ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΔΙΑΤΜΗΜΑΤΙΚΟ ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ ΣΤΗ ΔΙΟΙΚΗΣΗ ΚΑΙ ΟΙΚΟΝΟΜΙΚΗ ΤΩΝ ΤΗΛΕΠΙΚΟΙΝΩΝΙΑΚΩΝ ΔΙΚΤΥΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Real-time επεξεργασία για Big Data Location-based Services και υλοποίηση στο Apache Storm

Φώτης Β. Θάνος
Θεόδωρος Ε. Σάββας

Επιβλέπων: **Ευστάθιος Χατζηευθυμιάδης,** Αναπληρωτής Καθηγητής

ΑΘΗΝΑ

ΦΕΒΡΟΥΑΡΙΟΣ 2017

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Real-time επεξεργασία δεδομένων για Big Data Location-based services και υλοποιήση στο Apache Storm

Φώτης Β. Θάνος
Α.Μ.: ΜΟΠ430

Θεόδωρος Ε. Σάββας
Α.Μ.: ΜΟΠ436

ΕΠΙΒΛΕΠΩΝ: **Ευστάθιος Χατζηευθυμιάδης,** Αναπληρωτής Καθηγητής

ΕΞΕΤΑΣΤΙΚΗ ΕΠΙΤΡΟΠΗ: **Ευστάθιος Χατζηευθυμιάδης,** Αναπληρωτής Καθηγητής
Δημήτριος Βαρούτας, Επίκουρος Καθηγητής

Φεβρουάριος 2017

ΠΕΡΙΛΗΨΗ

Η διπλωματική εργασία αυτή εκπονήθηκε με σκοπό την προσέγγιση και την υλοποίηση μιας αναλυτικής λογικής για την real time επεξεργασία δεδομένων σε indoor (και όχι μόνο) δίκτυα αισθητήρων. Αρχικά ορίζονται και αναλύονται οι αρχές και τα βασικότερα δομικά στοιχεία των υπηρεσιών τοποθεσίας (location-based services). Συγκεκριμένα αναλύονται οι επικρατέστερες τεχνολογίες στον κλάδο και τα θεμέλια στοιχεία τους, δηλαδή το middleware, τα network components καθώς και οι επιμέρους παράγοντες. Γίνεται αναφορά στα στοιχεία και στα κριτήρια πάνω στα οποία θα υλοποιηθεί έπειτα ο κώδικας επεξεργασίας στο ορισμένο περιβάλλον λειτουργίας (Apache Storm). Έπειτα αναλύεται το stream processing ως βασικό στοιχείο για την παράλληλη επεξεργασία μιας συγκεκριμένης μορφής πληροφορίας, η οποία δεν απαιτεί περαιτέρω συμφωνία και παράλληλες ρυθμίσεις από τους κόμβους τους οποίους συνδέει. Αναφέρονται τα χαρακτηριστά που πρέπει να πληρούνται από τις εφαρμογές ώστε να μπορούν να χρησιμοποιούν κατάλληλα stream processing. Εν συνεχεία αναλύονται οι contextors ως βασικός μηχανισμός αφαιρετικής διανομής της πληροφορίας. Συγκεκριμενοποιώντας το stream processing στην εν λόγω διπλωματική γίνεται αναφορά και στα real time processing συστήματα ως προς τις stream processing λογικές που χρησιμοποιούν. Τέλος, γίνεται αναλυτική αναφορά στον κώδικα υλοποίησης της business logic για indoor positioning συστήματα που υλοποιήθηκε στο Apache Storm και σε γλώσσα προγραμματισμού Java.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Indoor Positioning System with Apache Storm

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: location-based services, real-time processing, stream processing, Apache Storm, indoor analytics and metrics, Υπηρεσίες τοποθεσίας

ABSTRACT

This master thesis was developed in order to approach and implement a detailed logic for real-time data processing in indoor (and not only) sensor networks. The principles and the basic building blocks of location-based services are defined and analyzed. Specifically, we analyze the prevalent technologies in the industry and the foundation elements, namely the middleware (and its construction principles), the network components and othere individual factors. A detailed reference is been made to the criteria on which we will then produce the source code in the specific operating environment we chose to work (Apache Storm). Then we proceed to the analysis of the stream processing as a key element for the parallel processing of a particular kind of information, which requires no further agreement (on software and configurations) or parallel arrangents of the nodes which connects. The characteristics that need to be fulfilled by the applications are reported as well in order to give a clear image of appropriate stream processing implementation. Then we analyze contextors as a key mechanism of deductive distribution of information. Furthermore, we specify the stream processing based systems in comparison with real-time processing systems. Finally, there is a detailed report of the implementation of business logic and of the source code itself. The system of implementation was Apache Storm and Java programming language.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Indoor Positioning System with Apache Storm

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: location-based services, real-time processing, stream processing, Apache Storm, indoor analytics and metrics

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

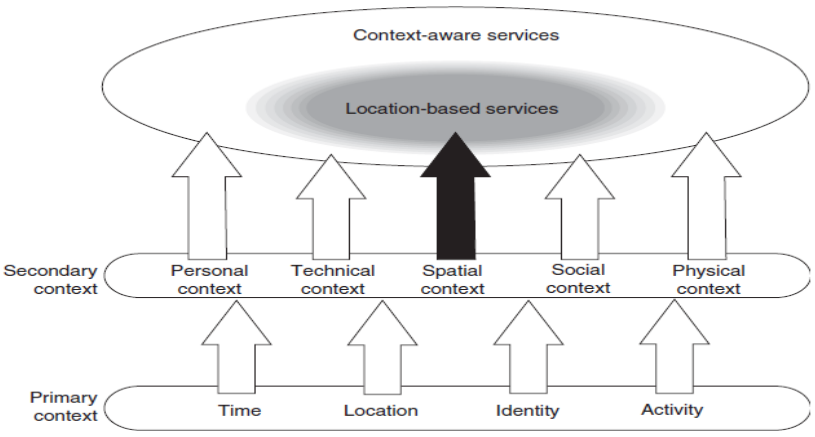
ΠΡΟΛΟΓΟΣ

Η εργασία αυτή υλοποιήθηκε τον Μάρτη έως τον Νοέμβριο του 2016 στην Αθήνα υπό την καθοδήγηση του αναπληρωτή καθηγητή Στάθη Χατζηευθυμιάδη. Η ομαδά εργασίας (Φώτης Θάνος και Θοδωρής Σάββας) διεκπεραίωσε το τελικό κείμενο καθώς και τον κώδικα της εργασίας στα πλαίσια της διπλωματικής τους εργασίας για το διατμηματικό μεταπτυχιακό πρόγραμμα «Οικονομική και Διοικητική των Τηλεπικοινωνιακών Δικτύων».

1. Location-based Services

1.1 Ο ορισμός της τοποθεσίας και των αντίστοιχων υπηρεσιών. Οι LBS τα είδη τους.

Ο όρος τοποθεσία συναρτάται με ένα συγκεκριμένο χωροθετημένο μέρος του πραγματικού κόσμου. Η ορολογία Location-based Services (εφεξής LBS) αναφέρεται σε υπηρεσίες που αφορούν φυσικές τοποθεσίες ή περιοχές. Στην πανεπιστημιακή έρευνα, οι LBS θεωρούνται συχνά ως ένα ειδικό υποσύνολο των λεγόμενων context-aware services (οι οποίες θεωρούνται η βασική πηγή του όρου location-aware services, εξέλιξη των οποίων αποτελούν και οι LBS). Γενικά οι context-aware υπηρεσίες ορίζονται ως οι υπηρεσίες εκείνες που προσαρμόζουν αυτόματα την συμπεριφορά τους, παραδείγματος χάριν φιλτράροντας ή παρουσιάζοντας ειδικές πληροφορίες, σε μια ή περισσότερες παραμέτρους αντανακλώντας το πλαίσιο (context) ενός γενικού στόχου. Το σύνολο των πληροφοριών περιεχομένου (context) ευρέως κατηγοριοποιείται και υποδιαιρείται σε προσωπικά, τεχνικά, χωρικά, κοινωνικά και φυσικά πλαίσια (context), όπως φαίνεται και στην εικόνα 1. .Μπορεί βέβαια να υποδιαιρεθεί περαιτέρω σε πρωταρχικά και δευτερεύοντα πλαίσια. Τα πρωταρχικά πλαίσια αποτελούνται από κάθε είδους ακατέργαστα δεδομένα που μπορούν να συλλεχθούν από sensors (αισθητήρες) – όπως είναι οι αισθητήρες φωτός, οι βιοαισθητήρες, τα μικρόφωνα, οι μετρητές επιτάχυνσης (αξελερόμετρα), οι αισθητήρες θέσης κ.ο.κ. Αυτά τα ακατέργαστα δεδομένα μπορούν να βελτιωθούν (ως προς την ποιότητα της παρεχόμενης πληροφορίας) με την χρήση συνδυασμού δεδομένων, αφαίρεση δεδομένων ή φιλτράρισμα ώστε να παραχθεί υψηλού επιπέδου context information, η οποία ορίζεται ως secondary context και είναι πιο κατάλληλη για επεξεργασία από τις υπηρεσίες. Στην συγκεκριμένη υλοποίηση της εργασίας μας τα δεδομένα προέρχονται από αισθητήρες θέσης και η εξόρυξη των secondary context πληροφοριών γίνεται εντός της αρχιτεκτονικής του Apache Storm (με την χρήση spout και bolt).



Εικόνα 1: Φαίνεται η σχέση των LBS με τις context-aware services καθώς και η υποδιαίρεση τους σε πρωταρχικό και δευτερεύον context level.

Οι τρεις βασικές κατηγορίες που διακρίνουμε στις Location-based Services είναι οι descriptive locations (περιγραφικές τοποθεσίες), οι χωρικές τοποθεσίες (spatial locations) και οι δικτυακές τοποθεσίες (network locations). Η 1η κατηγορία (περιγραφικές τοποθεσίες) αναφέρεται σε τοποθεσίες που σχετίζονται άμεσα με φυσικά γεωγραφικά αντικείμενα όπως περιοχές, βουνά ή λίμνες και με γεωγραφικά αντικείμενα που έχει δημιουργήσει ο άνθρωπος όπως είναι οι πόλεις, τα σύνορα, οι δρόμοι, τα δωμάτια εντός ενός κτηρίου κ.α. Οι γεωγραφικές αυτές δομές ορίζονται από περιγραφές (ονόματα, αριθμούς, αναγνωριστικά σημεία κ.α.) Η 3η κατηγορία (δικτυακές τοποθεσίες) αναφέρεται σε τοπολογίες δικτύων επικοινωνίας όπως είναι παραδείγματος χάριν το Internet ή τα κυψελικά τηλεπικοινωνιακά δίκτυα UMTS ή τα 4G δίκτυα. Αυτά τα δίκτυα αποτελούν το αποτέλεσμα σύνθεσης πολλών τοπικών δικτύων (υποδίκτυα) τα οποία είναι συνδεδεμένα μεταξύ τους με μια συγκεκριμένη ιεραρχημένη τοπολογία κυκλωμάτων κορμού (trunk circuits) και backbone δικτύων. Η παροχή υπηρεσιών σε αυτά τα δίκτυα προϋποθέτει ότι η θέση της συσκευής του χρήστη σε σχέση με την τοπολογία του δικτύου είναι γνωστή. Αυτό επιτυγχάνεται με τις διευθύνσεις του δικτύου (network addresses) σε συνδυασμό με τις υπηρεσίες καταλόγου (directory services), οι οποίες υλοποιούν την αντιστοίχηση αριθμών, αναγνωριστικών και ονομάτων ενός συγκεκριμένου συστήματος πάνω στην διεύθυνση του δικτύου.

Όσον αφορά την 2η κατηγορία (χωρικές τοποθεσίες), με την κυριολεκτική έννοια, μια χωρική τοποθεσία αντιπροσωπεύει ένα μοναδικό σημείο στον Ευκλείδειο χώρο. Ένας πιο «διαισθητικός» όρος για την περιγραφή της χωρικής τοποθεσίας είναι ο όρος θέση. Συνήθως εκφράζεται μέσω συντεταγμένων 2 ή 3ων διαστάσεων οι οποίες δίνονται ως διανύσματα αριθμών, το καθένα από τα οποία ορίζουν την θέση σε μια διάσταση. Οι χωρικές τοποθεσίες χρησιμοποιούνται για επιστημονικές ή επαγγελματικές εφαρμογές, όπως π.χ. είναι οι αεροπορικές ή ναυτιλιακές, οι οποίες απαιτούν εξαιρετικά ακριβείς πληροφορίες θέσης. Η έννοια των χωρικών θέσεων είναι συμπληρωματική και για την αποτύπωση και για την χαρτογράφηση των περιγραφικών τοποθεσιών. Η παρούσα εργασία εντάσσεται σε αυτήν την υποκατηγορία υλοποιώντας παράλληλα την προαναφερθείσα λειτουργικότητα (χαρτογράφηση περιγραφικών τοποθεσιών).

Μια ακόμη διάκριση μεταξύ των LBSs είναι η και ταξινόμησή τους μεταξύ reactive (διαδραστικές) και proactive (δυναμικές). Οι reactive LBSs είναι πάντα ενεργοποιημένες από τον χρήστη και η διάδραση του, κατά βάση, γίνεται ως εξής: ο χρήστης επικαλείται για πρώτη φορά το service και ανοίγει ένα session για την εν λόγω υπηρεσία (είτε μέσω μια κινητής συσκευής είτε μέσω ενός υπολογιστικού συστήματος). Έπειτα κάνει αίτηση για συγκεκριμένες συναρτήσεις ή πληροφορίες, τότε το service παίρνει δεδομένα για την τοποθεσία (είτε του χρήστη είτε κάποιου άλλου στόχου), τα επεξεργάζεται και επιστρέφει τα αποτελέσματα (που αναφέρονται στην τοποθεσία για την οποία έγινε αίτηση) στον χρήστη π.χ. μια λίστα κοντινών σε εκείνον ξενοδοχείων. Αυτό ο κύκλος request – response μπορεί να επαναληφθεί πολλές φορές πριν τον τερματισμό του session. Έτσι μια reactive LBS χαρακτηρίζεται από ένα πρότυπο σύγχρονης διάδρασης μεταξύ του χρήστη και της υπηρεσίας. Μια proactive LBS, από την άλλη μεριά, αρχικοποιείται αυτόματα όταν συμβεί ένα προκαθορισμένο γεγονός σε μια ορισμένη τοποθεσία, αν παραδείγματος χάριν ο χρήστης περάσει από ένα σημείο ενδιαφέροντος και πρέπει να ειδοποιείται μέσω SMS για τις ιστορικές λεπτομέρειες του σημείου ενδιαφέροντος. Έτσι, γίνεται κατανοητό ότι μια proactive LBS δεν ενεργοποιείται ρητά από τον χρήστη αλλά η επικοινωνία μεταξύ τους γίνεται ασύγχρονα – σε αντίθεση με τις reactive LBSs, όπου ο

χρήστης εντοπίζεται μια φορά στο σύστημα, στις proactive LBSs απαιτείται η διαρκής καταγραφή του ώστε να εντοπίζονται τα σημεία ενδιαφέροντος ή τα συμβάντα συγκεκριμένων τοποθεσιών.

1.2 Σενάρια και περιβάλλοντα λειτουργίας των LBSs

Για να αποσαφηνιστεί το περιεχόμενο και η λειτουργικότητα των LBSs καλύτερα, θα παραθέσουμε κάποια παραδείγματα εφαρμογής των LBSs.

Όσον αφορά τα LBSs τα οποία υλοποιούνται από εταιρικά/business περιβάλλοντα, ο στόχος τους είναι να αυξήσουν την κερδοφορία που παρέχουν στην εταιρεία, αυξάνοντας τον μέσο όρο του χρόνου σύνδεσης (airtime) ανά χρήστη, πουλώντας δεδομένα ή πληροφορίες τοποθεσίας σε τρίτους και προσφέροντας services προσαρμοσμένα στις ειδικές ανάγκες των χρηστών κινητής τηλεφωνίας. Ο πάροχος του service μπορεί είτε να παρέχει μόνος του την υπηρεσία είτε να συνάψει επιχειρηματικές σχέσεις με άλλους συντελεστές της αγοράς από τον κλάδο του εμπορίου και της διαφήμισης ή της αυτοκινητοβιομηχανίας και να παρέχει υπηρεσίες για αυτούς.

Ο απλούστερος και πιο διαδεδομένος τύπος εφαρμογών LBSs είναι οι υπηρεσίες enquiry and information που παρέχουν στον χρήστη κινητών συσκευών (κατά βάση) πληροφορίες για τα κοντινά σημεία ενδιαφέροντος. Κατόπιν αιτήματος ο χρήστης εντοπίζεται είτε αυτόματα από το δίκτυο κινητής τηλεφωνίας είτε, ενδεχομένως, με την χρήση της κατάλληλης τεχνολογίας εντοπισμού θέσης. Επιπλέον πρέπει να ορίσει επακριβώς τα σημεία ενδιαφέροντος (π.χ. πρατήρια υγρών καυσίμων ή εστιατόρια) καθώς και την επιθυμητική μέγιστη απόσταση που είναι διατεθειμένος να διασχίσει για να φτάσει στα σημεία ενδιαφέροντος. Το αίτημα του έπειτα μεταφέρεται σε ένα παροχέα υπηρεσιών (service provider), ο οποίος συγκεντρώνει ένα κατάλογο με τα κατάλληλα σημεία ενδιαφέροντος και τα επιστρέφει στον χρήστη. Στα σημερινά δίκτυα, οι υπηρεσίες αυτές είναι προσβάσιμες κατά βάση μέσω 3G ή 4G.

