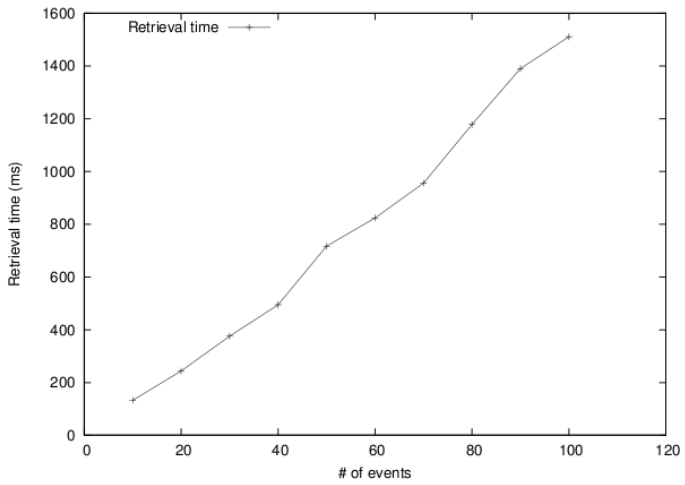
Ακολουθεί μέρος του κώδικα DTSManager.

```
public synchronized void sendEvent(final Event event) {
        // Check the connection to the station
        if (EchoProtocolManager.getInstance().getMyStation().
            isActive()) {
            try {
                // Creates a broadcast Datagram Connection
                dgConnection = (DatagramConnection) Connector.
                    open("radiogram://"
                            + EchoProtocolManager.getInstance().
                                getMyStation().getAddress().longValue
                                ()
                            + ":" + DTSPORT);

                // Creates a Datagram using the above Connection
                final Datagram datagram = dgConnection.
                    newDatagram(dgConnection.getMaximumLength());

                // Clean the Datagram
                datagram.reset();
                // Convert event to byte Array
                final byte[] eventArray = event.toByteArray();
                // Send Class Type
                datagram.writeUTF(event.getClass().getName());
                // Send length
                datagram.writeInt(eventArray.length);
                // Send array
                datagram.write(eventArray, 0, eventArray.length)
                    ;
                // Send the datagram
                dgConnection.send(datagram);
                dgConnection.close();

                // Something went wrong. Add events back to
                    storage.
            } catch (Exception ex) {
                Logger.getInstance().debug("Unable to send event
                    ", ex);
                Logger.getInstance().debug("Adding event to
```

```
                    storage: " + event.getDescription());
                // Close the open connection
                setDisconnected();

                // The attempt has failed
                failedUpdate = true;

                // Add event back to storage
                StorageService.getInstance().add(event);
}
```

Αξιολόγηση Μια σημαντική παράμετρος στη λειτουργία του μηχανισμού DTS, αποτελεί ο απαιτούμενος χρόνος για την ανάκτηση των αποθηκευμένων δεδομένων από τη Μνήμη. Ο χρόνος αυτός επηρεάζει τον συνολικό χρόνο απόκρισης και αποκτά βαρύτητα καθώς το χρονικό διάστημα για το οποίο ένας παίκτης βρίσκεται στην κάλυψη της υποδομής ενδέχεται να είναι μικρό.

Είναι αναμενόμενο, ο χρόνος ανάκτησης να μεγαλώνει όσο αυξάνεται ο αριθμός των Συμβάντων τα οποία βρίσκονται αποθηκευμένα και όσο μεγαλύτερο είναι το χρονικό διάστημα που οι παίκτες είναι αποσυνδεδεμένοι. Στην Εικόνα 4.14 απεικονίζεται ο χρόνος που απαιτείται για την ανάκτηση των Συμβάντων από τη μνήμη Flash και η τοποθέτησή τους σε έναν προσωρινό buffer, σε σχέση με τον αριθμό των αποθηκευμένων Events.



Σχήμα 4.14: Ο χρόνος ανάκτησης των αποθηκευμένων Συμβάντων από τη flash μνήμη του Sun SPOT.

Έναν τρόπος να αυξηθεί η περιοχή κάλυψης της υποδομής είναι να αυξηθεί ο αριθμός των μέγιστων βημάτων (max hop count) για τα οποία τα beacon

μηνύματα ενός Σταθμού αναμεταδίδονται. Όσο βολική κι αν φαίνεται αυτή η λύση, ωστόσο δημιουργεί προβλήματα. Το γεγονός της συνεχούς αναμετάδοσης των ίδιων μηνυμάτων έχει ως αποτέλεσμα τη δυσανάλογη αύξηση του φόρτου δικτύου στις περιοχές γύρω από τους Σταθμούς.

Το φαινόμενο αυτό οφείλεται στο ότι όλοι οι κόμβοι προωθούν και επεξεργάζονται τα beacon μηνύματα του Σταθμού περισσότερες από μία φορές χωρίς να είναι απαραίτητο. Συγκεκριμένα, σ'ένα πλήρως συνδεδεμένο δίκτυο αποτελούμενο από n κόμβους υπό την κάλυψη ενός Σταθμού, κάθε κόμβος λαμβάνει και επεξεργάζεται $(n-1)^{max\ hop\ count-1}$ μηνύματα. Για το λόγο αυτό, ο ορισμός των μέγιστων βημάτων αναμετάδοσης θα πρέπει να ορίζεται μεγαλύτερος τους 1 μόνο σε περιπτώσεις όπου η υποδομή είναι αδρή.

Ο DTS σε όλα τα επίπεδα της αρχιτεκτονικής Στο σημείο αυτό, μελετάται η επίδραση του DTS σε όλα τα επίπεδα της αρχιτεκτονικής του συστήματος. Υποθέτουμε αρχικά ότι η συσκευή είναι ήδη σε επικοινωνία με την υποδομή και θεωρούμε τους εξής χρόνους:

- $time_G$: ο χρόνος επεξεργασίας ενός Συμβάντος στο επίπεδο του Guardian πριν σταλεί στην υποδομή.
- $time_S$: ο χρόνος που ένα Συμβάν παραμένει στο επίπεδο του Station.
- $time_E$: ο χρόνος που απαιτείται από το υψηλότερο επίπεδο –Engine– μέχρι το Συμβάν να είναι διαθέσιμο.

Κατά τη διάρκεια των πειραμάτων, μεταδίδονται 100 Events κάθε φορά. Οι χρόνοι σε κάθε ένα επίπεδο ξεχωριστά έχουν ως εξής:

- $time_G$ – 38ms
- $time_S$ – 2.5ms
- $time_E$ – 5.5ms

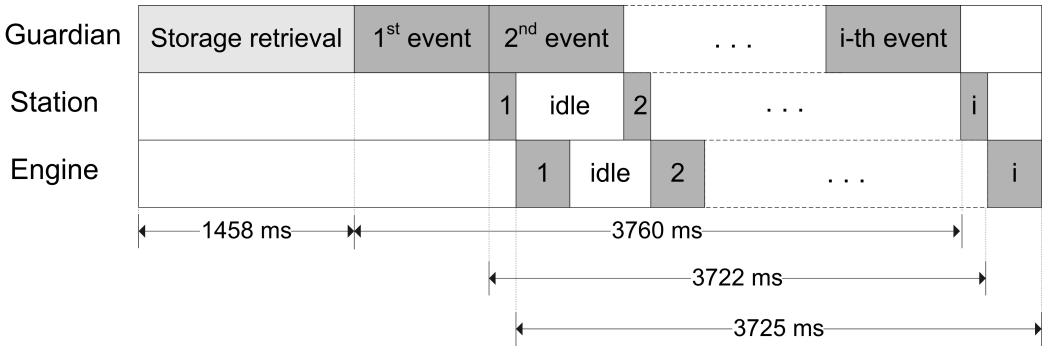
Είναι σαφές ότι το μεγαλύτερο ποσοστό (περίπου 83%) καταναλώνεται στο τελευταίο επίπεδο. Η καθυστέρηση αυτή οφείλεται στο μεγαλύτερο βαθμό σε εγγενείς καθυστερήσεις στην επικοινωνία που αφορούν τα Sun SPOT και που χρησιμοποιούνται και από την πλατφόρμα FinN. Οι λειτουργίες αυτές θέτουν και τα όρια αποδοτικότητας στην επικοινωνία καθώς καταλαμβάνουν το μεγάλο αυτό ποσοστό.

Στη συνέχεια αξιολογείται η λειτουργία του DTS στην περίπτωση όπου ένας αριθμός από Guardians συνδέεται σε ένα μοναδικό Station ενώ όλοι ταυτόχρονα

πραγματοποιούν αποστολή αποθηκευμένων Event. Τώρα, εξετάζεται ο απαιτούμενος χρόνος ώστε τα Events να αποθηκευτούν σε μια MySQL βάση δεδομένων σε σχέση με τον αριθμό των Guardians που πραγματοποιούν την αποστολή. Ο αριθμός των Events που αποστέλλεται παραμένει σταθερός και ίσος με 100, ενώ αυξάνεται αυτός των Guardians. Ιδιαίτερο ενδιαφέρον παρουσιάζει ο Λόγος Λήψης Συμβάντων/Χρόνου. (event reception rate) στο επίπεδο των Station καθώς επίσης και ο Λόγος Επεξεργασίας Συμβάντων/Χρόνου (event processing rate) στα επίπεδα Station και Engine

Αρχικά το πείραμα εκτελείται χρησιμοποιώντας έναν Guardian. Σε αυτή την περίπτωση ο Λόγος Λήψης Συμβάντων/Χρόνου είναι 37.6 ms/Συμβάν ενώ ο Λόγος Επεξεργασίας Συμβάντων/Χρόνου είναι 2.4ms/Συμβαν στο επίπεδο των Station και 5.3ms/Συμβάν sto επίπεδο του Engine. Είναι εμφανές ότι οι ρυθμοί επεξεργασίας είναι σημαντικά υψηλότεροι από τους ρυθμούς λήψης. Για το λόγο αυτό παρουσιάζεται ένα χρόνος της τάξης των 35.2 και 32.3 millisecond για το επίπεδο του Station και του Engine αντίστοιχα κατά το οποίο τα επίπεδα περαμένουν ανενεργά. Παρατηρείται λοιπόν το φαινόμενο συμφόρησης (bottleneck) στο πιο χαμηλό επίπεδο, αυτό των Guardian. Το φαινόμενο δικαιολογείται αν λάβει κανείς υπόψιν του τους περιορισμούς των συσκευών που ήδη αναφέρθηκαν.

Επιπλέον, η υλοποίηση που έχει γίνει επιτρέπει την ασύγχρονη επεξεργασία των Συμβάντων. Ως εκ τούτου, παρατηρείται το φαινόμενο pipelining όσο η πληροφορία ανεβαίνει σε επίπεδα. Το φαινόμενο αυτό και οι χρόνοι σε κάθε επίπεδο φαίνονται στο Σχήμα 4.15. Λόγω του φαινομένου, ο χρόνος που απαιτούν 100 Events για να εγγραφτούν σε μια βάση δεδομένων είναι $total_G + time_S + time_E$.