Ένας άλλος κλάδος που βρίσκουν εφαρμογή οι LBSs είναι στην παροχή traffic telematics (τηλεματική παροχή πληροφοριών οδικής μεταφοράς). Το πεδίο έρευνας αυτό ασχολείται με την υποστήριξη των οδηγών αυτοκινήτων με ένα σύνολο πολλαπλών υπηρεσιών που σχετίζονται με τα οχήματα τους. Περιλαμβάνει πληροφορίες πλοήγησης, αυτόματες ρυθμίσεις παραμέτρων των συσκευών καθώς και εσωτερική διάγνωση σφαλμάτων στο όχημα και διάδοση προειδοποιητικών μηνυμάτων. Ένα σημαντικό ζήτημα σε αυτό το πεδίο έρευνας είναι και η ασύρματη δια-επικοινωνία μεταξύ των οχημάτων που βασίζεται σε short-range telecommunicational protocols όπως είναι το WLAN ή το Bluetooth και περιλαμβάνει την αποστολή προειδοποιητικών μηνυμάτων, την αποστολή των τοπικών συνθηκών κυκλοφορίας ή τον ad hoc εντοπισμό σταθμών υγρών καυσίμων. Το περιεχόμενο των μηνυμάτων προέρχεται από διαφορετικές πηγές αλλά κατά βάση από την τεχνολογία αισθητήρων εντός των οχημάτων. Τα δεδομένα που παραδίδονται από αυτούς τους αισθητήρες χαρακτηρίζονται ως «floating car data» και αποτελούνται από παραμέτρους όπως η ταχύτητα του αυτοκινήτου, η κατεύθυνση και η θέση του. Για την άντληση πληροφοριών υψηλού επιπέδου, όπως για παράδειγμα οι τοπικές συνθήκες κυκλοφορίας, οι floating car data πρέπει να συνδυαστούν με τα data που λαμβάνονται από άλλα

οχήματα πριν να διαδοθούν σε κοντινά οχήματα. Η διεπικοινωνία μεταξύ των οχημάτων είναι ένα σύνθετο ζήτημα το οποίο απαιτεί ισχυρά και αξιόπιστα συστήματα, μηχανισμούς ασφάλειας, πρωτόκολλα δρομολόγησης και τεχνολογίες εντοπισμού θέσης.

Ενώ ο κλάδος των traffic telematics ασχολείται μόνο με την στήριξη, αυτόνομων οχημάτων ο κλάδος fleet management (διαχείριση στόλου) ασχολείται με τον έλεγχο και τον συντονισμό ενός συνόλου (στόλου) οχημάτων μέσω ενός central office (κεντρικής διεύθυνσης). Τυπικές ομάδες εφαρμογής αυτών των LBSs είναι υπηρεσίες εμπορευματικών μεταφορών, τα μέσα μαζικής μεταφοράς καθώς και υπηρεσίες έκτακτης ανάγκης Error! Reference source not found.**. Τα LBS συστήματα που ασχολούνται με fleet management έχουν την δυνατότητα να ζητούν την θέση των οχημάτων, την αναπαράσταση της στον χάρτη, τον προσδιορισμό της απόστασης μεταξύ διαφορετικών οχημάτων του στόλου καθώς και μεταξύ ενός οχήματος και του προορισμού του. Με βάσης τις προαναφερθείσες πληροφορίες, το central office μπορεί να αναθέσει δυναμικά νέες παραγγελίες και να προβλέψει τον χρόνο παράδοσης των παραγγελιών στον προορισμό τους. Αυτά τα συστήματα χρησιμοποιούνται για την υποστήριξη κάθε μορφής logistics και βελτιστοποιούν αυτές τις διαδικασίες ως προς την ταχύτητα, τους τρόπους μεταφοράς αλλά και την ανάπτυξη εναλλακτικών σεναρίων σε περίπτωση αποτυχιών.**

1.3 Indoor positioning υλοποιήσεις LBS συστημάτων

Παρότι όπως αναφέραμε και στην αρχή αυτού του κεφαλαίου η πλειονότητα των LBSs έχουν σχεδιαστεί για την υποστήριξη εφαρμογών εξωτερικού χώρου, οι δυνατότητες των εφαρμογών που σχετίζονται με εσωτερικούς χώρους (indoor positioning applications) είχαν διερευνηθεί ήδη από τις αρχές της δεκαετίας του '90. Κάποια τυπικά πεδία εφαρμογής ήταν και εξακολουθούν να είναι τα περιβάλλοντα γραφείων και τα εμπορικά κέντρα.

Παράλληλα, παρά την παγιωμένη διαφοροποίηση στο πεδίο της έρευνας και της ανάπτυξης των εσωτερικών από τα εξωτερικά συστήματα LBS, υπάρχει αυξημένη ζήτηση για μια ενοποιημένη προσέγγιση. Από την πλευρά των χρηστών, ζητούμενο είναι τα request προς outdoor και indoor LBSs να μπορούν να γίνονται από τις ίδιες κινητές συσκευές, ενώ οι πάροχοι υπηρεσιών θα ήθελαν μια ενοποιημένη προσέγγιση για την χρήση κοινών infrastructures, interfaces και πρωτοκόλλων. Το μεγαλύτερο πρόβλημα μιας ενοποιημένης προσέγγισης είναι η απουσία μιας κοινής, καθολικής τεχνολογίας εντοπισμού θέσης. Οι συμβατικοί GPS δέκτες δεν λειτουργούν αποδοτικά στο εσωτερικό κτιρίων, ενώ οι cellular μέθοδοι προσδιορισμού θέσης γενικά αποτυγχάνουν στο να παρέχουν ένα ικανοποιητικό επίπεδο ακρίβειας (τουλάχιστον στα 3G δίκτυα). Οι παραδοθείσες πληροφορίες για την θέση δεν μπορούν να χρησιμοποιηθούν για να αποφασίσουμε αν ένα συγκεκριμένο άτομο βρίσκεται εντός ή εκτός ενός κτιρίου ή για να εντοπίσουμε την διασπορά των δωματίων ή των ορόφων ενός κτιρίου. Έτσι, ελλείψει αξιόπιστων ολοκληρωμένων λύσεων, υπάρχει ζήτηση για stand-alone indoor λύσεις στις οποίες θα αναφερθούμε στην συνέχεια [1].

1.4 WLAN positioning

Τα Wireless Local Area Networks (WLANs) ορίζονται από τον IEEE (Institute of Electrical and Electronics Engineers) από την σειρά πρωτοκόλλων 802.11 Error! Reference source not found.**. Μια εγκατάσταση WLAN μέσα σε ένα κτίριο είναι κατά βάση ένα κυψελικό δίκτυο που περιλαμβάνει κυψέλες που εξυπηρετούνται – η κάθε μια – από ένα σταθμό βάσης. Οι σταθμοί βάσεις λέγονται access point (σημεία πρόσβασης) και η περιοχή κάλυψης ενός access point ονομάζεται basic service area (BSA) (βασική περιοχή υπηρεσίας δηλ. βασική περιοχή κάλυψης). Το σύνολο των τερματικών που εξυπηρετούνται από το access point ονομάζεται basic service set (BSS) (βασικό σύνολο υπηρεσιών). Αρκετά BSSs μαζί μπορούν να συνδεθούν μέσω ενσύρματης υποδομής για να δημιουργήσουν μια extended service set (ESS) (εκτεταμένο σύνολο υπηρεσιών). Συγκεκριμένα τα WLAN positioning συστήματα μπορούν να βρεθούν σε αρκετά κτίρια στις μέρες μας καθώς υπάρχει μεγάλη δυνατότητα συνδεσιμότητας των κινητών συσκευών.**

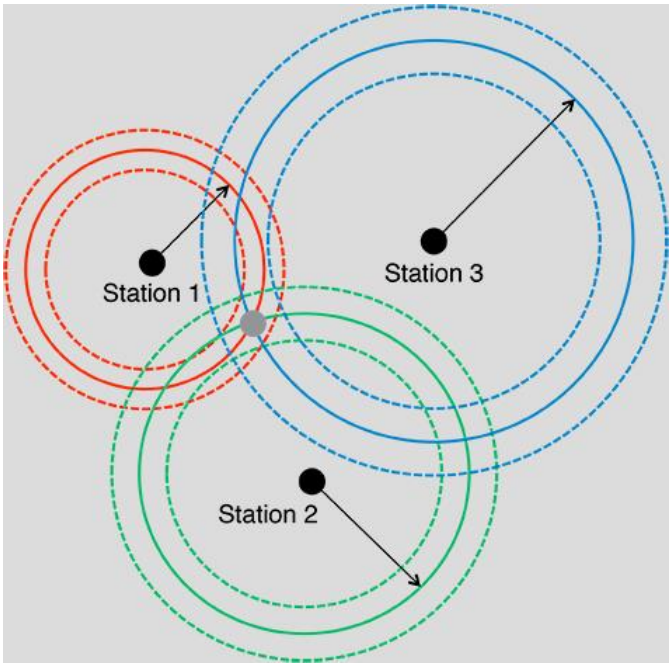
Τα περισσότερα πρότυπα WLAN συστήματα indoor position που έχουν αναπτυχθεί έως τώρα βασίζονται στις μετρήσεις ισχύος του λαμβανομένου σήματος (Received Signal Strength – RSS), τον λαμβανόμενο λόγο του σήματος ως προς τον θόρυβο (Signal-To-Noise – SNR), την ανίχνευση εγγύτητας και σπανιότερα την μέτρηση του χρόνου. Οι μετρήσεις SNR και RSS, βασίζονται στα λεγόμενα beacons (φάροι) τα οποία εκπέμπουν είτε στο uplink είτε στο downlink. Κατά την άφιξη ενός beacon, ο δέκτης μετρά το RSS (ή το SNR) και το κοινοποιεί στις εκάστοτε user-level εφαρμογές (διαδικασία η οποία είναι τυποποιημένο γνώρισμα της πλειονότητας του WLAN εξοπλισμού).

Για τις μετρήσεις στο uplink, όλα τα κινητά τερματικά πρέπει να δημιουργήσουν beacons, τα οποία πρέπει να μπορούν να λαμβάνονται από όλα τα access points που βρίσκονται εντός εμβέλειας. Αυτή είναι η βάση για την υλοποίηση των positioning μεθόδων σε ένα δίκτυο WLAN. Αντίστοιχα για τις μετρήσεις στο downlink, μια τυποποιημένη λειτουργία των WLAN γνωστή ως passive scanning (παθητική σάρωση), μπορεί να χρησιμοποιηθεί. Τα κινητά τερματικά εκτελούν συνεχώς την παθητική σάρωση για την ανίχνευση κοντινών access points και επιλέγουν το καλύτερο για μετάδοση. Για αυτό τον σκοπό, κάθε access point περιοδικά μεταδίδει ένα beacon, το οποίο περιέχει διάφορες παραμέτρους όπως είναι το timestamp (σφραγίδα χρόνου), οι υποστηριζόμενοι data types (τύποι δεδομένων) καθώς και ένα αναγνωριστικό μιας βασικής αρχιτεκτονικής οντότητας του δικτύου που αναφέραμε – τον Basic Service Set Identifier (αναγνωριστικό των BSSs). Το διάστημα εκπομπής μεταξύ δύο beacons μπορεί να παραμετροποιηθεί δυναμικά και είναι συνήθως δεκάδες ή εκατοντάδες χιλιοστά του δευτερολέπτου. Το τερματικό ακούει μόνιμα στα κανάλια λήψης beacons από τα κοντινά access points και καταγράφει τις πιθανές παραμέτρους και τις μετρημένες RSS και SNR τιμές. Έπειτα διαλέγει το access point με την καλύτερη ποιότητα σήματος για την εκπομπή. Αυτή η διαδικασία είναι αντίστοιχη με την διαδικασία που εκτελείται από ένα τερματικό GSM για την επιλογή του κατάλληλου σταθμού βάσης. Εναλλακτικά αν το τερματικό δεν λάβει ένα beacon κατά την διάρκεια του passive scanning στον ορισμένο χρόνο (due time) – πιθανότατα επειδή το διάστημα έχει ρυθμιστεί να είναι πολύ μεγάλο, μπορεί να στείλει ένα αίτημα probe (αίτημα διερεύνησης), οπότε όλα τα access points απαντάνε με ένα beacon. Αυτή η διαδικασία συντονισμού ονομάζεται active scanning. Έτσι, τόσο το active όσο και το passive scanning μπορούν να

χρησιμεύσουν ως βάση για την δημιουργία ενός συστήματος indoor positioning το οποίο είτε βασίζεται είτε υποβοηθείται από τα τερματικά (terminal-based ή terminal-assisted σύστημα).

Η παρατήρηση των beacons ή στο downlink ή στο uplink οδηγεί στον ορισμό 3^{ων} βασικών μεθόδων positioning:

- Proximity sensing (ανίχνευση εγγύτητας). **Βάσει αυτής της μεθόδου η ανίχνευση της θέσης του τερματικού γίνεται από την θέση του access point του οποίου το τερματικό ανίχνευσε ως βέλτιστο (με την καλύτερη ποιότητα σήματος).**
- Lateration. **Η θέση του τερματικού (σε σχέση με τα ενεργά access points) υπολογίζεται από το απώλεια ισχύος κατά την διαδρομή (path-loss) που μετράει ένα (ή περισσότερα beacons) κατά την διάρκεια της μετάδοσης. Ένα παράδειγμα tri-lateration φαίνεται στην Εικόνα 2.**



Εικόνα 2: Παράδειγμα εντοπισμό της θέσης ενός target terminal μέσω tri-lateration που μετριέται σε ένα σύστημα τριών stations, δηλαδή μέσω της μέτρησης του path-loss που έχουμε στα beacons κατά την μετάδοση τους εντός του πλέγματος των τριών σταθμών.

Fingerprinting (μέθοδος «δακτυλικών αποτυπωμάτων»). **Τα παρατηρηθέντα RSSs τα οποία προέρχονται από ή λαμβάνονται στα access points συγκρίνονται με ένα πίνακα προκαθορισμένων, πρότυπων RSS που συλλέχθηκαν σε διάφορες θέσεις. Η θέση για την οποία αυτή η σύγκριση δίνει το βέλτιστο αποτέλεσμα (προσομοιάζει δηλαδή καλύτερα), υιοθετείται στην συνέχεια ως θέση του τερματικού.**

Η μέθοδος proximity sensing (εγγύτητα ανίχνευσης), όπως και στα κυψελοειδή συστήματα, πάσχει από προβλήματα μειωμένης ακρίβειας αν και είναι η πιο απλή στην εφαρμογή, για την υποστήριξη της μεθόδου χρησιμοποιείται μια βάση

(δεδομένων) πού αντιστοιχεί τα BSSIs στους αριθμούς δωματίων. Κατά μια προχωρημένη προσέγγιση αυτής της μεθόδου, τα access points μπορούν να μεταδίδουν τους αριθμούς του δωματίου όπου βρίσκεται το καθένα, καθιστώντας έτσι την προαναφερθείσα βάση δεδομένων περιττή. Η ακρίβεια επιτυγχάνεται σε μια περιοχή δεκάδων ή εκατοντάδων μέτρων και εξαρτάται από την ισχύ του εκπεμπόμενου σήματος και την πυκνότητα των access points στο κτίριο. Στην χείριστη περίπτωση, ένα τέτοιο σύστημα μπορεί να χρησιμοποιηθεί μόνο για τον εντοπισμό ή μη ενός tagged person (ατόμου στόχου) εντός ενός κτιρίου ή σε μια συγκεκριμένη πτέρυγα του κτιρίου. Επίσης, μπορεί να είναι δύσκολο να γίνει διάκριση μεταξύ διαφορετικών ορόφων, το οποίο καθιστά την proximity sensing μέθοδο μη χρήσιμη για πολλές εφαρμογές. Στην καλύτερη περίπτωση, υπάρχει η δυνατότητα να εξάγουμε κάποιες πληροφορίες για την τοποθεσία αν είναι από πριν γνωστή η διασπορά των δωματίων. Για την εφαρμογή της μεθόδου lateration, απαιτείται η ακριβής ευθυγράμμιση των θέσεων των access points εντός του κτιρίου. Αυτές οι θέσεις μπορούν να συμβολιστούν μέσω ενός τοπικού καρτεσιανού συστήματος συντεταγμένων που επικαλύπτουν το σύνολο του κτιρίου είτε μέσω ενός αντιστοίχου συστήματος παγκόσμιας εμβέλειας όπως είναι το ECEF (Earth-centered, earth-fixed). Ωστόσο, περισσότερο προτιμητέα είναι η πρώτη επιλογή (τοπικό καρτεσιανό σύστημα) καθώς τα υπολογισθέντα σημεία θέσης μπορούν εύκολα να ανατεθούν στους αριθμούς των δωματίων (indexed room numbers), λαμβάνοντας υπόψιν τα σχέδια κατασκευής του κτιρίου. Σε σύγκριση με τα κυτταρικά ή τα δορυφορικά positioning systems, η μέθοδος lateration στα indoor positioning systems γενικά πάσχει από τις αρνητικές επιπτώσεις της διάδοσης πολλαπλών διαδρομών (multipath propagation), εφόσον δεν υπάρχει γραμμική οπτική επαφή (line of sight). Όταν μεταφέρονται από τον αποστολέα στον δέκτη, πολλές φορές τα σήματα beacon αντανακλώνται και διασπείρονται σε τοίχους και οροφές και παρουσιάσουν μια εξασθένιση που είναι δύσκολο να προβλεφθεί. Έτσι καθίσταται αδύνατη η εξαγωγή ενός ασφαλούς συμπεράσματος από το path-loss για την πιθανή απόσταση μεταξύ των access points και επιβάλλονται σημαντικά σφάλματα στον καθορισμό των θέσεων. Πρέπει να σημειωθεί ότι το path-loss δεν μπορεί να προέλθει αποκλειστικά από το RSS, είναι απαραίτητο να ξέρουμε και την ισχύ του μεταδιδόμενου σήματος, η οποία απαιτεί dedicated signaling (ειδική σηματοδοσία) μεταξύ του αποστολέα και του δέκτη.