Σχήμα 4.15: Το φαίνόμενο pipeline μεταξύ των τριών επιπέδων, Guardian, Station και Engine. Ένας Guardian ενημερώνει την υποδομή αποστέλλοντας i Συμβάντα. Οι χρόνοι που σημειώνονται ισχύουν για $i = 100$.

Για την δημιουργία ενός ρεαλιστικού σεναρίου, αυξάνεται σταδιακά ο αριθμός των συσκευών που εκτελούν την παραπάνω διαδικασία με σκοπό την αναπαραγωγή της κατάστασης όπου ένας μεγαλύτερος αριθμός Guardians ενημερώνει

ταυτόχρονα την υποδομή. Περιμένει κανείς ότι με αυτή τη διαδικασία ο Λόγος Λήψης Συμβάντων/Χρόνου θα μειωθεί στο ελάχιστο και έτσι ο χρόνος αδράνειας στα παραπάνω επίπεδα θα εξαλειφθεί. Τα αποτελέσματα των μετρήσεων παρουσιάζονται τον Πίνακα 4.2.

	Guardian	Station		Engine		
Κόμβοι	Συνολο	Σύνολο	Επεξ.	Σύνολο	Επεξ.	Σύνολο
1	3760	3722	242	3725	531	3786
2	6031	5993	272	5996	840	6036
3	7106	7068	425	7070	1216	7111
4	8709	8669	511	8674	1839	8715
5	9115	9074	620	9079	2295	9121
6	21009	20969	768	20973	2648	21015

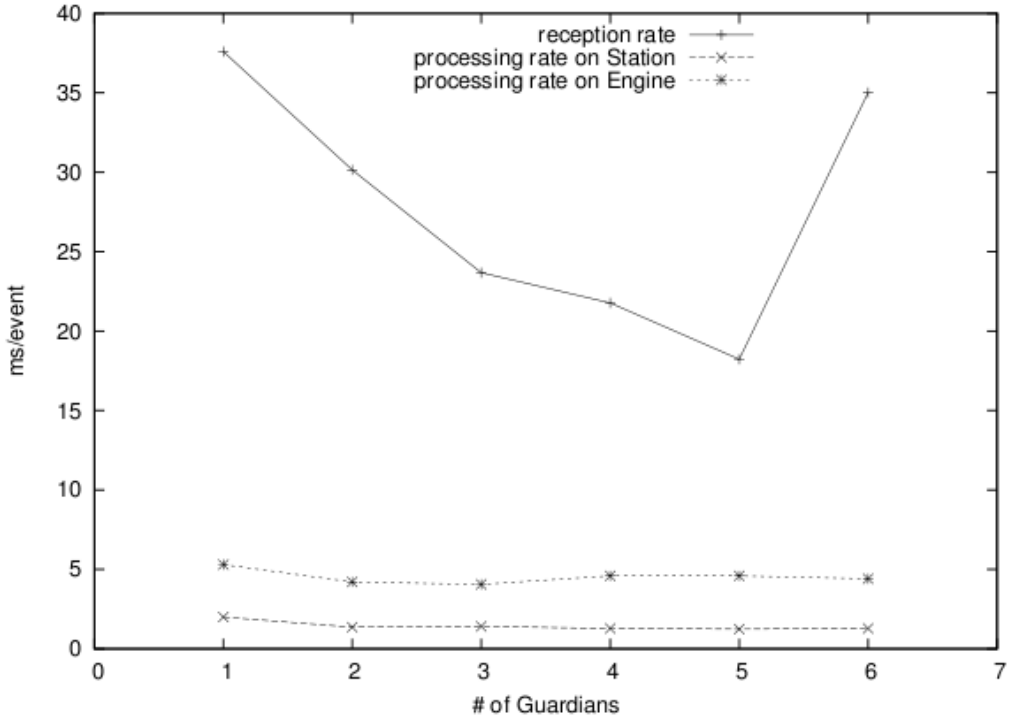
Πίνακας 4.2: Κατανομή του χρόνου (σε milliseconds) μεταξύ των τριών επιπέδων. Λόγω του φαινομένου bottleneck ο Συνολικός χρόνος είναι περίπου ίδιος με τον συνολικό χρόνο στο επίπεδο των Guardian. Ο τίτλος Επεξ. υποδηλώνει το χρόνο επεξεργασίας σε κάθε επίπεδο.

Παρατηρεί κανείς στο Σχήμα 4.16 πως, όντως αρχικά, όταν ο αριθμός των Guardians αυξάνεται, ο Λόγος Λήψης Συμβάντων/Χρόνου μειώνεται. Ωστόσο, η τάση αυτή μεταστρέφεται όταν προστίθεται μια έκτη συσκευή. Φαίνεται πως η αυξημένη πυκνότητα του δίκτυου καθώς επίσης ο μεγάλος όγκος Συμβάντων έχει αυξημένες απαιτήσεις στους ήδη περιορισμένους πόρους συστήματος των Sun SPOT. Επιπλέον, η επικάλυψη των πακέτων του 802.15.4 έχουν ως αποτέλεσμα αναμεταδόσεις με αποτέλεσμα την αύξηση του Ρυθμού Λήψης Συμβάντων/Χρόνου. Με την περαιτέρω αύξηση των Guardiann αύξηση αυτή γίνεται μεγαλύτερη και το δίκτυο ασταθές.