1.4.1 WLAN fingerprinting για indoor positioning συστήματα

Σε συνέχεια των βασικών αρχών για τον fingerprinting [2], τις οποίες παραθέσαμε αναφορικά στο αμέσως προηγούμενο κεφάλαιο, συνεχίζουμε με την επεξήγηση της λειτουργίας της σύγκρισης των αποτελεσμάτων με τον πίνακα. Ο πίνακας (radio map) έχει καταχωρήσεις της μορφής (p, RSS_1, … RSS_n), όπου το p συμβολίζει την θέση αναφοράς και τα RSS_i (για i = 1,…,n) αναπαριστούν την τιμή RSS για το i-στο access point. Η τιμή του εκάστοτε RSS εξαρτάται από τις line-of-sight συνθήκες που επικρατούν στο σημείο, και αυτές οι συνθήκες μπορούν να διαφέρουν ανάλογα με την κατεύθυνση του στόχου. Για αυτό τα περισσότερα συστήματα καταγραφής των RSSs είναι της μορφής (p, d, RSS_1, …,RSS_n), όπου d είναι οι διαφορετικές κατευθύνσεις για κάθε ένα σημείο αναφοράς, δηλαδή d = north, south, west, east ή d= 0^o, 90^o, 180^o, 270^o. Στον πίνακα 1 έχουμε ένα παράδειγμα ενός radio map που υπακούει στις παραπάνω σχέσεις.

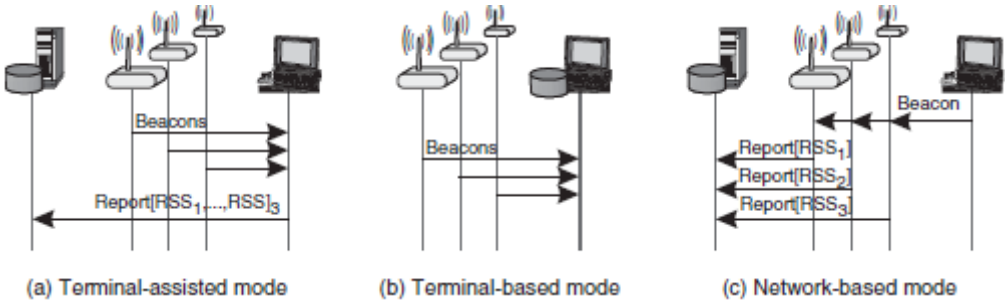
Πίνακας 1: Παράδειγμα συγκρότησης ενός radio map στον οποίο φαίνονται από 3 διαφορετικές θέσεις τα σήματα 3 διαφορετικών WLAN συσκευών.

Table 9.1 Example of a radio map

Position	Direction	RSS/[dBm] from 00:02:2D:51:BD:1F	RSS/[dBm] from 00:02:2D:51:BC:78	RSS/[dBm] from 00:02:2D:65:96:92
p_1	0°	−59	−75	−71
	90°	−54	−73	−67
	180°	−49	−72	−69
	270°	−55	−73	−65
p_2	0°	−35	−64	−50
	90°	−27	−64	−43
	180°	−40	−65	−52
	270°	−30	−60	−46
p_3	0°	−69	−66	−73
	90°	−65	−60	−68
	180°	−63	−66	−70
	270°	−68	−62	−76

Η αρχικοποίηση του πίνακα, γίνεται στην λεγόμενη off-line φάση, κατά την οποία έχουμε την εκπαίδευση ή την βαθμονόμηση του πίνακα με κάποιες αρχικές τιμές. Έπειτα στην on-line φάση οι τιμές των RSSs που σχετίζονται με τα σημεία στόχους (position target) καταγράφονται και συγκρίνονται έπειτα με τα RSS πεδία των καταχωρήσεων. Η θέση των position targets εξάγεται από τα σημεία αναφοράς της καταχώρησης που πλησιάζει περισσότερο στα position targets. Η διαδικασία της αντιστοίχησης γίνεται την χρήση ειδικών αλγορίθμων.

Η μέθοδος του fingerprinting μπορεί αν εκτελεστεί με 3 βασικούς τρόπους: terminal-assisted (υποβοηθούμενη από το τερματικό), terminal-based (βασισμένη στο τερματικό) καθώς και network-based (βασισμένη στο δίκτυο) (όπως φαίνεται και στην εικόνα 3).



Εικόνα 3: Τα 3 διαφορετικά modes για την για την εκτέλεση του fingerprinting και οι διαφορετικές ροές αιτημάτων και απαντήσεων που χρησιμοποιεί το καθένα.

Στα modes terminal-assisted και terminal based, ο πίνακας δημιουργείται από τις μετρήσεις RSS στο downlink και σε διάφορες θέσεις αναφοράς (reference positions) κατά την off-line φάση. Για αυτόν τον σκοπό, το τερματικό καταγράφει τα beacons που μεταδίδονται από τα access points προς όλες τις κατευθύνσεις

στην καλυπτόμενη περιοχή και καταγράφει τις σχετικές RSS τιμές. Η διαδικασία κατά την on-line φάση είναι η εξής: το τερματικό-στόχος μόνιμα καταγράφει τα RSSs με ίδιο τρόπο όπως και κατά την off-line φάση αλλά όμως κατά την κατεύθυνση των κινήσεων του χρήστη.

Στο terminal-assisted mode, στέλνει (το τερματικό στόχος) συχνές μετρήσεις σε ένα server στο δίκτυο, ο οποίος καταγράφει τον χάρτη και εκτελεί το ταίριασμα (matching process) των RSSs για να συμπεράνει την θέση. Στο terminal-based mode, ο πίνακας διατηρείται στο τερματικό και το matching process γίνεται τοπικά.

Για το network-based mode, η αρχικοποίηση είναι διαφορετική και ο πίνακας δημιουργείται από τις μετρήσεις των RSSs στο uplink. Στην off-line φάση, το τερματικό περιοδικά μεταδίδει beacons διαφορετικών κατευθύνσεων σε κάθε σημείο αναφοράς. Τα access points της καλυπτόμενης περιοχής λαμβάνουν αυτά τα beacons και καταγράφουν τα σχετικά RSSs. Τα αποτελέσματα των μετρήσεων συγχωνεύονται κεντρικά για την δημιουργία του πίνακα. Κατά την διάρκεια της on-line φάσης, το τερματικό στόχος πρέπει να μεταδίδει beacons περιοδικά για την μέτρηση των κοντινών access points, τα οποία (access points) έπειτα εκπέμπουν το report με τις μετρήσεις στον server, όπου γίνεται η διαδικασία του position matching.

Ένα βασικό μειονέκτημα της μεθόδου fingerprinting είναι το overhead για την δημιουργία του πίνακα στην off-line φάση (χρόνος, αποθηκευτικός χώρος κλπ), ιδιαίτερα αν αυτές οι μετρήσεις πρέπει να εκτελεσθούν για ένα ολόκληρο κτίριο. Το μειονέκτημα αυτό γίνεται πιο αντιληπτό κάθε φορά που αλλάζουν ή πρέπει να αλλάξουν οι ρυθμίσεις των access points, αν παραδείγματος χάριν αφαιρείται, διαγράφεται ή μετακινείται κάποιο από αυτά. Μια βελτίωση του συγκεκριμένου μειονεκτήματος της μεθόδου είναι η παραγωγή των πινάκων από μαθηματικά μοντέλα τα οποία υπολογίζουν τις ζητούμενες ρυθμίσεις λαμβάνοντας υπόψιν τις θέσεις των access points, την ισχύ των σημάτων που μεταδόθηκαν, το path-loss στον ελεύθερο χώρο και τα εμπόδια που ανακλούν ή σκορπούν τα σήματα (τείχη, έπιπλα στον χώρο κλπ). Με αυτόν τον τρόπο είναι εφικτή και εύκολη η δημιουργία και η αναβάθμιση κάθε φορά των πινάκων για ένα μικρό δίκτυο σημείων αναφοράς, όποτε οι ρυθμίσεις των access points αλλάζουν. Αυτός ο τρόπος ονομάζεται προσέγγιση μοντελοποίησης (modeling approach) ενώ η δημιουργία του πίνακα από μετρήσεις ονομάζεται εμπειρική προσέγγιση (empirical approach).

Η εμπειρική προσέγγιση μπορεί να διαιρεθεί περισσότερο σε δύο υποκατηγορίες: τη ντετερμινιστική και την πιθανολογική. Κατά την πρώτη, έχουμε την μέτρηση κάποιον δειγμάτων RSS, το οποία καταγράφονται για κάθε σημείο αναφοράς και κατεύθυνση, και έπειτα δημιουργείται ο πίνακας από τη μέση τιμή αυτών των δειγμάτων. Κατά την on-line φάση, η διαδικασία του matching ανάμεσα στα παρατηρηθέντα και στα καταγεγραμμένα RSS γίνεται βάσει κάποιων μετρικών. Μια βασική μέθοδος, που χρησιμοποιείται, είναι ο υπολογισμός της Ευκλείδειας απόστασης squareroot[$(RSS_{o,1} - RSS_{r,1})^2$ + … + $(RSS_{o,n} - RSS_{r,n})^2$], όπου ($RSS_{o,1}$,….,$RSS_{o,n}$) είναι το παρατηρηθέν RSS και το ($RSS_{r,1}$,….,$RSS_{r,n}$) είναι το καταγεγραμμένο RSS σε ένα συγκεκριμένο σημείο αναφοράς. Διατρέχονται όλα τα σημεία αναφοράς που είναι αποθηκευμένα στον πίνακα και το σημείο με την μικρότερη Ευκλείδεια απόσταση τίθεται ως η τρέχουσα θέση του στόχου μας. Αυτή η μέθοδος είναι γνωστή και ως Nearest Neighbor in Signal Space (NNSS) και παρουσιάστηκε από τον Bahl και τον Padmanabhan το 2000 (το paper στο οποίο είδαμε την εφαρμογή του εν λόγω

αλγορίθμου είναι το Error! Reference source not found.**). Βέβαια δεν αποκλείεται και η χρήση (και) άλλων μετρικών.**

Το μειονέκτημα των ντετερμινιστικών μεθόδων είναι ότι για να βρεθεί η «λύση» (το ταίριασμα δηλαδή της θέσης) για κάθε σημείο πρέπει να γίνει αναζήτηση σε ολόκληρο τον χάρτη και το matching γίνεται μόνο βάση του μέσου όρου του κάθε RSS. Αυτό μπορεί να επιφέρει μεγάλη υποβάθμιση στο επιθυμητό επίπεδο ακρίβειας αν τα RSSs παρουσιάζουν μεγάλες διαφοροποιήσεις. Μια πιο προηγμένη προσέγγιση είναι η μη καταγραφή των μέσων όρων (των RSSs) αλλά η καταγραφή των διαφοροποιήσεων που παρατηρήθηκαν στην ισχύ των σημάτων μέσω πιθανοτικών κατανομών κατά την off-line φάση (πιθανολογική προσέγγιση).

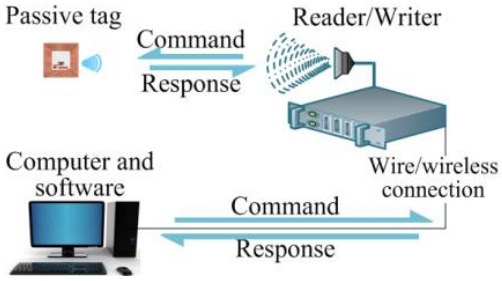
Τα τελευταία χρόνια τα συστήματα fingerprinting που έχουν αναπτυχθεί παρουσιάσουν ένα ανεκτό επίπεδο ακρίβειας για την χρήση τους σε indoor positioning περιβάλλοντα. Το βασικό πλεονέκτημα της μεθόδου WLAN fingerprinting είναι ότι βασίζεται στις υπάρχουσες εγκαταστάσεις WLAN και στον βασικό εξοπλισμό που βρίσκεται πλέον σε κινητά τηλέφωνα, laptops και άλλες συσκευές. Δεν απαιτεί τροποποιήσεις στο hardware παρά μόνο software για την δημιουργία των πινάκων καθώς και για την παρατήρηση των RSS στην on-line φάση.

1.5 RFID positioning

Η τεχνολογία RFID (Radio Frequency Identification) [6] χρησιμοποιεί ραδιοκύματα για την ασύρματη μετάδοση της ταυτότητας (π.χ. ενός ιδιαίτερου serial number ή ενός ιδιαίτερου tag) και άλλων πληροφοριών ενός αντικειμένου. Από το 2000 και μετά θα μπορούσαμε να πούμε ότι η τεχνολογία αυτή διανύει μια μεγάλη περίοδο ανάπτυξης και ακμής καθώς συνεπικουρούμενη από την ανάπτυξη της τεχνολογίας των αισθητήρων, έχει γίνει αναπόσπαστο κομμάτι της τεχνολογίας πυρήνα του IoT (Internet Of Things).

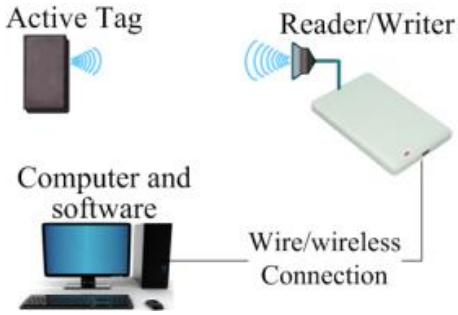
Υπάρχουν 3 βασικοί τύποι RFID συστημάτων: τα passive (παθητικά), τα semi-passive (ημί-παθητικά) και τα active (ενεργά). Ένα σύνηθες RFID σύστημα περιλαμβάνει tags (ετικέτες) – οι οποίες ονομάζονται αλλιώς και smart tags, smart labels ή radio barcodes (ραδιο-γραμμοκώδικες) –, ένα reader (ο οποίος αποτελείται από αποκωδικοποιητή, πομπό, δέκτη, interrogator (μετατροπέας για τον σένσορα) και writer) και ένα host υπολογιστή (ο οποίος μπορεί να είναι ένας υπολογιστής ή ένα σύστημα που έχει software αλλά και υποδομή). Ο reader με τον host υπολογιστή επικοινωνούν ενσύρματα ή ασύρματα.

Ένα passive RFID σύστημα ξεκινά από ένα passive tag το οποίο διαβάζεται από τον reader. Όταν ένα ραδιοσήμα στέλνεται από ένα reader και το tag έχει εισέλθει στην καλυπτόμενη περιοχή του reader, εκείνος θα ενεργοποιηθεί και θα πάρει το ID και τα δεδομένα από το tag και θα στείλει την πληροφορία αυτή στον host υπολογιστή. Ο υπολογιστής, έχοντας εγκατεστημένο πάνω του το RFID middleware – στην προκείμενη εργασία το Apache Storm – θα επεξεργαστεί τις πληροφορίες και θα τις στείλει πίσω στον reader. Έπειτα ο reader θα εκπέμψει τα επεξεργασμένα δεδομένα στο tag. Η δομή ενός passive RFID συστήματος φαίνεται στην εικόνα 4.



Εικόνα 4: Η αρχιτεκτονική ενός passive RFID συστήματος.

Ένα active RFID – εικόνα 5 – σύστημα συνήθως χρησιμοποιεί active tags (με ενσωματωμένη μπαταρία) και κάθε tag περιοδικά εκπέμπει τα δεδομένα του τα οποία περιλαμβάνουν την ταυτότητα του καθώς και άλλες πληροφορίες που αφορούν το σύστημα μας όπως τοποθεσία, τιμή, χρώμα ή ημερομηνία αγοράς (αν αναφερόμαστε σε σύστημα που υλοποιείται σε ένα εμπορικό κέντρο ή ένα supermarket). Ο RFID reader θα παραπέμψει τα δεδομένα του tag στη βάση δεδομένων που περιέχει. Σε σύγκριση με ένα passive σύστημα, ένα active σύστημα μπορεί ταυτοχρόνως να διαβάζει αρκετά tags στον χώρο και το εύρος ανάγνωσης που έχει είναι μεγαλύτερο με την ενέργεια που χρειάζεται παράλληλα να είναι λιγότερη. Η απόσταση ανάμεσα σε ένα reader και στον host υπολογιστή είναι συνήθως μικρότερη από 500 μέτρα αν υπάρχει ασύρματη ζεύξη σύνδεσης των δύο.



Εικόνα 5: Η αρχιτεκτονική ενός active RFID συστήματος.

Ένα semi-passive σύστημα είναι παρόμοιο με ένα passive εκτός από την μπαταρία που υπάρχει ενσωματωμένη στα semi-passive tags που χρησιμοποιούνται εντός του συστήματος. Η μπαταρία παρέχει on-board αρκετή ενέργεια (σε αντίθεση με το passive σύστημα, στο οποίο η μπαταρία τροφοδοτείται από τον reader) για την παρακολούθηση των τηλεμετρικών στοιχείων και των ενεργητικών στοιχείων του tag. Βέβαια η on-board power δεν χρησιμοποιείται απευθείας για την δημιουργία ραδιοσυχνοτικά ηλεκτρομαγνητικής ενέργειας.

1.5.1 RFID tags

Ένα τυπικό RFID tag αποτελείται από ένα microchip το οποίο συνδέεται με μια ραδιοκεραία η οποία είναι τοποθετημένη κάτω από ένα υπόστρωμα. Το microchip μπορεί να αποθηκεύσει δεδομένα από 26 bits έως 128 kbits. Οι κατηγορίες των RFID tags μπορούν να ταξινομηθούν αντιστοίχως σε 3 τύπους: passive, semi-passive (το οποία είναι γνωστά και ως battery-assisted passive – BAP) και active.

Τόσο τα active όσο και τα passive tags χρησιμοποιούν ενέργεια RF (radio frequency) για να επικοινωνήσουν με τον reader αλλά διαφέρουν ουσιαστικά ως προς την μέθοδο με την οποία τροφοδοτούν (με ενέργεια) τα tags. Ένα active tag χρησιμοποιεί εσωτερική πηγή τροφοδοσίας, συνήθως μπαταρίες, η οποία συνεχώς τροφοδοτεί το tag και τα RF κυκλώματα επικοινωνίας, ενώ ένα passive tag εξαρτάται εξ ολοκλήρου από την ενέργεια που λαμβάνει από τον reader. Η διάκριση έχει μεγάλο αντίκτυπο στην λειτουργικότητα του κάθε συστήματος. Τα semi-passive tags ξεπερνούν 2 βασικά μειονεκτήματα των passive RFID tags. Το ένα είναι η έλλειψη συνεχούς πηγής ενέργειας για τα κυκλώματα και το άλλο είναι το μικρό εύρος ανάγνωσης. Επίσης από κατασκευαστικής άποψης, τα semi-passive tags είναι ιδανικά για μια γρήγορη και αποδοτική ανάπτυξη προσαρμοσμένων RFID tags αφού δεν χρειάζονται πιστοποίηση FCC.

Υπάρχουν 2 βασικοί τύποι chic για τα RFID tags: read-only (μόνο ανάγνωσης) και read-write (ανάγνωσης και εγγραφής). Τα read-only chips και ο εξοπλισμός που χρειάζονται για να λειτουργήσουν είναι επαρκώς φθηνά, υλοποιώντας παράλληλα μια από τις βασικές λειτουργίες του RFID που είναι η μείωση της εμπλοκής του χειριστή (στην διαδικασία αντιστοίχησης και εξακρίβωσης των πληροφοριών θέσης κ.α.). Το μέγεθος των RFID tags ποικίλει ανάλογα με τους σκοπούς της κάθε εφαρμογής. Παραδείγματος χάριν τα RFID tags που χρησιμοποιούνται για τον εντοπισμό φορτηγών σε λιμάνια είναι στο μέγεθος ενός οικοδομικού τούβλου. Άλλα μπορούν να είναι μικροσκοπικά, όπως το «powder» της Hitachi (RFID powder 2007) το οποίο μπορεί να είναι όσο μικρό όσο ένας κόκκος άμμου (μεγέθους 0.4mm X 0.4mm ή ακόμα και 0.05mm X 0.05mm X 0.005mm, τα οποία έχουν 128-bit μνήμη σε read-only mode).

1.5.2 Τα δομικά συστατικά ενός RFID δικτύου

Αρχικά θα αναλύσουμε το αμέσως επόμενο (μετά τα tags) συστατικό του δικτύου, τον RFID reader. Ο RFID reader διαβάζει τα δεδομένα από τα tags και λειτουργεί ως αγωγός ή ως «γέφυρα» μεταξύ των tags και των controllers ή του middleware του συστήματος. Το πιο σημαντικό χαρακτηριστικό του reader είναι η το εύρος ανάγνωσης (reading range), το οποίο μπορεί να επηρεαστεί από ένα σύνολο παραγόντων όπως είναι η συχνότητα, η ενίσχυση της κεραίας (antenna gain), ο προσανατολισμός και η πόλωση που έχει η κεραία του, όπως επίσης και η κεραία του πομποδέκτη και η τοποθέτησή τους στον χώρο.

Το επόμενο συστατικό (host computer) μπορεί να είναι desktop ή laptop και πρέπει να είναι κοντά στους readers. Λαμβάνει δεδομένα από τους readers και εκτελεί διαδικασίες επεξεργασίας δεδομένων όπως φιλτράρισμα και ταξινόμηση. Χρησιμεύει επίσης και ως μια συσκευή παρακολούθησης της εύρυθμης, ασφαλούς και επικαιροποιημένης λειτουργίας του reader. Ο host computer και οι

readers επικοινωνούν μεταξύ τους συχνά με ένα πρωτόκολλο EPCglobal (EPC global Reader Protocol standard). Το πρωτόκολλο αυτό είναι δημιουργήθηκε από κοινές διαβουλεύσεις των οργανισμών GS1 και του GS1 US, οι οποίες αποσκοπούσαν στην δημιουργία ενός παγκοσμίου standard πρωτοκόλλου για το RFID και γενικότερα τις τεχνολογίες IoT.

Στους software παράγοντες, ο πρώτος που θα αναφέρουμε είναι το RFID middleware. Αρμοδιότητα του είναι να διευκολύνει την επικοινωνία μεταξύ των RFID readers και του enterprise system που βρίσκεται στο back-end του συστήματος. Συλλέγει, φιλτράρει και εφαρμόζει τους προαποφασισμένους κανόνες business logic κατά τους οποίους το σύστημα επεξεργάζεται τα δεδομένα που λαμβάνονται από τους readers. Επίσης, είναι υπεύθυνο για την παροχή λειτουργικοτήτων όπως είναι η διαχείριση και η παρακολούθηση της λειτουργικότητας και της συνδεσιμότητας των readers, της σωστής τους παραμετροποίησης και λειτουργίας. Το implementation του middleware μπορεί να ποικίλλει επίσης καθώς το σύστημα μπορεί να στηθεί σε ένα (και μόνο) host υπολογιστή, σε ένα κεντρικό server ή ακόμα και distributed σε «ευφυείς» αναγνώστες.

Όσον αφορά τις ραδιοσυχνότητες τις οποίες χρησιμοποιούν τα RFID συστήματα ξεκινάνε από τις χαμηλές και υψηλές συχνότητες και φτάνουν έως τα μικροκύματα και συγκριτικά με άλλες short-range τεχνολογίες έχουν μεγαλύτερο αναγνωστικό εύρος. Γενικά, μπορούμε να αναφέρουμε 4 κύριες ζώνες συχνοτήτων που διατίθενται για τις χρήσεις των RFID: χαμηλές συχνότητες (LF), υψηλές συχνότητες (HF), πολύ υψηλές συχνότητες (UHF) και πάρα πολύ υψηλές συχνότητες ή μικροκύματα (SHF/microwaves). Τα χαρακτηριστικά και οι επιδόσεις καθεμιάς από τις προαναφερθείσες ζώνες συχνοτήτων φαίνονται στο πίνακα 2. Επιγραμματικά οι βασικές εφαρμογές των LF συστημάτων είναι τα βοηθητικά συστήματα εντοπισμού κατοικίδιων ζώων, κάρτες πρόσβασης σε οριοθετημένους/ιδιωτικούς χώρους και τα συστήματα immobilization των αυτοκινήτων. Τα συστήματα HF χρησιμοποιούνται για εφαρμογές smart shelf σε malls και εμπορικά καταστήματα, για κάρτες πρόσβασης καθώς και έξυπνες κάρτες. Οι τεχνολογίες αυτές χρησιμοποιούνται επίσης και για επικοινωνίες NFC (Near Field Connection). Χαρακτηριστικό πλεονέκτημα αυτών των συστημάτων είναι η ικανότητα τους να λειτουργούν κοντά σε νερό ή μέταλλα. Από την άλλη μεριά, έχουν short reading range (< 1μ για τα passive RFID συστήματα). Τα UHF έχουν μεγαλύτερο reading range και μεγαλύτερο ρυθμό μεταφοράς δεδομένων από τα HF και έτσι είναι πιο χρήσιμα για την παρακολούθηση κινητών αντικειμένων ή ανθρώπων. Τα SHF συστήματα (τα οποία έχουν κύρια συχνότητα 2.45GHz – όπως το Wi-Fi και το bluetooth) έχουν αρκετά μικρότερα tags (σε σχέση με τα UHF) αλλά έχουν μικρότερο reading range.

Πίνακας 2: Παρουσιάζονται οι 4 ζώνες συχνοτήτων των RFID συστημάτων και τα κρίσιμα χαρακτηριστικά της καθεμιάς και τα γνωρίσματα τους ως προς τους διαφορετικούς τύπους RFID συστημάτων.

	LF	HF	UHF	SHF
FR (MHz)	< 0.135	3~28	433-435, 860-930	2400~2454 5725~5875
RR(P)	≤ 0.5 m	≤ 3 m	≤ 10 m	≤ 6 m
RR(A)	≤ 40 m	300 m	≤ 1 km	≤ 300 m
TRR	Slower ⟷ Faster			
ARMW	Better ⟷ Worse			

FR: Frequency Range
RRP: Typical Reading Range of Passive Tags
RRA: Typical Reading Range of Active Tags
TRR: Tag Reading Rate
ARMW: Ability to Read near Metal or Water

1.6 LBS middleware

Κατά αναλογία με τα «παραδοσιακά» middleware συστήματα (ένα σύνολο APIs, πρωτοκόλλων ή ακόμα και infrastructure services τα οποία υποστηρίζουν κατά βάση κατανεμημένες υπηρεσίες) ένα LBS middleware πρέπει να παρέχει τα προαναφερθέντα στοιχεία και θα πρέπει να είναι συμβατό με ευρύ φάσμα LBS εφαρμογών [7].

Πιο συγκεκριμένα, ένα LBS middleware είναι το σύνολο των υπηρεσιών, των αφαιρέσεων (data and programming modeling) και των μοντέλων που εφαρμόζουν τον συντονισμό του κινητού αντικειμένου ή του κινητού χρήστη, την συσχέτιση των πληροφοριών καθώς και τη μορφοποίηση των πληροφοριών διάδοσης. Ένας σημαντικός παράγοντας του LBS είναι η ενσωμάτωση της τοποθεσίας (location) ή των πληροφοριών της θέσης (position information). Για αυτό είναι κρίσιμο να γίνουν αντιληπτές οι δυνατότητες και οι περιορισμοί των υφιστάμενων τεχνολογιών εντοπισμού θέσης και το πως γίνεται η ενεργοποίηση της εγκατάστασης και της ανάπτυξης των εφαρμογών. Αυτές είναι οι αρχές πάνω στις οποίες πρέπει να σχεδιαστεί το LBS middleware.

Πιο συγκεκριμένα χαρακτηριστικά τα οποία πρέπει να έχει ένα LBS middleware, ανάλογα με την εφαρμογή που υλοποιεί, θα αναλύσουμε στην συνέχεια.

- **Push-based έναντι pull-based εφαρμογές**. Σε μια εφαρμογή LBS pull-based, τα αιτήματα πρέπει να αρχικοποιούνται από ένα κινητό τερματικό (π.χ. το τερματικό του χρήστη). Σε μια εφαρμογή που είναι push-based, η υποδομή αυτόνομα και με προρύθμιση προωθεί πληροφορίες στα κινητά τερματικά βάσει της εμφάνισης ενός γεγονότος ή ένα trigger μιας συγκεκριμένης κατάστασης.

- **Direct (άμεσο) έναντι indirect («έμμεσου») profile.** Μια εξατομικευμένη εφαρμογή συσχετίζει ένα service request με ένα προφίλ requester (αιτούντα) με τις ανάλογες πληροφορίες. Το προφίλ αυτό με τις πληροφορίες για τον αιτούντα μπορεί να συγκροτηθεί και να συγκεντρωθεί άμεσα από τον χρήστη στη φάση της δημιουργίας του λογαριασμού/προφίλ του ή αποκτώντας τις πληροφορίες από τρίτους. Το προφίλ μπορεί επίσης να περιέχει πληροφορίας σχετικά με το τρέχον context (πλαίσιο περιγραφής) του αιτούντα. Προφανώς, σε αυτόν τον τρόπο λειτουργίας εγείρονται πολλές ανησυχίες για την προστασία της ιδιωτικής ζωής και του ιδιωτικού απορρήτου αλλά δεν έχουν να κάνουν με τα χαρακτηριστικά του συστήματος καθ' αυτά.

- **Η διαθεσιμότητα των πληροφοριών του προφίλ.** Οι πληροφορίες του προφίλ μπορούν να διατίθενται κατά τον χρόνο της εκάστοτε αίτησης ή μπορεί να ήδη στην διάθεση του LBS συστήματος. Στην πρώτη περίπτωση, η πιο πιθανή «τοποθέτηση» των πληροφοριών του προφίλ θα ήταν το κινητό terminal. Έτσι, κάθε request για πληροφορίες θα υποβάλλει μέρος ή το σύνολο του προφίλ. Είναι σαφές ότι μια προσέγγιση επιλεκτικής μεθόδου push μοντέλου δεν μπορεί να υποστηριχτεί κατά αυτήν την προσέγγιση. Το πλεονέκτημα έγκειται στον τομέα του privacy καθώς οι πληροφορίες του προφίλ του κάθε χρήστη είναι προστατευμένες και δεν εκτίθενται, προς την οποιαδήποτε πιθανή υστερόβουλη εκμετάλλευση, στο κεντρικό site του υπηρεσίας LBS. Προφανώς βέβαια αυτή η επιλογή οδηγεί σε μεγάλη αύξηση του request payload.

- **Διαφορετικά σενάρια για την υλοποίηση του interaction.** Οι οντότητες που αλληλεπιδρούν, κινητά terminals και κεντρικό LBS ή γενικότερα, υπηρεσία requester και υπηρεσία provider, μπορούν να είναι είτε κινητές είτε σταθερές (stationary). Αυτή η παρατήρηση μας οδηγεί σε 4 πιθανά σενάρια αλληλεπίδρασης. Κατά το πρώτο σενάριο, τόσο ο requester όσο και ο provider είναι σταθεροί, και δεν υπάρχει ανάγκη για δυναμική διαχείριση των πληροφοριών τοποθεσίας. Κατά το δεύτερο ή το τρίτο σενάριο, όπου είτε ο requester είτε κινητός και ο provider είναι σταθερός είτε αντίστροφα, και οι δύο αυτές περιπτώσεις παρουσιάσουν συμμετρικότητα ως προς την οργάνωση του LBS middleware, ωστόσο η ερμηνεία και η συγκεκριμενοποίηση τους εξαρτάται από το πώς διαμορφώνεται η εφαρμογή (δηλαδή ποια οντότητα από τις δύο έχει τον ρόλο του provider και ποια του requester). Για παράδειγμα, για τις εφαρμογές εντοπισμού οχημάτων, το όχημα που εκπέμπει πληροφορίες για την τοποθεσία του είναι ο provider, ενώ οι LBS που αναλαμβάνουν τον ρόλο του requester (…των πληροφοριών για την εκάστοτε θέση του οχήματος). Στο τέταρτο σενάριο, τόσο ο requester όσο και ο provider είναι κινητοί, ενώ το σύστημα LBS αναλαμβάνει τον ρόλο του συντονιστή (και είναι σταθερό). Εφαρμογές που ταιριάζουν με αυτό το σενάριο είναι εφαρμογές στις οποίες οι κινητοί χρήστες ενδιαφέρονται για την θέση των υπολοίπων (π.χ. εφαρμογές friend finder, location-based παιχνίδια και augmented reality εφαρμογές/παιχνίδια). Οι εφαρμογές εκείνες οι οποίες δεν βασίζονται ρητά σε σταθερό συντονιστή χαρακτηρίζονται ως location-based ad-hoc. Σε αυτήν την περίπτωση, οι πληροφορίες βρίσκονται εξ ολοκλήρου σε κινητά τερματικά τα οποία τρέχουν όλη την εφαρμογή με συντονισμένο τρόπο και χωρίς να βασίζονται σε υποδομή δικτύου για τον συγχρονισμό της επικοινωνίας τους.

- **Η πηγή των πληροφοριών τοποθεσίας.** Οι πληροφορίες μπορούν να παρέχονται οικειοθελώς από τον χρήστη ή να διατίθενται από την υποδομή του δικτύου ή ακόμα και να παρέχονται από τρίτους. Στην πρώτη περίπτωση, οι πληροφορίες τοποθεσίας είναι μέρος του service request. Στις υπόλοιπες περιπτώσεις, μπορούν είτε να βρεθούν με ειδικό query από την εφαρμογή LBS ή να μεταδίδονται από το κινητό τερματικό.

- **Η ακρίβεια των πληροφοριών τοποθεσίας.** Ανάλογα με την τεχνολογία εντοπισμού θέσης που χρησιμοποιείται στην υποδομή του δικτύου προκύπτουν διαφορετικοί βαθμοί ακρίβειας για τις αιτήσεις εντοπισμού των κινητών τερματικών. Οι βαθμοί ακρίβειας κυμαίνονται από τον εντοπισμό ενός κινητού τερματικού σε ακτίνα μερικών μέτρων έως και πολλών χιλιομέτρων. Ο παράγοντας αυτός επηρεάζεται σε μεγάλο βαθμό από το είδος της εφαρμογής που πρέπει να υποστηριχθεί από το σύστημα.

- **Το επίπεδο statefulness του συστήματος (δηλαδή ο βαθμός της διατήρησης των διαφορετικών καταστάσεων από το interface του συστήματος).** Ένα interface χαρακτηρίζεται ως stateful αν το LBS σύστημα διατηρεί την κατάσταση σε πολλαπλές αιτήσεις παροχής υπηρεσιών και ως stateless αν δεν διατηρείται η κατάσταση και κάθε request υποβάλλεται σε επεξεργασία ανεξάρτητα από τα προηγούμενα requests. Για παράδειγμα, σε μια εφαρμογή παρακολούθησης, μπορεί να είναι σημαντικό να παρακολουθούνται οι παλαιότερες τοποθεσίες του αντικειμένου στόχου για να εξισορροπηθεί ο τρόπος πρόβλεψης (calibration of the prediction scheme) για την ακριβέστερη πρόβλεψη της μελλοντικής συμπεριφοράς του αντικειμένου.