4.4 Εντοπισμός θέσης

Το Πρόβλημα Σε κάποια παιχνίδια οι συσκευές των παικτών χρειάζεται να γνωρίζουν τη θέση τους σε σχέση με συγκεκριμένα σημεία στο χώρο. Μια απλοϊκή προσέγγιση στην εκτίμηση του πόσο κοντά βρίσκονται οι παίκτες σε άλλους παίκτες ή σε άλλα σημεία στο χώρο γίνεται με τη χρήση των μηνυμάτων του Echo Protocol. Ρυθμίζοντας κατάλληλα την ισχύ του εκπεμπόμενου σήματος και θεωρώντας ότι η παίκτες κρατούν τη συσκευή πάντα με τον ίδιο τρόπο, μια συσκευή γνωρίζει εάν βρίσκεται ή όχι "κοντά" σε ένα ένα άλλο αντικείμενο.

Η προσέγγιση αυτή όμως, είναι ικανοποιητική σε ένα μόνο βαθμό. Σε άλλες περιπτώσεις ο εντοπισμός θέσης χρειάζεται να έχει περισσότερη ακρίβεια και



Σχήμα 4.16: Ο Λόγος Παραλαβής και Επεξεργασίας Συμβάντων/Χρόνου σε κάθε επίπεδο.

να παρέχει πληροφορία για την απόλυτη θέση του παίκτη στο χώρο. Μια τέτοια διαδικασία αποδεικνύεται ιδιαίτερα απαιτητική στα ασύρματα δίκτυα αισθητήρων. Η χρήσης GPS δεν θεωρείται πρακτική λύση και είναι αδύνατη όταν τα παιχνίδια διαδραματίζονται σε εσωτερικούς χώρους. Ο εντοπισμός θέσης βάσει του Χρόνου Άφιξης (Time of Arrival) υπερηχητικών σημάτων [37] είναι επίσης πρακτικά αδύνατος ιδιαίτερα όταν στα παιχνίδια εμπλέκεται μεγάλος αριθμός παικτών και εκτυλίσσονται σε θορυβώδη περιβάλλοντα .

Η Λύση Ο εντοπισμός θέσης της συσκευής του παίκτη πραγματοποιείται χρησιμοποιώντας τους Σταθμούς της κεντρική υποδομής και τα περιοδικά μηνύματα που λαμβάνουν από τις συσκευές των παικτών [38]. Στην περίπτωση αυτή οι σταθμοί ονομάζονται Anchor Station και για τη σωστή λειτουργία πρέπει να είναι τουλάχιστον τρεις. Για τα περιοδικά μηνύματα χρησιμοποιούνται τα beacon messages του Echo Protocol.

Κάθε φορά που οι Σταθμοί λαμβάνουν ένα μήνυμα εξάγονται δύο δείκτες σχετικοί με την ισχύ και την ποιότητα του σήματος:

1. Received Signal Stregth Indicator (RSSI)

2. Link Quality Indicator (LQI)

Οι δείκτες αυτοί προωθούνται διαρκώς σε ένα κεντρικό σημείο όπου και υπολογίζεται η θέση της συσκευής βάσει κάποιου Localization Αλγορίθμου.

Η Υλοποίηση Χρησιμοποιώντας τους δείκτες RSSI και LQI, μπορεί να εφαρμοστεί ένα πλήθος από κεντρικοποιημένους αλγορίθμους με σκοπό τον προσδιορισμό της θέσης ενός κόμβου.

Στους περισσότερους υλοποιημένους αλγορίθμους που ακολουθούν, απαιτείται να υπάρχει ένα σύνολο τιμών προς σύγκριση. Το σύνολο αυτό δημιουργείται πριν οι αλγόριθμοι εκτελεστούν και αποτελεί τις τιμές τις εκπαίδευσης τους συστήματος (trained values)

Simple Localization Κάθε Anchor Station αντιστοιχεί σε ένα σημείο στον δισδιάστατου χώρο. Η θέση της συσκευής ταυτίζεται με τη θέση του Station με την καλύτερη ληψη σήματος. Για την εφαρμογή αυτής της μεθόδου δεν απαιτείται εκπαίδευση του συστήματος.

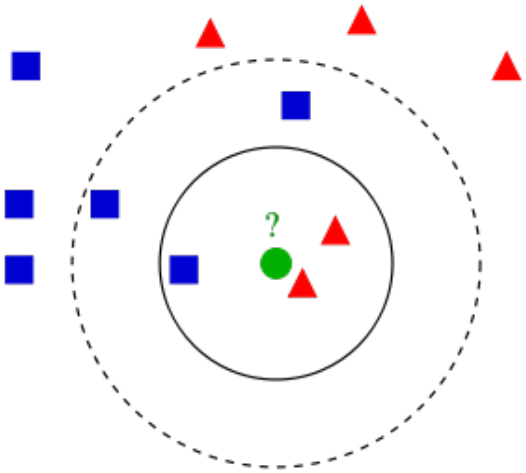
Average Localization Έστω υπάρχει μια εγκατάσταση από N Anchor Station και n προκαθορισμένες θέσεις στο χώρο όπου διεξάγεται το localization.