- **Το είδος των πηγών πληροφόρησης.** Τα LBS services βασίζονται στον αποτελεσματικό συσχετισμό των πληροφοριών που προέρχονται από διαφορετικές πηγές. Γίνεται διάκριση μεταξύ των δυναμικών και των στατικών πηγών πληροφόρησης. Οι στατικές πηγές πληροφόρησης είναι οι βάσεις δεδομένων που αναφέρονται στο γεωγραφικό περιβάλλον (π.χ. τα οποία περιλαμβάνουν διαφορετικά είδη χαρτών, τοποθεσίες-αξιοθέατα και κτίρια στόχους). Οι δυναμικές πηγές πληροφόρησης προσφέρουν πληροφορίες σχετικά με την αλλαγή των ευρύτερων συνθηκών του περιβάλλοντος όπως η οδική κυκλοφορία, οι καιρικές συνθήκες και οι πληροφορίες για επιλεγμένες εκδηλώσεις. Οι πληροφορίες θέσης κινητών τερματικών μπορούν να ταξινομηθούν ως δυναμικές και κατά βάση αλλάζουν πολύ συχνά. Όπως αναφέραμε και σε προηγούμενα χαρακτηριστικά των LBS, τα προφίλ των χρηστών αποτελούν μια επιπλέον αλλά βασική πηγή πληροφοριών. Ανάλογα με τη διαθεσιμότητα τους είναι είτε στατικά διαθέσιμη στο LBS service είτε παρέχεται δυναμικά με κάθε request. Τέλος, οι πληροφορίες που παρέχονται από τρίτους παρόχους περιεχομένου αποτελούν άλλη μια πηγή πληροφοριών. Βέβαια είναι συνήθως βραχύβιο περιεχόμενο, όπως π.χ. διαφημιστικό περιεχόμενο, εκτός και αν μόνιμα αποθηκεύεται και επανεκπέμπεται από την υπηρεσία πληροφοριών.

Μια επιπλέον ταξινόμηση μπορεί να γίνει και βάσει του είδους των εφαρμογών που υλοποιούνται. Τα βασικά χαρακτηριστικά (όπως αναφέρθηκαν παραπάνω) των LBS middleware βάσει των διαφορετικών κατηγοριών εφαρμογών φαίνεται στον παρακάτω πίνακα 3. Οι κατηγορίες που εμφανίζονται είναι infotainment («πληροφοριο-διασκέδαση»), παρακολούθηση, διάδοση πληροφοριών, παιχνίδια, χρέωση και έκτακτης ανάγκης.

Πίνακας 3: Ο πίνακας των διαφορετικών χαρακτηριστικών των LBS middleware ανά διαφορετικό είδος LBS εφαρμογής. Τα είδη των εφαρμογών είναι 6.

	Infotainment	Tracking	Dissemination	Games	Emergency	Billing
Push vs. pull	Pull	Push toward tracking entity Tracking entity pull	Push	Pull and push	Pull and push	Push toward billing infrastructure
Direct vs. indirect profile	Both types of profiles possible	Direct profiles	Both types of profiles possible	Both types of profiles possible	Direct profiles	Direct profiles
Availability of profile	Request-time/on device and stored/at service	Stored/at service	Request-time/on device and stored/ at service	Stored/at service	Stored/at service	Stored/at service
Interaction	Mobile requester and stationary provider	All four cases	Mobile requester and stationary provider	All four cases	Mobile requester and stationary provider	Mobile requester and stationary provider
Source of location information	Mobile terminal and network	Mobile terminal and network	Network	Mobile terminal and network	Network	Network
Accuracy of location position	Less than 1 km accuracy is desirable Outdoor	Less than 200 m accuracy is desirable Outdoor	Less than 1 km accuracy is desirable Outdoor	Less than 100 m accuracy is desirable Outdoor	Less than 20 m accuracy is desirable Outdoor/Indoor	Less than 0.5 km accuracy is desirable Outdoor/Indoor
State/stateless	Stateless	State-based	Stateless	State-based	Stateless	Stateless
Static/dynamic source	Mostly static	n/a	Static and dynamic	Static and dynamic	n/a	n/a

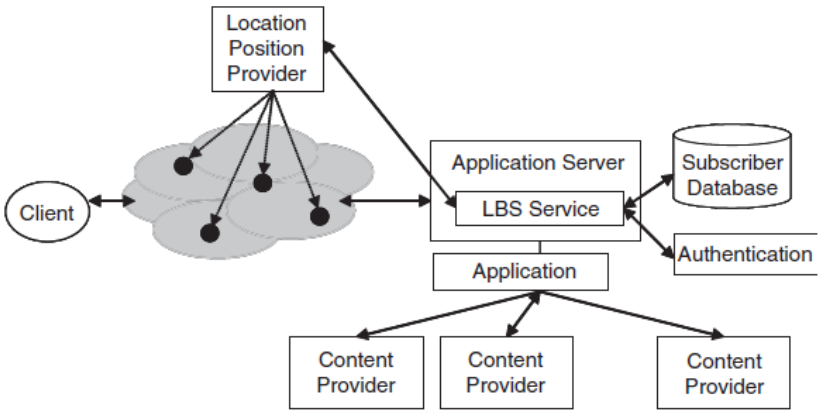
1.6.1 Στοιχεία αρχιτεκτονικής και μοντελοποίησης ενός LBS middleware

Ένα middleware γενικά εντός ενός συστήματος έχει την αρμοδιότητα της «απόκρυψης» των λεπτομερειών και των ιδιαιτεροτήτων του λειτουργικού συστήματος (λειτουργία masking) που βρίσκεται «από κάτω», των πρωτοκόλλων του δικτύου, των κατανεμημένων συστημάτων interacting καθώς και πιθανών τοπικών αποτυχιών που μπορεί να σημειωθούν. Τα APIs ενός middleware είναι θεμιτό να έχουν μεγάλη μεταφερσιμότητα και διαλειτουργικότητα γι' αυτό και

πρέπει να τυποποιούνται από διεθνείς οργανισμούς ή έστω να χρησιμοποιούν ευρέως αποδεκτές τυποποιήσεις [8].

Ένα LBS middleware προφανώς έχει τις ίδιες αρμοδιότητες. Αποσκοπεί στη διευκόλυνση της ανάπτυξης και της εγκατάστασης των LBS εφαρμογών σε ετερογενή δικτυακά περιβάλλοντα. Πρέπει να γεφυρώνει τα πρωτόκολλα και την δικτυακή με την ασύρματη τεχνολογία καθώς επίσης και την πιο σύγχρονη τεχνολογία του Internet. Είναι κοινός τόπος για τους developers αυτών των συστημάτων ότι το LBS middleware πρέπει να ενσωματώνει υποσυστήματα που είναι συνδεδεμένα όπως το SS7 (Signaling System 7) δικτυακή τεχνολογία, IP-based Internet δίκτυα και την τελευταία λέξη της ασύρματης τεχνολογίας.

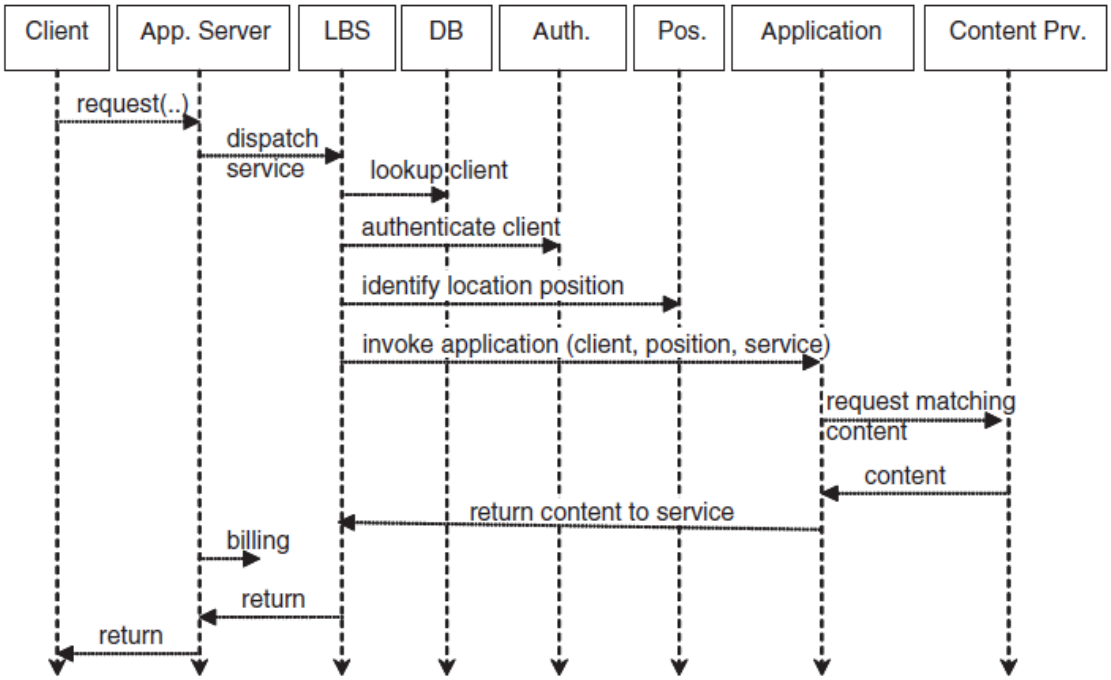
Το LBS middleware μπορεί να αναπτυχθεί είτε στο εντός του network operator της υπηρεσίας είτε να φιλοξενηθεί από μια εφαρμογή παροχής υπηρεσιών. Στα πλαίσια της ανάπτυξης (deployment), το LBS middleware συνδέει τους πελάτες (οι οποίοι βρίσκονται σε κινητά τερματικά) με το Internet, με άλλους παροχείς υπηρεσιών και τους network operators ώστε να προσφέρει τελικά μια ενιαία πύλη location-based εφαρμογών, που να αποτελείται από πολλές χωριστά παραμετροποιήσιμες υπηρεσίες. Το middleware ενοποιεί την υποδομή του δικτύου, συμπεριλαμβανομένων των location servers, των πυλών gateways επικοινωνίας, τις πύλες υπηρεσίας των συνδρομητών, την εξυπηρέτηση των πελατών, τις υπηρεσίες ενεργοποίησης των λογαριασμών των πελατών, τα συστήματα τιμολόγησης και τα λειτουργικά συστήματα. Υλοποιεί δηλαδή μια αρχιτεκτονική από άκρο σε άκρο (end-to-end) όπως παρουσιάζεται και στην εικόνα 6, στην οποία φαίνονται οι χρήστες που συνδέονται από κινητά τερματικά, ο network operator, οι άλλοι πάροχοι εφαρμογών καθώς και τα είδη των υπηρεσιών που παρέχονται στους συνδρομητές, στον network operator και των πάροχο της εφαρμογής.



Εικόνα 6: Η συνολική end-to-end αρχιτεκτονική και η αλληλεπίδραση μεταξύ των παραγόντων του συστήματος, η οποία συντονίζεται από το middleware.

Σε ένα LBS middleware διαφέρει ανά το είδος των υπηρεσιών που παρέχονται σε ένα συνδρομητή, στον network provider και στον πάροχο της εφαρμογής. Για τους συνδρομητές, ο ακριβής έλεγχος στα προφίλ, στις τοποθεσίες και στο context της υπηρεσίας είναι σημαντικά χαρακτηριστικά. Θα πρέπει να υπάρχει η δυνατότητα απενεργοποίησης και ενεργοποίησης των πληροφοριών για την τοποθεσία και το προφίλ ανά μεμονωμένη εφαρμογή υπηρεσίας (π.χ. να μπορεί να επιλέξει ποια από τις προσφερόμενες εφαρμογές θα πρέπει να λαμβάνει τις πληροφορίες για την τοποθεσία χρήστη και σε context). Οι network operators

και οι άλλοι παροχείς εφαρμογών (third-party application provider) θα πρέπει να εξυπηρετούνται με τιμολόγηση σε πραγματικό χρόνο, με συστήματα κατανομής των εσόδων, καθώς και με πρόσβαση στις πληροφορίες των συνδρομητών, αλλά και άλλες υπηρεσίες. Οι εφαρμογές βρίσκονται σε ανώτερα layers από το middleware, χωρίς να χρειάζεται να ξέρουν λεπτομέρειες για την υλοποίηση των υπηρεσιών χαμηλοτέρου επιπέδου. Μια λεπτομερής λογική αρχιτεκτονική του LBS middleware φαίνεται στην εικόνα 6, και μια invocation του συστήματος από άκρο σε άκρο που προέρχεται από το κινητό τερματικό, που διέρχονται από τις διαφορετικές στοίβες λογισμικού και τα architecture components φαίνονται στην εικόνα 7.



Εικόνα 7: Το διάγραμμα ακολουθίας ενός request σε μια LBS υπηρεσία, αρχίζοντας την ακολουθία από την πλευρά του client.

Δεν υπάρχει ένα μοναδικό πρότυπο ή μοντέλο αναφοράς που να περιγράφει με μοναδικό τρόπο τα στοιχεία ενός LBS middleware. Το πιο κρίσιμο στοιχείο του middleware συστήματος είναι η αντιστοίχιση της matching engine (ή event correlator, filter engine ή matching kernel). Ο σχεδιασμός αυτού του module υπαγορεύει τα είδη των υπηρεσιών που υποστηρίζονται συνολικά από το middleware. Συγκεκριμένα για την υλοποίηση της συγκεκριμένης διπλωματικής η βασική τοπολογία προσομοίωσης αφορά σε ένα δωμάτιο (ή γενικά ένα φυσικό χώρο) ο οποίος έχει 9 αισθητήρες σε μια συγκεκριμένη διάταξη όπως φαίνεται στην παρακάτω εικόνα: (εικόνα με αισθητήρες αριθμημένους και σε διάταξη).

2. Stream processing

Με τον όρο stream processing καλείται ένα paradigm (μοντέλο προγραμματισμού), που επιτρέπει σε εφαρμογές την ευκολότερη εκμετάλλευση μιας –περιορισμένης- μορφής παράλληλης επεξεργασίας. Τέτοιες εφαρμογές μπορούν να χρησιμοποιήσουν πολλαπλές προγραμματιστικές μονάδες όπως τα FPU μιας GPU ή τα FPGA (field programmable gate arrays), χωρίς τον ρητό χειρισμό δέσμευσης μνήμης, συγχρονισμού ή επικοινωνίας μεταξύ αυτών των μονάδων [9].

Το μοντέλο stream processing απλοποιεί το παράλληλο λογισμικό και υλικό περιορίζοντας το επίπεδο παράλληλης επεξεργασίας που μπορεί να επιτευχθεί. Δεδομένης μίας ακολουθίας δεδομένων (ενός stream), **μια σειρά από λειτουργίες (kernel functions) μπορεί να εφαρμοστεί σε κάθε στοιχείο του stream.**

2.1 Εφαρμογές

Το stream processing αποτελεί στην ουσία έναν συμβιβασμό, οδηγούμενο από ένα data-centric μοντέλο [10] που λειτουργεί πολύ καλά για παραδοσιακές DSP και GPU-type εφαρμογές (όπως η επεξεργασία εικόνας, βίντεο και ψηφιακού σήματος) αλλά λιγότερο κατάλληλου για επεξεργασία γενικού σκοπού όπως η τυχαία προσπέλαση δεδομένων σε μία βάση. Με τη θυσία κάποιας ευελιξίας στο μοντέλο, επιτυγχάνεται ευκολότερη, ταχύτερη και πιο αποτελεσματική εκτέλεση. Ανάλογα τα πλαίσια της εφαρμογής, η σχεδίαση του επεξεργαστή μπορεί να ρυθμιστεί με βάση την μέγιστη αποδοτικότητα ή με συμβιβασμό για ευελιξία .

Το Stream processing είναι ιδιαίτερα κατάλληλο για εφαρμογές που παρουσιάζουν τα εξής τρία χαρακτηριστικά:

-Compute Intensity, **το πλήθος των αριθμητικών πράξεων ανά I/O ή σε αναφορές της μνήμης συνολικά. Σε πολλές εφαρμογές επεξεργασίας σήματος σήμερα η αναλογία είναι 50 προς 1 και αυξάνει.**

-Data Parallelism, **υπάρχει σε έναν πυρήνα εάν η ίδια συνάρτηση εφαρμόζεται σε όλες τις εγγραφές ενός input stream και ένας αριθμός εγγραφών μπορεί να δεχθεί επεξεργασία ταυτόχρονα χωρίς αναμονή για τα αποτελέσματα των προηγούμενων εγγραφών.**

-Data Locality **είναι ένα ειδικός τύπος προσωρινής τοπικότητας κοινός στις εφαρμογές επεξεργασίας σήματος και πολυμέσων όπου τα δεδομένα παράγονται μία φορά, διαβάζονται μία ή λίγες φορές αργότερα στην εφαρμογή, και ποτέ ξανά.**

Παραδείγματα εγγραφών εντός streams περιλαμβάνονται:

1) Στην επεξεργασία εικόνων, κάθε εγγραφή μπορεί να είναι ένα μοναδικό pixel της εικόνας.

2) Σε έναν κωδικοποιητή βίντεο, κάθε εγγραφή μπορεί να είναι 256 pixel δημιουργώντας ένα macroblock δεδομένων

3) Στην επεξεργασία ασύρματου σήματος, κάθε εγγραφή μπορεί να είναι μια ακολουθία δειγμάτων που λαμβάνονται από μία κεραία.