Εκμεταλλευόμενος τα μηνύματα του Echo Protocol ο αλγόριθμος συλλέγει τις RSSI και LQI μετρήσεις για το συγκεκριμένο μήνυμα για κάθε έναν από τους σταθμούς. Στη συνέχεια δημιουργείται ένα N-διάστατο διάνυσμα V οι συντεταγμένες του οποίου είναι οι εξαγόμενοι δείκτες. Η διαδικασία επαναλαμβάνεται για κάθε μία από τις n θέσεις. Κάθε φορά το διάνυσμα αυτό αποθηκεύεται και συσχετίζεται με την αντίστοιχη θέση στο φυσικό χώρο. Όλα τα διανύσματα αποτελούν την εκπαίδευση του συστήματος.

Στη συνέχεια, για κάθε ένα μήνυμα που λαμβάνεται ακολουθείται η ίδια διαδικασία και δημιουργείται ένα διάνυσμα V_c. Υπολογίζεται η ευκλείδεια απόσταση του V_c από όλα τα αποθηκευμένα διανύσματα V_i, με $i = 1, 2, ..., k$. Στο τέλος της διαδικασίας, εξάγεται ένα διάνυσμα V_min. Ως τρέχουσα θέση της συσκευής θεωρείται η φυσική θέση με την οποία το διάνυσμα V_min έχει συσχετιστεί.

Αλγόριθμος K κοντινότερων γειτόνων Η μέθοδος αυτή βασίζεται στην ιδέα του αλγορίθμου των k εγγύτερων γειτόνων (Σχήμα 4.17) για τον εντοπισμό της θέσης της συσκευής. Όπως και πριν, στη μέθοδο Average Localization, για κάθε μια από τις n φυσικές θέσεις δημιουργείται N-διάστατο διάνυσμα. Στην περίπτωση αυτή όμως, η διαδικασία επαναλαμβάνεται περισσότερες φορές –έστω m– για την ίδια θέση. Έτσι, σε κάθε μία φυσική θέση, αντιστοιχεί ένα σύνολο διανυσμάτων, ίδιου μεγέθους για κάθε θέση. Στη συνέχεια, υπολογίζεται και πάλι η

ευκλείδεια απόσταση μεταξύ του τρέχοντος διανύσματος V_c και όλων των άλλων διανυσμάτων V_ij με $i = 1, 2, ...n$ και $j = 1, 2, ...m$. Στην συνέχεια, δημιουργείται ένα άλλο σύνολο K μεγέθους k, το οποίο περιέχει τα k διανύσματα με την μικρότερη απόσταση από το V_c. Ως τρέχουσα θέση της συσκευής θεωρείται η φυσική θέση με την οποία είναι συσχετισμένος ο μεγαλύτερος αριθμός από διανύσματα στο σύνολο K.



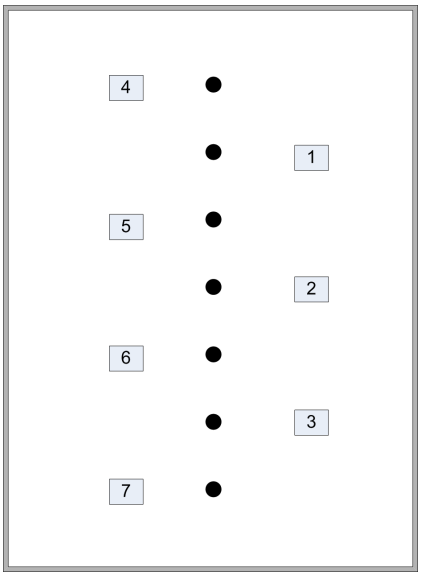
Σχήμα 4.17: Η μέθοδος classification K κοντινότερων γειτόνων. Ο κύκλος είναι πιο κοντά στη γειτονιά των τριγώνων καθώς γειτνιάζει με περισσότερα τρίγωνα σε ένα πρώτο επίπεδο. Αντίθετα, ανήκει στην γειτονιά των τετραγώνων σε δεύτερο επίπεδο.

Hybrid Localization Η εκτίμηση της θέσης των παικτών σε αυτή την περίπτωση πραγματοποιείται σε δύο βήματα. Εν πρώτοις, περιορίζεται ο χώρος που βρίσκεται η συσκευή σε μια συγκεκριμένη περιοχή (Region). Κάθε περιοχή περιλαμβάνει ένα σύνολο φυσικών σημείων ενώ όλες οι περιοχές έχουν οριστεί πριν την εκτέλεση του αλγορίθμου. Στο δεύτερο βήμα, γίνεται ένας πιο ακριβής εντοπισμός της θέσης της συσκευής μέσα στα όρια της περιοχής που έχει επιλεχθεί αρχικά.

Για τα δύο βήματα, χρησιμοποιείται κάποιος από τους υλοποιημένους αλγορίθμους. Για παράδειγμα, προκειμένου να γίνει μια γρήγορη και πρόχειρη εκτίμηση χρησιμοποιείται ο Simple Localization αλγόριθμος ενώ στη συνέχεια για τον ακριβέστερο προσδιορισμό εφαρμόζεται ο Average Localization.

Αξιολόγηση Για την πιστοποίηση της ορθής λειτουργίας του συστήματος εντοπισμού θέσης πραγματοποιήθηκαν δύο πειράματα σε δύο διαφορετικούς χώρους και με διαφορετική εγκατάσταση στον καθένα.

Πρώτο Πείραμα Κατά το πρώτο πείραμα 7 Anchor Station τοποθετούνται στην οροφή του δωματιού (περίπου 3 μέτρα ύψος) όλοι με την ίδια κατεύθυνση. Το πεδίο εκπαίδευσης αποτελείται από 7 προκαθορισμένες θέσεις grid, όλες τοποθετημένες σε μία γραμμή. Η απόσταση μεταξύ δύο σημείων είναι 0.8 μέτρα. Η κάτοψη της εγκατάστασης φαίνεται στο Σχήμα 4.18.



Σχήμα 4.18: Η εγκατάσταση του πρώτου πειράματος για τον εντοπισμό θέσης.

Σε κάθε θέση του grid τοποθετείται κάθε μια συσκευή Sun SPOT που εκπέμπει μηνύματα beacon με την ίδια κατεύθυνση. Κάθε Station καταγράφει το RSSI κάθε μηνύματος. Συνολικά, 100 beacons εκπέμπονται από κάθε θέση του grid. Η ισχύς εκπεμπόμενου σήματος έχει οριστεί στα -10dB καθώς πειραματικά έχει προκύψει ως ισχύς με τη μεγαλύτερη διαβάθμιση στο συγκεκριμένο χώρο.

Τα αποτελέσματα των μετρήσεων με τις μέσες τιμές τους φαίνονται στους πίνακες 4.3 μέχρι 4.9.

0	1	2	3	4	5	6
-48.8	-43.9	-47.2	-40.233334	-37.166668	-44.6	-49.066666

Πίνακας 4.3: Μέσες μετρήσεις για τον Σταθμό 0014.4F01.0000.47A4 (ID 1)

0	1	2	3	4	5	6
-40.4	-33.933334	-23.666666	-27.533333	-30.7	-32.1	-36.966667

Πίνακας 4.4: Μέσες μετρήσεις για τον Σταθμό 0014.4F01.0000.6164 (ID 2)

0	1	2	3	4	5	6
-50.0	-44.7	-42.466667	-35.366665	-30.433332	-37.5	-38.9

Πίνακας 4.5: Μέσες μετρήσεις για τον Σταθμό 0014.4F01.0000.48FB (ID 3)

0	1	2	3	4	5	6
-32.433334	-36.1	-22.1	-26.466667	-31.3	-34.733334	-34.866665

Πίνακας 4.6: Μέσες μετρήσεις για τον Σταθμό 0014.4F01.0000.73E6 (ID 4)

0	1	2	3	4	5	6
-40.4	-38.966667	-32.066666	-31.133333	-37.066666	-35.7	-40.4

Πίνακας 4.7: Μέσες μετρήσεις για τον Σταθμό 0014.4F01.0000.4C5D (ID 5)

0	1	2	3	4	5	6
-40.933334	-49.266666	-44.466667	-31.1	-35.266666	-31.633333	-29.033333

Πίνακας 4.8: Μέσες μετρήσεις για τον Σταθμό 0014.4F01.0000.739A (ID 6)

0	1	2	3	4	5	6
-39.766666	-38.933334	-38.0	-33.966667	-31.233334	-33.033333	-30.7

Πίνακας 4.9: Μέσες μετρήσεις για τον Σταθμό 0014.4F01.0000.46A8 (ID 7)

Δεύτερο Πείραμα Στο δεύτερο πείραμα εκτελέστηκε η ίδια διαδικασία με αυτή του πρώτου αλλάζοντας την εγκατάσταση και τον χώρο διεξαγωγής.

Τα σημεία του grid είναι τώρα 10 και βρίσκονται πάλι σε μια ευθεία γραμμή και σε απόσταση 1.20 μέτρων. Η κάτοψη της εγκατάστασης φαίνεται στο Σχήμα 4.19.

Σε κάθε θέση του grid τοποθετείται κάθε μια συσκευή Sun SPOT που εκπέμπει μηνύματα beacon με την ίδια κατεύθυνση. Κάθε Station καταγράφει το RSSI κάθε μηνύματος. Συνολικά, 100 beacons εκπέμπονται από κάθε θέση του grid. Η ισχύς εκπεμπόμενου σήματος έχει οριστεί στα 10dB καθώς πρόκειται για μεγαλύτερο χώρο.

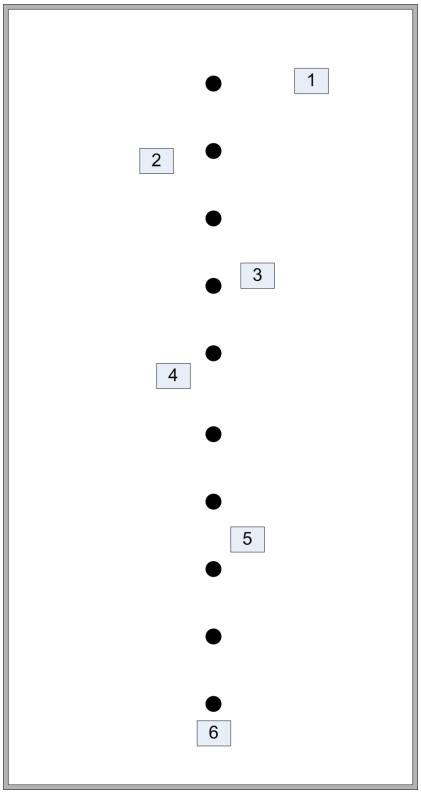
Τα αποτελέσματα των μετρήσεων με τις μέσες τιμές τους φαίνονται στους πίνακες 4.10 μέχρι 4.15.

0	1	2	3	4	5	6	7	8	9
-50.0	-49.566666	-49.566666	-40.866665	-42.8	-31.466667	-32.933334	-30.2	-31.533333	-38.5

Πίνακας 4.10: Μέσες μετρήσεις για τον Σταθμό 0014.4F01.0000.6164 (ID 1)

0	1	2	3	4	5	6	7	8	9
-50.0	-50.0	-50.0	-50.0	-46.1	-50.0	-49.5	-40.866665	-43.366665	-44.833332

Πίνακας 4.11: Μέσες μετρήσεις για τον Σταθμό 0014.4F01.0000.47A4 (ID 2)



Σχήμα 4.19: Η εγκατάσταση του πρώτου πειράματος για τον εντοπισμό θέσης.