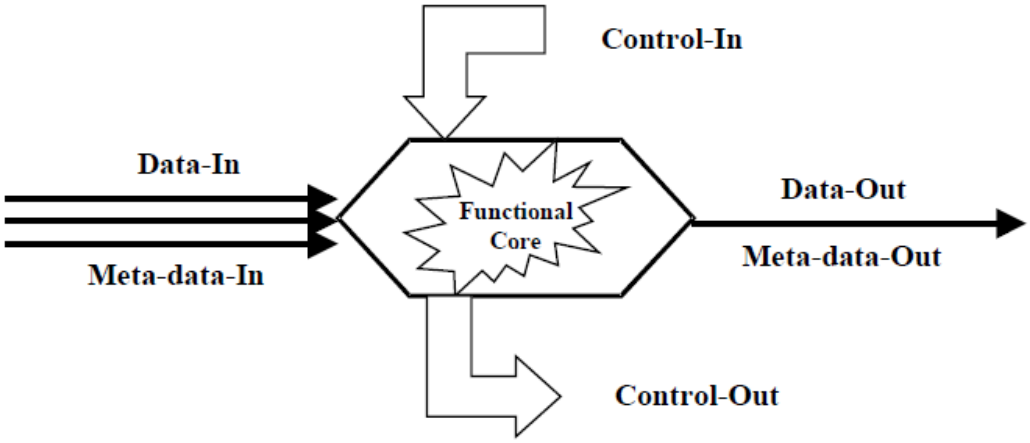
Για κάθε εγγραφή μπορούμε μόνον να διαβάσουμε από το Input, να την επεξεργαστούμε, και να γράψουμε στο output. Είναι επιτρεπτό να έχουμε πολλαπλά inputs και outputs, αλλά ποτέ κάποιο κομμάτι μνήμης που να είναι ταυτόχρονα αναγνώσιμο και ενγράψιμο.

2.2 Contextors

Contextor: capture and dynamic distribution of contextual information. Ως contextor μπορεί να νοηθεί ο μηχανισμός αφαιρετικότητας που εξάγει και διαμοιράζει δυναμικά την συναφή-επιθυμητή πληροφορία στα πλαίσια ενός Παρατηρούμενου Συστήματος (Observed System Context) [11].

Σύμφωνα με το Foundations for a Theory of Contextors των Coutaz και Rey υπάρχει διαφοροποίηση μεταξύ της έννοιας μιας στιγμιαίας κατάστασης ή «φωτογραφίας» των παρατηρήσιμων στοιχείων (μια περίπτωση), και της «σύνθεσης» των παρατηρήσιμων στοιχείων σε βάθος χρόνου (ένα context). Ο contextor είναι μια αφαιρετικότητα λογισμικού (software abstraction) που μοντελοποιεί σχέσεις μεταξύ των παρατηρήσιμων. Οι contextors εκμεταλλεύονται μία δομή Input/Output συμπεριλαμβανομένων των καναλιών ελέγχου και των μεταδεδομένων για να διασφαλίσουν QoS (δηλαδή ακρίβεια και σταθερότητα) καθώς και ιδιότητες όπως ανακλαστικότητα και διατηρησιμότητα.

Όπως φαίνεται στο ακόλουθο σχήμα (εικόνα 8), ένας contextor αποτελείται από έναν λειτουργικό πυρήνα, και από τα κανάλια επικοινωνίας εισόδου και εξόδου.



Εικόνα 8: Γραφική αναπαράσταση Contextor

Ο λειτουργικός πυρήνας (functional core) του Contextor υλοποιεί μία σχέση, μεταξύ των μεταβλητών στα πλαίσια του Παρατηρούμενου Συστήματος.

Για παράδειγμα, σε ένα Σύστημα ελέγχου θερμοκρασίας, όπου λαμβάνονται τιμές μέσω ενός αισθητήρα θερμότητας, η σχέση μεταξύ δύο μεταβλητών V1 και V3, όπου συμβολίζουν αντίστοιχα τη θερμοκρασία σε βαθμούς Farenheit και Κελσίου, το σενάριο μπορεί να προσομοιωθεί υπολογιστικά από έναν Contextor , του οποίου ο λειτουργικός πυρήνας εμπεριέχει την μετάφραση των δεδομένων Farenheit σε βαθμούς Κελσίου. Τα κανάλια εισόδου **ενός contextor είναι δύο τύπων:**

- Data-In **κανάλι το οποίο αντιστοιχεί στις μεταβλητές στα πλαίσια του Παρατηρούμενου Συστήματος που χρησιμοποιούνται ως είσοδοι από τον λειτουργικό πυρήνα του contextor. Κάθε τιμή εισόδου χαρακτηρίζεται από κάποιο Meta-data-in που περιγράφει ποιοτικά αυτή την τιμή εισόδου.**

- Control-In **κανάλι το οποίο αντιστοιχεί στις εντολές που λαμβάνονται από άλλους contextors προκειμένου να ορισθούν οι εσωτερικές παράμετροι του contextor. Οι παράμετροι αυτές αφορούν τόσο την λειτουργική συμπεριφορά του contextor καθώς και μη λειτουργική συμπεριφορά όπως το QoS που αναμένεται από τους άλλους contextors. Για παράδειγμα ένας contextor μπορεί να δεχτεί ένα σήμα "switch-off" στο κανάλι Control-In επειδή έχει αναγνωριστεί ως ελαττωματικός από έναν άλλον. Ή μπορεί να δεχτεί αίτημα QoS που εκφράζει το επίπεδο της ακρίβειας των τιμών που αναμένονται να παραδοθούν από τον contextor στο κανάλι Data-out.**

Κατά συμμετρία, τα κανάλια εξόδου **του contextor είναι δύο τύπων:**

- Data-out **κανάλι που αντιστοιχεί στις τιμές κάποιων μεταβλητών που επιστρέφονται, στα πλαίσια του Παρατηρήσιμου Συστήματος. Αντίστοιχα με τα δεδομένα εισόδου, έτσι και τα δεδομένα εξόδου συνοδεύονται από**

Meta-data-Out τα οποία περιγράφουν ποιοτικά την έξοδο που παράγεται από τον contextor.

- Control-Out **κανάλι που χρησιμοποιείται από τον contextor για την αποστολή σημάτων ελέγχου σε άλλους contextors. Για παράδειγμα, βασιζόμενος στα meta-data που σχετίζονται με τα δεδομένα που λαμβάνει από έναν άλλο contextor B, ένας contextor μπορεί να αποφασίσει να στείλει μία εντολή "switch-off" στον B.**

Επιπροσθέτως:

-Το κανάλι Data-In **δέχεται δεδομένα από το Data-out κανάλι, του οποίου ο τύπος δεδομένων είναι συμβατός με αυτόν του Data-In καναλιού.**

-Το Control-In **κανάλι δέχεται δεδομένα από το Control-Out κανάλι του οποίου ο τύπος δεδομένων είναι συμβατός με αυτόν του Control-In καναλιού.**

-Οι συνδέσεις **μεταξύ** Input και Output καναλιού **μπορεί να είναι** στατικές **(hardcoded),** ημιστατικές **(υπολογίζονται κατά το run-time όταν εκκινείται το σύστημα), ή και** παροδικές **(μπορούν να μεταβάλλονται δυναμικά).**

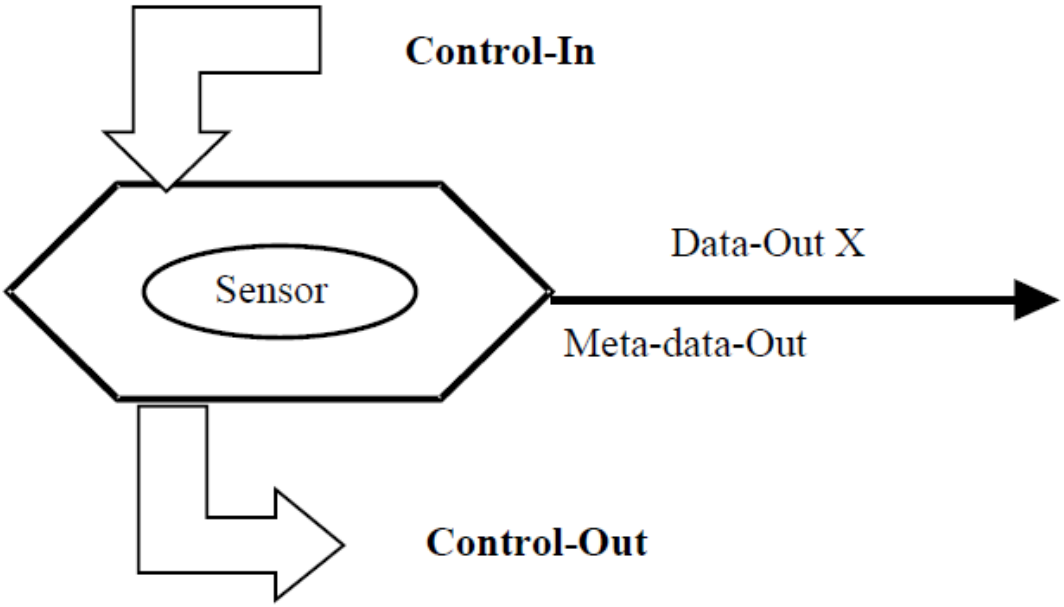
Δεδομένου ενός contextor A, ένας source contextor είναι ένας contextor που παρέχει στον A τιμές Data-In, και sink contextor είναι ένας contextor που λαμβάνει τιμές Data-Out από το A.

2.2.1 Ταξινόμηση των Contextors

Στο paper τους "Foundations for a theory of contextors" οι Joëlle Coutaz και Gaëtan Rey, προτείνουν μια κατηγοριοποίηση των contextors βασιζόμενοι στην επανεμφάνιση συγκεκριμένων λειτουργικών αναγκών των context aware συστημάτων. Τα είδη των contextors αναφέρονται παραπάκατω:

1. **Στοιχειώδης Contextor**

Ο στοιχειώδης Contextor δεν περιέχει κανάλι Data-In. Παρέχει έναν απλό τρόπο για να συμπεριλαμβάνεται ένας φυσικός σένσορας όπως ένα θερμόμετρο ή οποιοδήποτε υπολογιστικό στοιχείο μπορεί να χρησιμοποιηθεί σαν πηγή δεδομένων.

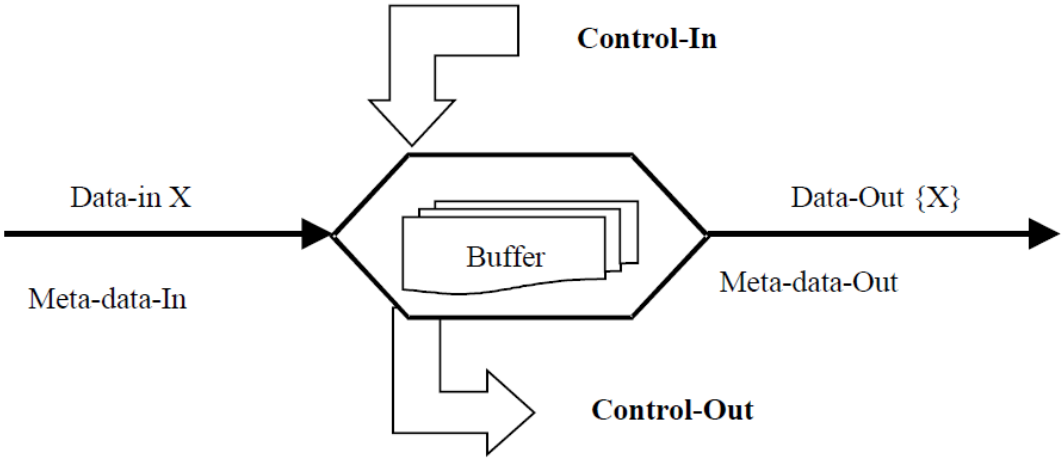


Εικόνα 9: Στοιχειώδης Contextor

Παραδείγματος χάριν, ας υποθέσουμε ότι συμπεριλαμβάνουμε μία βιντεοκάμερα εντός ενός στοιχειώδους contextor. Τότε, τα Data-out αντιστοιχούν στη ροή του video, το Control-in επιτρέπει σε άλλους contextors να ρυθμίζουν το ρυθμό ροής δεδομένων του βίντεο , καθώς και τις ρυθμίσεις pan και tilt για την κίνηση της κάμερας. Τα Meta-data-Out μπορούν να χρησιμοποιηθούν για να εκφράσουν την ανάλυση, τον αριθμό των bits ανά pixel, και γενικότερα, κάθε παράμετρο που μπορεί να περιγράψει ποιοτικά τις εικόνες που αποδίδει ο contextor. Το Control-out χρησιμοποιείται από τον contextor προκειμένου να αναγνωρισθεί η εκτέλεση των εντολών που έλαβε ο Contextor από το Control-in κανάλι.

2. History Contextor

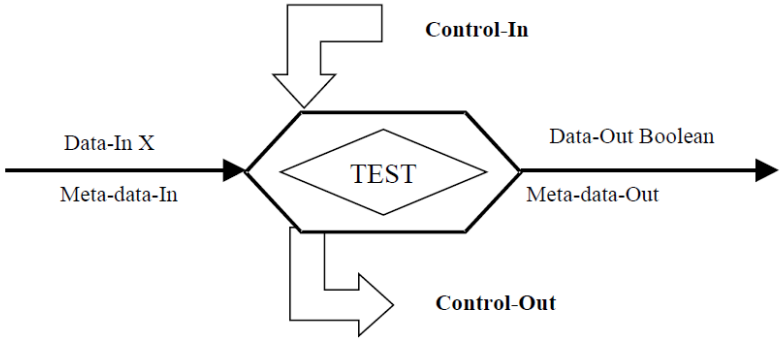
Ο history contextor αποθηκεύει τις τιμές και τα μεταδεδομένα τους που έχει λάβει με επιτυχία μέσω του Data-In καναλιού. Για παράδειγμα, το Data-In κανάλι ενός History Contextor λαμβάνει τις τιμές θερμοκρασίας που παρέχονται από έναν στοιχειώδη contextor θερμομέτρου. Εξ' ορισμού ο history contextor αποθηκεύει τις τιμές που λαμβάνει. Το Control-In κανάλι επιτρέπει σε άλλους contextors να ρυθμίσουν για παράδειγμα το πλήθος των τιμών προς αποθήκευση ή το χρόνικό περιθώριο πρωτού γίνει κλήση του garbage collector για εκ νέου περισυλλογή της μνήμης. Από το Control-Out, ο history contextor είναι δυνατόν να ενημερώσει τους source contextors προκειμένου να μειώσουν την ροή εκπομπής δεδομένων επειδή η μνήμη είναι σχεδόν πλήρης.



Εικόνα 10: History contextor

3. Threshold Contextor

Ο threshold contextor επιστρέφει την τιμή true εάν η τιμή στο κανάλι Data-In ικανοποιεί ένα συγκεκριμένο κατώφλι, αλλιώς επιστρέφει False.



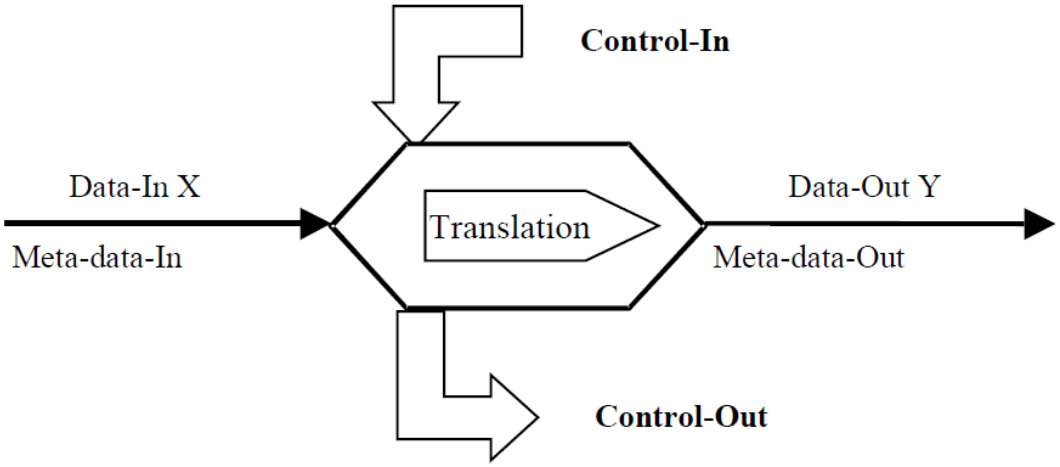
Εικόνα 11: Threshold Contextor

Για παράδειγμα, ας υποθέσουμε ότι ο contextor ενός κατωφλιού δέχεται τιμές θερμοκρασίας από ένα στοιχειώδη contextor θερμόμετρου. Το Data-out του threshold contextor σε αυτή την περίπτωση λοιπόν μπορεί να εκφράζει τις απαντήσεις στην ερώτηση «έχει ζέστη;» .

Το Control-In κανάλι επιτρέπει σε έναν άλλον contextor «απορρόφησης δεδομένων» (sink contextor) να σετάρει την οριακή θερμοκρασία. Για παράδειγμα, το χειμώνα, το κατώφλι θερμοκρασίας θα μπορούσαν να είναι οι 22 βαθμοί κελσίου, ενώ το καλοκαίρι οι 35 βαθμοί θα ήταν πιο κατάλληλοι για να χαρακτηριστεί μία ημέρα «ζεστή». Τέλος από το Control-Out κανάλι, ο threshold contextor μπορεί να για παράδειγμα ειδοποιήσει τον contextor πηγή να αλλάξει το ρυθμό ροής του καναλιού Data-out.