0	1	2	3	4	5	6	7	8	9
-50.0	-50.0	-44.6	-43.066666	-40.766666	-44.733334	-44.866665	-46.666668	-50.0	-50.0

Πίνακας 4.12: Μέσες μετρήσεις για τον Σταθμό 0014.4F01.0000.48FB (ID 3)

0	1	2	3	4	5	6	7	8	9
-49.4	-41.566666	-46.933334	-48.7	-50.0	-50.0	-50.0	-50.0	-50.0	-50.0

Πίνακας 4.13: Μέσες μετρήσεις για τον Σταθμό 0014.4F01.0000.73E6 (ID 4)

0	1	2	3	4	5	6	7	8	9
-42.6	-45.766666	-36.8	-41.766666	-47.133335	-49.3	-49.633335	-50.0	-49.866665	-50.0

Πίνακας 4.14: Μέσες μετρήσεις για τον Σταθμό 0014.4F01.0000.4C5D (ID 5)

0	1	2	3	4	5	6	7	8	9
-43.233334	-45.533333	-35.7	-35.7	-40.6	-47.133335	-50.0	-49.866665	-50.0	-50.0

Πίνακας 4.15: Μέσες μετρήσεις για τον Σταθμό 0014.4F01.0000.5444 (ID 6)

Με την εκτέλεση των πειραμάτων προκύπτει πώς οι μετρήσεις RSSI και LQI επηρεάζονται και από άλλους πέρα από την απόσταση μεταξύ του αποστολέα

και του παραλήπτη. Οι μετρήσεις αλλάζουν σε πολύ μεγάλο βαθμό όταν το περιβάλλον εκτέλεσης αλλοίωνεται (παρουσία εμποδίων, ανθρώπων κ.τ.λ) ή η κατεύθυνση της συσκευής μεταβάλλεται.

Κεφάλαιο 5

Συμπεράσματα - Στόχοι

Στη διπλωματική εργασία αυτή παρουσιάζεται η πλατφόρμα Fun In Numbers, μια πλατφόρμα για τη δημιουργία, τον έλεγχο και την εμπειρία χρήσης Παιχνιδιών Περιρρέουσας Υπολογιστικής Νοημοσύνης με χρήση Ασύρματων Δικτύων Αισθητήρων. Καθορίστηκε ένα σύνολο αρχών στα οποία τέτοιου είδους συστήματα χρειάζεται να υπακούν. Παρουσιάστηκε η αρχιτεκτονική της πλατφόρμας και οι βασικοί μηχανισμοί που παρέχουν ενοποιημένες λύσεις στα προβλήματα και τα θέματα που παρουσιάζονται. Μέσω μιας σειράς πειραμάτων και μετρήσεων αξιολογήθηκε η απόδοση των μηχανισμών αυτών, προέκυψαν συμπεράσματα σχετικά με τον αριθμό των συμμετεχόντων υπό ορισμένες συνθήκες και εξετάστηκε ο χρόνος απόκρισης των επιμέρους υποσυστημάτων. Σημαντικό επίσης είναι το γεγονός ότι πιστοποιήθηκε στην πράξη η συνολική ορθή λειτουργία του συστήματος μέσα από έναν αριθμό υλοποιημένων παιχνιδιών.

Όσον αφορά τους μελλοντικούς στόχους, αυτοί αφορούν τον πειραματισμό πάνω σε νέες πλατφόρμες, ευρύτερα χρησιμοποιούμενες όπως τα κινητά τηλέφωνα με σκοπό τον εμπλουτισμό της διεπαφής του χρήστη και την αξιοποίηση περισσότερων τεχνολογιών. Στόχο αποτελεί η αξιοποίηση των δυνατοτήτων τέτοιων συσκευών στο άμεσο μέλλον, καθώς σε ένα βαθμό παρέχουν δυνατότητες διάδρασης συμβατές με την πλατφόρμα FinN. Η τεχνολογία των επιφανειών πολλαπλής αφής εμφανίζεται επίσης σε προϊόντα όπως το Microsoft Surface, που είναι επί της ουσίας ένας ηλεκτρονικός υπολογιστής σε σχήμα τραπεζιού όπου το πάνω μέρος είναι μια οθόνη πολλαπλής αφής. Υπάρχουν και άλλες εταιρίες με παρόμοια προϊόντα όπως η Multitouch (http://multitouch.fi/), η Touchtable (http://www.touchtable.com) και η Reactable (http://www.reactable.com/), ενώ κάποιες από αυτές προσφέρουν και το λογισμικό για την ανάπτυξη εφαρμογών για τα προϊόντα τους.