4. Translation Contextor

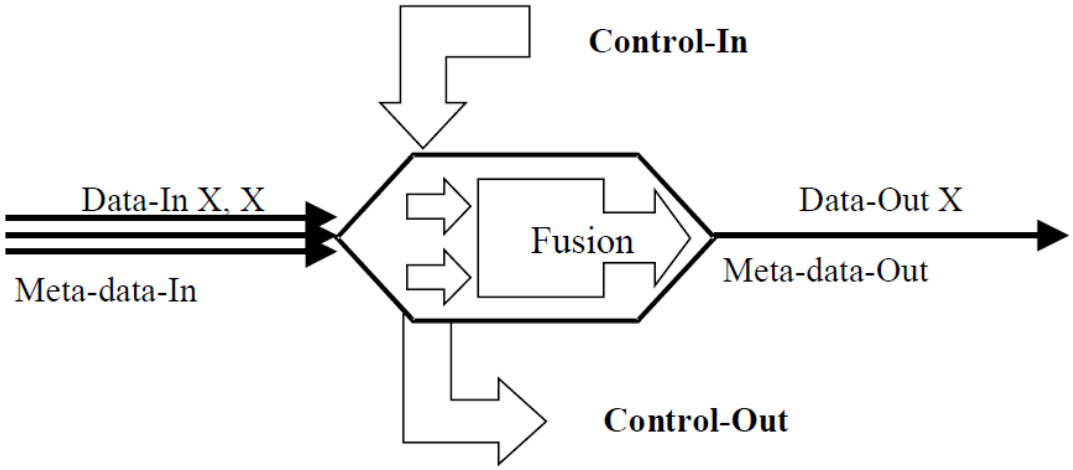
Ο translation contextor πραγματοποιεί recasting στους τύπους των μεταβλητών , αλλά δεν μεταβάλει κατά οποιοδήποτε άλλον τρόπο την σημασία ή το επίπεδο αφαιρετικότητατας των τιμών που έχουν ληφθεί από το κανάλι Data-In. Για παράδειγμα, ο translation contextor χρησιμοποιείται για να μεταμορφώνει τις θερμοκρασίες εισόδου, από ένα σύστημα αναπαράστασης σε ένα άλλο , δηλαδή από Κελσίου σε Fahrenheit ή Kelvin. Το κανάλι του Control-In επιτρέπει σε έναν sink contextor του να ρυθμίζει τον επιθυμητό τύπο δεδομένων για το Data-Out κανάλι.



Εικόνα 12: Translation contextor

5. Fusion Contextor

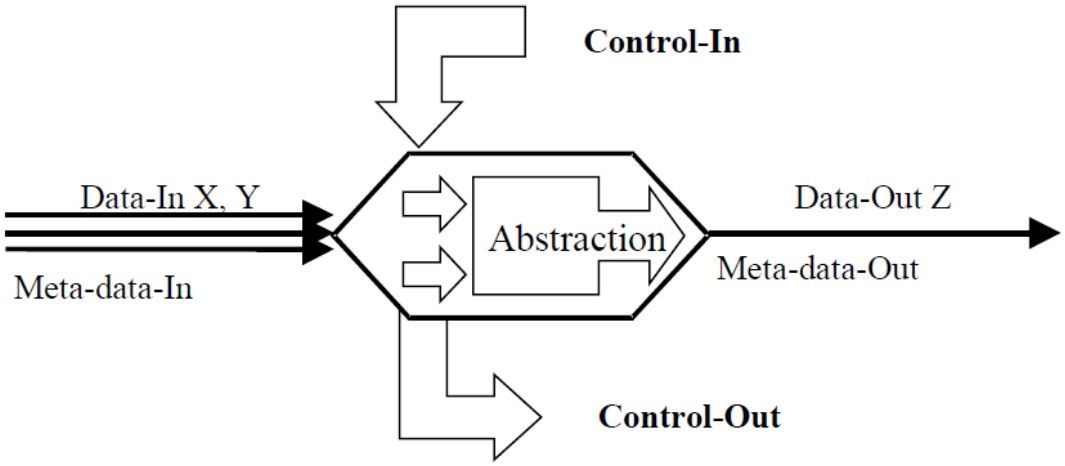
Οι Fusion contextors έχουν πολλαπλά Data-In του ίδιου τύπου, με κάθε ένα να φέρει διαφορετικού τύπου metadata. Ο ρόλος του fusion contextor είναι να παράγει ένα και μοναδικό Data-out του ίδιου τύπου, του οποίου η ποιότητα έχει βελτιωθεί από αυτή που είχε στο input data. Συνήθως, ένας fusion contextor λαμβάνει τον αριθμό τον ατόμων σε ένα δωμάτιο και από video και από audio trackers. Εξάγει τον αριθμό των ατόμων στο δωμάτιο αλλά με μεγαλύτερη ακρίβεια από ότι παρέχει είτε το video είτε το audio tracking μόνα τους. Το Control-In χρησιμοποιείται για να εκφράσει την σχετική σημαντικότητα των source contextors, ενώ το Control-out στέλνει εντολές στους source contextors όπως εκκίνηση/παύση της αποστολής τιμών.



Εικόνα 13: Fusion Contextor

6. Abstraction Contextor

Ο Abstraction contextor έχει πολλαπλά Data-In. Σκοπός του abstraction contextor είναι να παράγει ένα μοναδικό Data-Out του οποίου ο τύπος είναι σε ένα υψηλότερο επίπεδο αφαιρετικότητας από αυτό των input data types.



Εικόνα 14: Abstraction Contextor

Παράδειγμα: Ένας source contextor παρέχει σε έναν abstraction contextor το βάρος των κινούμενων αντικειμένων επάνω στο δάπεδο της αίθουσας ενός μουσείου. Ένας άλλος παρέχει στον abstraction contextor την ατμοσφαιρική θερμοκρασία του δωματίου. Από αυτές τις 2 ροές δεδομένων, ο abstraction contextor επιστρέφει το επίπεδο πληρότητας εντός του δωματίου. Μέχρι στιγμής έχουμε δει τις βασικές κατηγορίες contextors. Στη συνέχεια θα δούμε πως συντίθενται.

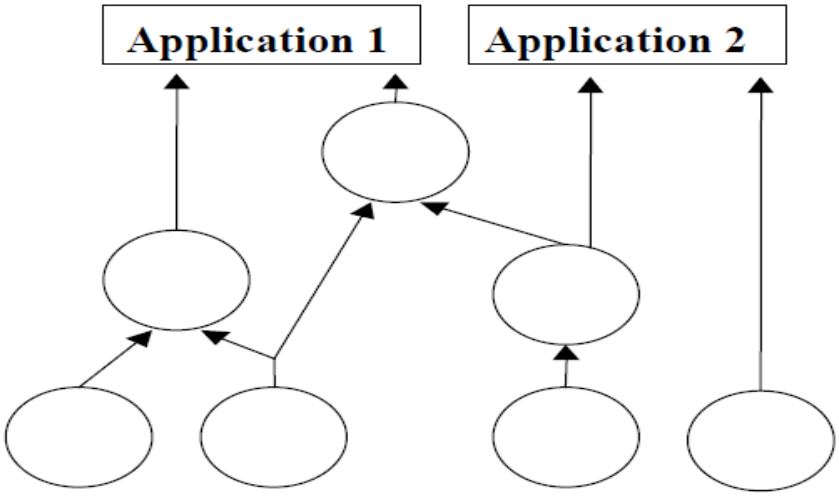
2.2.2 Συνθέτωντας Contextors

Οι contextors μπορεί να συντίθενται με 2 τρόπους: είτε μέσω της σύνδεσης καναλιών δεδομένων είτε μέσω της ενθυλάκωσης.

Σύνδεση καναλιών δεδομένων:

Οι contextors συντίθενται από τη σύνδεση Data-In καναλιών με συμβατά Data-Out κανάλια για να συνθέσουν έναν κατευθυνόμενο γράφο. Δύο κανάλια είναι συμβατά εάν χειρίζονται δεδομένα του ίδιου τύπου. Όπως φαίνεται στην εικόνα 15, το αποτέλεσμα είναι ένας κατευθυνόμενος γράφος που αποτελείται από μία

«αποικία» contextors. Οι contextors στην κορυφή του γράφου παρέχουν εφαρμογές με συναφή δεδομένα και στο κατάλληλο επίπεδο αφαιρετικότητας.

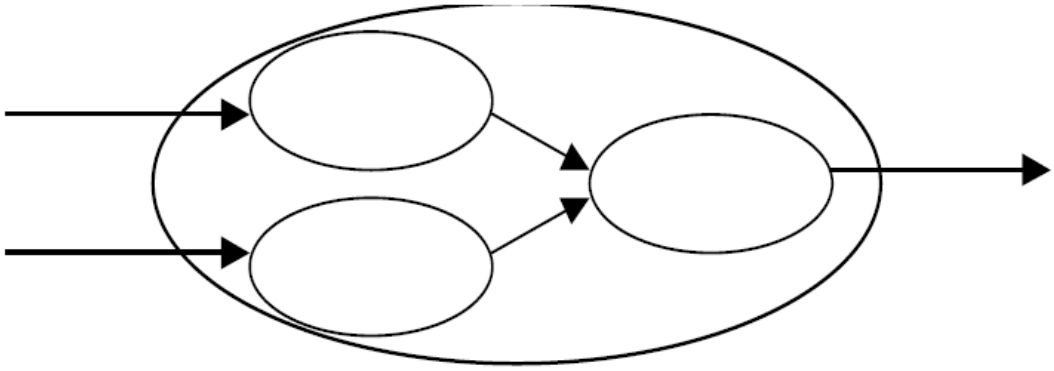


Εικόνα 15: Αποικία Contextors

Προς το παρόν μπορούμε να πούμε ότι το Control-In κανάλι ενός contextor είναι συνδεδεμένο με το Control-out κανάλι των sink contextors του, και ότι το Control-out κανάλι ενός contextor είναι συνδεδεμένο με το Control-In κανάλι των αντίστοιχων source contextors του.

Ενθυλάκωση:

Η ενθυλάκωση χρησιμοποιείται για την ομαδοποίηση μιας αποικίας contextors σε μία νέα κλάση contextor του οποίου η εσωτερική σύνθεση είναι κρυφή στους υπόλοιπους contextors. Η ακόλουθη εικόνα δείχνει ένα παράδειγμα abstraction contextor χτισμένου από μία αποικία απλούστερων contextors.



Εικόνα 16: Μία αποικία Contextors ενθυλακωμένων σε μία νέα κλάση contextor

2.3 Quality of Service και Ιδιότητες

Εν τέλει, σε ένα περιβάλλον αβεβαιότητας, το σύστημα πρέπει να είναι σε θέση να κρίνει αυτόματα την ποιότητα των υπηρεσιών που παρέχει, ώστε να μπορεί να προσαρμόζει την συμπεριφορά του. Σαν αποτέλεσμα οι contextors είναι σε θέση να μετρούν την ίδια την QoS που παρέχουν. Ένας Contextor τέλος, θα πρέπει να ικανοποιεί τις ακόλουθες ιδιότητες:

- Ανακλαστικότητα: **Ένας contextor είναι αυτο-περιγραφόμενος με την έννοια ότι η κατηγορία του, καθώς και τα input και output κανάλια του είναι ανιχνεύσιμα από άλλα στοιχεία λογισμικού, συμπεριλαμβανομένων άλλων contextors. Επιπροσθέτως, η ανακλαστικότητα παρέχει στον contextor την δυνατότητα να προσαρμόζει τη συμπεριφορά του με βάση τις αιτήσεις QoS που δέχεται από το Control-In κανάλι.**

- Διατηρησιμότητα: **Ένας contextor είναι δυνατόν να εξαφανιστεί εντελώς από την αποικία και, εάν είναι απαραίτητο, να επαναφέρει την κατάσταση του όταν επανέλθει σε εκτέλεση εντός της ίδιας ή κάποιας άλλης αποικίας.**

- Κινητικότητα: **Ένας contextor μπορεί να κινείται εντός της αποικίας, ρυθμίζοντας δυναμικά την συνδεσιμότητα του Input και Output καναλιού του.**

2.4 Stream Processing vs Real-Time Processing

Ένα σύστημα χαρακτηρίζεται ως real time stream processing [13] όταν υπάρχουν αυστηρές προθεσμίες εντός των οποίων κάποιο αποτέλεσμα είναι εγγυημένο [12]**. Για παράδειγμα, μπορούμε να χαρακτηρίσουμε την σύγχρονη smart TV ως ένα real time processing σύστημα: δεδομένου ενός ψηφιακού σήματος, εντός κάποιων milliseconds, πρέπει να φωτιστούν τα αντίστοιχα pixels στην οθόνη. Στα πλαίσια τώρα των software συστημάτων, ένα σύστημα συνήθως καλείται real time εάν έχει απαντήσεις εγγυημένα, στα πλαίσια «πραγματικών συνθηκών» με αυστηρές προθεσμίες. Ένα σύστημα που επεξεργάζεται τη ροή δεδομένων από ένα stock στον NASDAQ καθώς έρχεται από το δίκτυο, και εντός μερικών milliseconds θα καλούνταν σύστημα επεξεργασίας πραγματικού χρόνου.**

Στην πραγματικότητα, τα συστήματα πραγματικού χρόνου είναι εξαιρετικά δύσκολα στην υλοποίηση τους με τη χρήση κοινών συστημάτων λογισμικού. Για παράδειγμα, ο vanilla πυρήνας των linux δεν είναι πυρήνας πραγματικού χρόνου. Συγκεκριμένες λειτουργίες όπως το process scheduling, η επεξεργασία πακέτων δικτύου και άλλα υλοποιούνται χρησιμοποιώντας αλγορίθμους που δεν τηρούν αυστηρά κριτήρια προθεσμίας. Εάν μια διεργασία παραγκωνιστεί από τους πόρους της CPU από κάποια άλλη διεργασία υψηλότερης προτεραιότητας, ο scheduler μπορεί να μην της δώσει τους απαιτούμενους πόρους CPU που χρειάζεται για να απαντήσει εγγυημένα σε κάποια καθορισμένη προθεσμία, ανάλογα τον χρησιμοποιούμενο αλγόριθμο. Το ίδιο πράγμα ισχύει και για τα δικτυακά πακέτα. Υπάρχουν, φυσικά, εκδώσει του kernel που παρέχουν εγγυήσεις scheduling πραγματικού χρόνου όπως το QNX. Τα συστήματα λογισμικού σε αυτή την περιοχή συνήθως ανήκουν σε κάποιου είδους real time

processing ονόματι soft real-time processing όπου οι προθεσμία δεν είναι απόλυτη, αλλά περισσότερο μία πιθανότητα. Για παράδειγμα, το Amazon απαιτεί όλα τα υποστοιχεία λογισμικού στη σελίδα του να παρέχουν ένα αποτέλεσμα ή αποτυχία εντός 100-200 ms για 99% των αιτήσεων. Αυτό δίνει μια "soft" εγγύηση ότι μία σελίδα θα είναι έτοιμη προς απεικόνιση εντός του χρονικού ορίου.

Από την άλλη, με τον όρο stream processing **νοείται μία μέθοδος για συνεχείς υπολογισμούς, οι οποίοι συμβαίνουν καθώς τα δεδομένα βρίσκονται εν κινήσει μέσα σε ένα σύστημα. Στο πλαίσιο του stream processing δεν υπάρχουν αυστηροί χρονικοί περιορισμοί και εγγυήσεις για την ολοκλήρωση της επεξεργασίας και των υπολογισμών. Για παράδειγμα, ένα σύστημα το οποίο κατά το 99,9% απλά εξάγει τον πλήθος των λέξεων που παρουσιάζονται σε ένα Tweet στο Twitter αλλά κατά 0,1% εξάγει το πλήθος των μηνυμάτων στο οποίο εμφανίζεται η λέξη "Shakespeare" είναι ένα έγκυρο σύστημα stream processing. Δεν υπάρχει καθορισμένη προθεσμία για την λήψη output από το σύστημα όταν αυτό δεχθεί ένα input: τα δεδομένα δέχονται επεξεργασία καθώς εισέρχονται στο σύστημα και συχνά δεδομένα μπορεί να αναμένουν την επεξεργασία τους. Ο μόνος περιορισμός σε ένα τέτοιο σύστημα stream processing είναι ότι ο μακροπρόθεσμος ρυθμός output θα πρέπει να είναι ταχύτερος ή τουλάχιστον ίσος με τον ρυθμό άφιξης των inputs, αλλιώς η ανάγκες του συστήματος σε αποθηκευτικό χώρο θα αυξανόταν δίχως όριο. Επιπρόσθετα θα πρέπει να διαθέτει αρκετή μνήμη ώστε να διατηρήσει τα inputs που πιθανόν χρειαστεί να μπουν σε ουρά αναμένοντας εκτέλεση.**

Σημειωτέον ότι ο όρος «real-time» δεν έχει ακριβή ορισμό όσον αφορά την επεξεργασία δεδομένων, αλλά συνήθως μεταφράζεται ως λήψη αποφάσεων με χαμηλή καθυστέρη επάνω σε fast moving δεδομένα. Η μόνη περίπτωση που ίσως μπορούμε να μιλήσουμε για πραγματικά real-time συστήματα είναι σε safety-critical συστήματα. Αυτά τα συστήματα έχουν πολύ συγκεκριμένα κριτήρια λειτουργίας και προθεσμίες που πρέπει να τηρούνται. Για παράδειγμα, λογισμικά ελέγχου πτήσης ή το σύστημα ABS στα αυτοκίνητα. Παρόλα αυτά, τα συστήματα αυτά εξασφαλίζουν την αποκρισιμότητα τους αυτή με το να είναι ειδικά σχεδιασμένα για τη συγκεκριμένη εργασία, εξαλείφοντας όσο το δυνατόν περισσότερους εξωτερικούς παράγοντες, και χρησιμοποιώντας στοιχεία όπως firmware και hardware, τα οποία είναι εξαιρετικά προβλέψιμα και μετρήσιμα ως προς την καθυστέρηση τους.

Πέραν όμως των περιπτώσεων που περιγράφησαν από πάνω, το real-time μπορεί να σημαίνει απόκριση αρκετά γρήγορη ώστε να επιρρεάζει με τον επιθυμητό χρόνο το σύστημα στο οποίο βρίσκεται και να παρέχει έναν ρυθμό απάντησης που να μπορεί να αναπαραχθεί ακόμη και καθώς ο όγκος των δεδομένων αυξάνει, ακόμα και με την χρήση επιπρόσθετων κόμβων επεξεργασίας.