Βιβλιογραφία

[1] Mark Weiser Ubiquitous Computing Website, http://www.ubiq.com/hypertext/weiser/UbiHome.html

[2] Ubiquitous Computing: An Interesting New Paradigm by Marcia Riley, http://www.cc.gatech.edu/classes/cs6751_97_fall/projects/say-cheese/marcia/mfinal.html

[3] Integrated Project on Pervasive Gaming, http://iperg.sics.se/index.php

[4] A Survey of Augmented Reality [Quick Edit] Presence, Vol. 6 (1997), pp. 355-385. by Ronald T. Azuma

[5] CatchBob! http://craftwww.epfl.ch/research/catchbob/

[6] Uncle Roy All Around You: Implicating the City in a Location-Based Performance by Steve, Benford and Martin, Flintham and Adam, Drozd and Rob, Anastasi and Duncan, Rowland and Nick, Tandavanitj and Matt, Adams and Ju, Row-Farr and Amanda, Oldroyd and Jon, Sutton

[7] NETATTACK http://radio-weblogs.com/0105910/2004/04/29.html

[8] Human Pacman: a sensing-based mobile entertainment system with ubiquitous computing and tangible interaction by Cheok, Adrian David and Fong, Siew Wan and Goh, Kok Hwee and Yang, Xubo and Liu, Wei and Farzbiz, Farzam. Proceedings of the 2nd workshop on Network and system support for games

[9] The Scatterweb MSB-A2 Platform for Wireless Sensor Network, http://cst.mi.fu-berlin.de/projects/ScatterWeb/mod_MSB-A2.html

[10] Wireless Sensor Network Devices on CSIRO, http://www.ict.csiro.au/page.php?cid=87

[11] Tmote sky datasheet, http://www.bandwavetech.com/download/tmote-sky-datasheet.pdf

[12] MICA2 datasheet, https://www.eol.ucar.edu/rtf/facilities/isa/internal/CrossBow/DataSheets/mica2.pdf

[13] MICAZ datasheet, http://www.openautomation.net/uploadsproductos/micaz_datasheet.pdf

[14] TelosB datasheet, http://www.willow.co.uk/TelosB_Datasheet.pdf

[15] iSense website, http://www.coalesenses.com/

[16] Sun SPOT World website, http://sunspotworld.com/

[17] Environmental Sensor Networks: A revolution in the earth system science? by Jane K. Hart , Kirk Martinez , http://eprints.ecs.soton.ac.uk/13093/1/esn.pdf

[18] Energy-efficient wireless sensor network design and implementation for condition-based maintenance by Ankit Tiwari, Prasanna Ballal and Frank L. Lewis. ACM Transactions on Sensor Networks, Volume 3 Issue 1, March 2007

[19] Ad hoc On-Demand Distance Vector Routing by Charles E. Perkins and Elizabeth M. Royer. Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, New Orleans, LA, February 1999, pp. 90-100.

[20] Highly Dynamic Destination-Sequenced Distance Vector (DSDV) for Mobile Computers by C. E. PERKINS, P. BHAGWAT, Proc. of the SIGCOMM 1994 Conference on Communications Architectures, Protocols and Applications, Aug 1994, pp 234-244.

[21] Real Time Wireless Sensor Network System for Environmental Monitoring: The Greenhouse Case Study by Panagiotis Kikiras, Ioannis Kalavros, Theofilos Nikolaou and Leonidas Perlepes University of Thessaly. 1st Student Workshop on Wireless Sensor Networks, November 2008 Athens, Greece

[22] Sensorium games: usability considerations for pervasive gaming by Mount, S. N. I. and Gaura, E. I. and Newman, R. M.. Proceedings of the 23rd annual international conference on Design of communication: documenting & designing for pervasive information

[23] Wireless sensor network based mobile pet game by Liu„ Liang and Ma„ Huadong. NetGames '06: Proceedings of 5th ACM SIGCOMM workshop on Network and system support for games

[24] Alix system boards website, http://www.pcengines.ch/alix.htm

[25] Distributed algorithms by Nancy Ann Lynch. Page 184, Section 7.3.2

[26] Wii website, http://wii.com/

[27] Kinect website, http://www.xbox.com/en-US/kinect

[28] Towards a Better Understanding of Context and Context-Awareness by Anind K. Dey and Gregory D. Abowd

[29] Personal area networks by T. Zahariadis. Commun. Eng. – June 2003 – Volume 1, Issue 3, p.12–15

[30] A Delay-Tolerant Network Architecture for Challenged Internets by Kevin Fall SIGCOMM, August 2003.

[31] Chipcon CC2420 datasheet, http://inst.eecs.berkeley.edu/ cs150/Documents/CC2420.pdf

[32] IEEE 802.15 WPAN Task Group, http://www.ieee802.org/15/pub/TG1a.html

[33] Squawk Java ME VM on java.net, https://squawk.dev.java.net/

[34] A Java Virtual Machine Architecture for Very Small Devices by Nik Shaylor, Douglas N. Simon and William R. Bush. http://www.grundu.net/papers/squawk.pdf

[35] Impact of radio irregularity on wireless sensor networks,by Gang Zhou, Tian He, Sudha Krishnamurthy and John A. Stankovic. MobiSys '04 Proceedings of the 2nd international conference on Mobile systems, applications, and services

[36] The Observer Design pattern on IBS Research, http://www.research.ibm.com/designpatterns/example.htm

[37] The Cricket Location-Support System by Nissanka B. Priyantha, Anit Chakraborty, and Hari Balakrishnan. Proc. of the Sixth Annual ACM International Conference on Mobile Computing and Networking (MOBICOM), August 2000.

[38] Indoor location and orientation determination for wireless personal area networks by Zekeng Liang, Ioannis Barakos, Stefan Poslad. MELT'09 Proceedings of the 2nd international conference on Mobile entity localization and tracking in GPS-less environments

[39] Hibernate website, http://www.hibernate.org/

[40] What is Multi-touch, http://solutions.3m.com/wps/portal/3M/en_US/TouchTopics/Home /Terminology/WhatIsMultitouch/