Το τελευταίο αποτελεί καίριο σημείο όταν κανείς εξετάζει εάν ένα σύστημα επεξεργασίας stream είναι πραγματικού χρόνου. Σε υψηλό επίπεδο, το διάγραμμα αρχιτεκτονικής ενός stream processing application έχει πολλές ομοιότητες με ένα σχήμα hardware κυκλώματος. Και τα δύο λειτουργούν βασιζόμενα στην επεξεργασία των δεδομένων record-by-record, στην επεξεργασία pipelines μέσω κόμβων επεξεργασίας/τροποποίησης, κάνοντας χρήση χρονοπαράθυρων και system clock, και έχοντας μία αρχιτεκτονική παράλληλης επεξεργασίας.

Με τον παραπάνω ορισμό, θα μπορούσαμε να πούμε ότι το Apache Storm είναι μια πραγματική πλατφόρμα Stream Processing καθώς επεξεργάζεται τα

αφιχθέντα δεδομένα record by record. Μπορούμε όμως να πούμε πως είναι και real-time; Είναι αρκετά εύκολο να παρέχουμε, πρακτικά, real-time response με την έννοια της χαμηλής καθυστέρησης, ας πούμε κάτω του δευτερολέπτου, με το αποτέλεσμα αυτό να αναπαράγεται, όταν ο ρυθμός δεδομένων είναι χαμηλός- για παράδειγμα της τάξεως των μερικών εκατοντάδων εγγραφών το δευτερόλεπτο [14].

2.5 Το Real-time S.P. ως ο μεγάλος παίχτης στον κόσμο των Big Data.

Η ανάγκη για επεξεργασία streams έχει αυξηθεί εκθετικά στη σύγχρονη εποχή. Ο λόγος είναι ότι συχνά, η επεξεργασία μεγάλου όγκου δεδομένων δεν είναι αρκετή. Τα δεδομένα θα πρέπει να επεξεργάζονται γρήγορα, έτσι ώστε ένας οργανισμός ή μία εταιρία να μπορεί να αντιδρά σε πραγματικό χρόνο απέναντι στις εναλλασσόμενες καταστάσεις και συνθήκες. Τέτοιες απαιτήσεις υπάρχουν τόσο στον τομέα του εμπορίου, στην ανίχνευση απάτης, στην παρακολούθηση συστημάτων, καθώς και πολλά άλλα παραδείγματα. Μια αρχιτεκτονική «εκτός χρόνου» δε μπορεί να εξυπηρετήσει τις ανωτέρω περιπτώσεις [15] .

2.5.1 Big Data vs Fast Data

Τα Big Data [[16] αποτελούν έναν από τους δημοφιλέστερους όρους στις μέρες μας. Ο καλύτερος τρόπος για να ορίσουμε τι είναι Big Data είναι τρείς παράμετροι:

-ο όγκος δεδομένων Volume **(tera, petabytes),**

-η ταχύτητα δεδομένων Velocity **(real or near-realtime),**

-η ποικιλία δεδομένων Variety **(social networks, blog posts, sensors).**

Μια Big Data αρχιτεκτονική περιέχει αρκετά μέρη. Συχνά, η μάζα δομημένων και ημιδομημένων ιστορικών δεδομένων αποθηκεύονται σε Hadoop (Volume+Variety). Από την άλλη μεριά, το stream processing χρησιμοποιείται για τις απαιτήσεις fast data (Velocity+Variety). Και τα δύο συμπληρώνουν το ένα το άλλο πολύ αποτελεσματικά.

2.5.2 Stream-processing και Streaming Analytics

Το stream processing ([17],[18]) αποτελεί την ιδανική πλατφόρμα για την επεξεργασία data streams ή sensor data (που συνήθως έχουν υψηλή σχέση event throughput έναντι πλήθους queries), ενώ από την άλλη το "complex event processing" (CEP) χρησιμοποιεί επεξεργασία event by event και συνάθροιση (για παράδειγμα σε γεγονότα out-of-order από μία ποικιλία πηγών).

Το stream-processing, το οποίο μελετάται, έχει σχεδιαστεί για να αναλύει και να δρα επάνω σε real-time streaming δεδομένα, χρησιμοποιώντας «συνεχείς επερωτήσεις» (continuous queries) (π.χ. επερωτήσεις τύπου SQL που δρούν εντός χρονοπαραθύρων). Βασικό κομμάτι του stream processing είναι το streaming analytics, ή η δυνατότητα για συνεχή υπολογισμό μαθηματικών ή στατιστικών σχέσεων εντός του stream και καθώς αυτό βρίσκεται εν κινήσει. Οι λύσεις που στοχεύονται στο stream processing είναι σχεδιασμένες να διαχειρίζονται υψηλό όγκο δεδομένων σε πραγματικό χρόνο με μία κλιμακούμενη, ευρέως διαθέσιμη και με ανοχή σε σφάλματα αρχιτεκτονική. Αυτό επιτρέπει την ανάλυση των δεδομένων εν κινήσει.

Σε αντίθεση με το παραδοσιακό μοντέλο των βάσεων, όπου τα δεδομένα πρώτα αποθηκεύονται και κατατάσσονται με τον σωστό τρόπο και ύστερα δέχονται επεξεργασία από επερωτήσεις, το stream processing παίρνει τα εισερχόμενα δεδομένα καθώς διέρχονται μέσα από έναν server. Με το stream processing είναι επίσης δυνατό μέσω εξωτερικών συνδέσεων, διάφορες εφαρμογές να εισάγουν συγκεκριμένα δεδομένα εντός της ροής, ή να ανανεώσουν μια εξωτερική βάση δεδομένων με την επεξεργαζόμενη πληροφορία. Ένα εργαλείο stream processing έχει να επιλύσει διάφορες προκλήσεις όπως:

- Την επεξεργασία τεράστιων ποσοτήτων streaming γεγονότων (φιλτράρισμα, συνάθροιση, αυτοματοποίηση, πρόβλεψη, δράση, παρακολούθηση και προειδοποίηση).

- Ανταπόκριση σε πραγματικό χρόνο στις μεταβαλλόμενες συνθήκες του συστήματος.

- Επιδόσεις και κλιμάκωση καθώς ο όγκος των δεδομένων αυξάνει σε μέγεθος και πολυπλοκότητα.

- Ταχεία ενσωμάτωση με τις υπάρχουσες δομές και πηγές δεδομένων τόσο για το input όσο και το output.

- Analytics: Ζωντανή παρακολούθηση των δεδομένων, συνεχής επεξεργασία επερωτήσεων και αυτοματοποιημένες προειδοποιήσεις και αντιδράσεις.

2.5.2.1 Περιπτώσεις χρήσης Stream Processing

Το Stream processing βρήκε μία από τις πρωταρχικές του εφαρμογές στο χώρο των οικονομικών, καθώς μετοχές και κεφάλαια μεταπήδησαν από floor based διακίνηση σε ηλεκτρονική. Σήμερα, μπορεί να βρει εφαρμογή σχεδόν σε κάθε τομέα, οπουδήποτε γεννώνται δεδομένα μέσω ανθρώπινης δραστηριότητας, μέσω μηχανημάτων ή αισθητήρων. Στην περίπτωση που το Internet of Thing απογειωθεί, θα πολλαπλασιάσει τον όγκο των δεδομένων, την ποικιλία και την ταχύτητα τους, οδηγώντας σε μία δραματική αύξηση των εφαρμογών για επεξεργασία stream [19].

Κάποιες περιπτώσεις χρήσης όπου το stream processing μπορεί να δώσει τη λύση σε προκλήσεις που παρουσιάζονται είναι:

- Network monitoring

- Intelligence and surveillance

- Risk management

- E-commerce

-Fraud detection

-Transaction cost analysis

-Pricing and analytics

-Market data management

-Algorithmic trading

-Data warehouse augmentation

3. ΑΝΑΛΥΣΗ ΤΟΥ ΚΩΔΙΚΑ ΥΛΟΠΟΙΗΣΗΣ

Το περιβάλλον υλοποίησης ήταν το Apache Storm [20] και το Eclipse IDE [21]. Οι εκδόσεις τους ήταν οι (Eclipse) Version Neon Release (4.6.0) και (Apache Storm) 1.0.2. Το λειτουργικό σύστημα βάσης ήταν το Ubuntu 16.04. Με την χρήση JAR components και εξαγωγή του κώδικα ως Project μέσω του Eclipse δοκιμάστηκε η

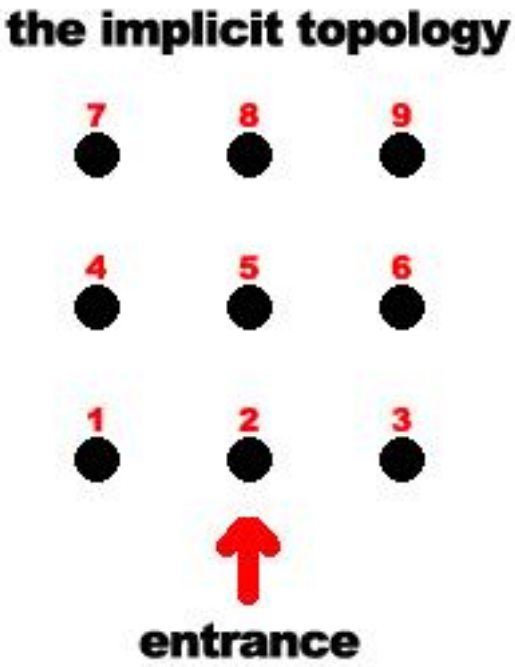
διαλειτουργικότητα της πλατφόρμας (σε άλλα λειτουργικά συστήματα – MAC OS 10.5 και Windows 7 και 8) με πλήρη επιτυχία.

Αναλυτικότερα στον φάκελο lib είναι φορτωμένα (τα βασικά στοιχεία του πυρήνα του Storm) τα .jar αρχεία που φαίνονται στον Πίνακα 5.

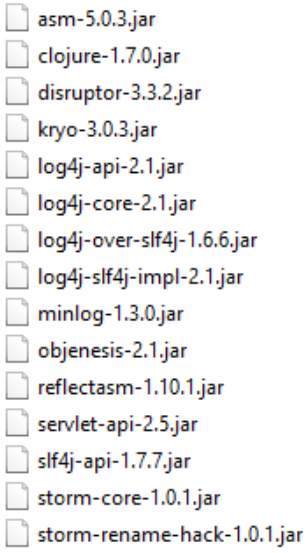
Παράλληλα όπως φαίνεται και στον πηγαίο κώδικα, η είσοδος (trace RFID από τις κεραίες του indoor συστήματος) πρέπει να βρίσκεται σε ένα αρχείο test.txt (το οποίο μπορεί προφανώς να περιλαμβάνει πάρα πολύ μεγάλο αριθμό traces της ίδιας μορφής δηλάδη: input/antena_code”antenanumber”{"tag": "tagnumber","timestamp": "timestampvalue"}) όπως επίσης και το σήμα KILL_SIG του οποίου η λειτουργικότητα επεξηγείται αμέσως παρακάτω.

Η τοπολογία που υπονοείται στο σύστημα μπορεί να φανεί στον πίνακα 5. Οι μαύρες βούλες συμβολίζουν τις 9 κεραίες που δίνουν σήμα στο σύστημα. Η είσοδος στο δωμάτιο βρίσκεται στην κεραία 2.

Πίνακας 4: Η υπονοούμενη τοπολογία που εισάγουμε στο σύστημα, μπορεί να φανεί η διάταξη των 9 κεραιών.



Πίνακας 5: Τα απαραίτητα .jar αρχεία που απαιτούνται για το build του προγράμματος



3.1 Ανάλυση τη αρχιτεκτονικής και των λειτουργιών που υλοποιήθηκαν

Παρακάτω θα αναφέρουμε ολόκληρη την αρχιτεκτονική και τις επιμέρους λειτουργίες που υλοποίησαμε με βάση τα εκάστοτε bolts and spouts. Έχουμε συνολικά 8 οντότητες, από τις οποίες η μια είναι υποβοηθητική για το «σεττάρισμα» του stream (Trace.java) και από τις υπόλοιπες το StormBringer.java είναι από το οποίο εκκινεί η λειτουργία όλου του συστήματος και έχουμε 1 spout (Line Reader Spout) και 5 bolts (TimeStampBolt, RouteTracerBolt, WordSpitterBolt, BusyAntennaCounterBolt, AvgTimePerTagBolt).

StormBringer: **Εδώ σετάρεται η τοπολογία. Ορίζονται τα bolts και τα spouts μας, τα streams στα οποία το κάθε bolt θα "ακούει" και θα "μιλάει". Για παράδειγμα, δημιουργούμε το spout LineReaderSpout με stream id** "line-reader-spout"**. Το bolt WordSpitterBolt ακούει από το stream line-reader-spout, και βγάζει output στο stream του με id** "word-spitter"**. Με όμοια λογική, όλα τα bolts δέχονται input από output streams άλλων bolts ανάλογα τα δεδομένα που χρειάζεται να αποκτήσουν. Τέλος γίνεται εκκίνηση της τοπολογίας.**

Trace: **Το trace είναι μια βοηθητική κλάση που έχει όλα τα fields που υπάρχουν στο trace. Έχει ανάλογους set -ers και get -ers καθώς και έναν initializer.**

LineReaderSpout:**Το Spout μας. Διαβάζει από txt file ένα input γραμμή γραμμή (record by record) Για την προσομοίωση των inputs από τους sensors γίνεται η χρήση αρχείου που περιέχει δεδομένα εφάμιλλα και με τη μορφή αυτών που περιέχει το trace από τους sensors και με μία καθυστέρηση κάποιων ms που ορίζουμε ώστε να προσομοιώνονται οι «αφίξεις από sensors» . Αφού διαβάσει μία γραμμή την κάνει emit ως tuple στο** default stream **του.**

WordSpitterBolt: **Με id word-spitter, ακούει στο default stream του line-reader-spout και εκπέμπει σε 3 διαφορετικά streams. Παίρνει το trace από τον**

LineReaderSpout και το χειρίζεται με string splits, concatenations και manipulations ώστε να εξάγουμε σε 3 μεταβλητές τα antenna_code , tag και timestamp. Εάν αντί για το σύνηθες trace διαβάσει το σήμα που έχουμε ορίσει ως KILL_SIG, τότε θα το στείλει αμέσως, χωρίς να πραγματοποιήσει επεξεργασία πάνω του. Το signal αυτό έχει σκοπό όταν διαβάζεται να εκτυπώνει τη μέχρι στιγμή πρόοδο και στατιστικά των άλλων bolts, τα οποία θα δούμε αργότερα. Στο WordSpitterBolt:

a) Δημιουργούμε ArrayList με όνομα route_inputs στο οποίο κρατάμε τα δεδομένα αυτά και ύστερα το εκπέμπουμε εξ'ολοκλήρου στο route_stream**.**

b) Ξεχωριστά, εκπέμπουμε μόνο τη μεταβλητή με το antenna σε άλλο stream με όνομα antennaStream**.**

c) Ξεχωριστά, εκπέμπουμε μόνο τη μεταβλητή με τα timestamps σε άλλο stream με όνομα timestampStream**.**

BusyAntennaCounterBolt: **Ακούει στο stream antennaStream. Δέχεται τα antenna code μονάχα από το κάθε input και τα καταμετρά uniquely. Συγκεκριμένα, γίνεται χρήση TreeMap<String,Integer> με το όνομα** counters **όπου τον ρόλο του key παίζει το εισερχόμενο tuple, δηλαδή στο συγκεκριμένο bolt το ο κωδικός της κεραίας antenna_code. Αφού γίνεται δυναμικά προσμέτρηση των κεραιών που συναντόνται, τότε εάν αντί για το σύνηθες trace διαβάσει το σήμα που έχουμε ορίσει ως KILL_SIG, καλεί τη συνάρτηση progress η οποία περνάει τα δεδομένα από το** counters **σε ένα Set με όνομα** set**, το οποίο το περνάμε στη συνέχεια σε Arraylist με όνομα** list **την οποία και ταξινομούμε. Έτσι, εμφανίζουμε τα antenna codes με φθίνουσα σειρά από το πιο "Hot", αυτό δηλαδή από το οποίο έχουν περάσει οι περισσότεροι, προς το λιγότερο "Hot".**

TimeStampBolt: **Το Bolt αυτό ακούει στο timestampStream από τον word-spitter. Δέχεται μονάχα τα timestamps από το κάθε input. Βρίσκει τη διαφορά μεταξύ της κάθε χρονικής στιγμής που είχαμε input μέχρι την επόμενη χρονική στιγμή που είχαμε input. Ύστερα αθροίζει τις διαφορές αυτές, και διαιρεί με το (πλήθος - 1) των inputs που είχαμε προκειμένου να βρεθεί ο μέσος όρος που μεσολαβεί από το ένα input στο επόμενο.**

RouteTracerBolt: **Το Bolt αυτό ακούει στο routeStream από τον word-spitter. Αποθηκεύει κάθε unique trace σε ένα arraylist με όνομα** cleanup_array **και με χρήση του πεδίου timestamp να υπολογίζεται χρόνος ως καθαρός αριθμός προκειμένου να γίνεται ευκολότερος υπολογισμός των χρονικών αποστάσεων. Η διαδικασία αυτή "γεμίσματος" του cleanup_array γίνεται στην execute του bolt. Όταν ολοκληρωθεί το τρέξιμο της τοπολογίας τότε καλείται η cleanup μέθοδος στην οποία το cleanup_array μεταβιβάζεται σε ένα array of objects (array of Trace), με το όνομα trace_final. Έπειτα με την συνάρτηση sort ταξινομούμε σε ascending order αυτό το array ως προς timestamp,ενώ σε δεύτερη φάση υποταξινομείται εσωτερικά ως προς tag (δηλαδή ταξινομούνται τα tag με ίδιο αναγνωριστικό ως προς μεταξύ τους, αφού έχουν ήδη ταξινομηθεί εσωτερικά ως προς τη χρονική στιγμή. Τέλος, για κάθε μοναδικό tag, θα εκτυπωθεί η διαδρομή του μέσα στον χώρο. Εκμεταλλευόμενοι την υλοποίηση μας στην clean_up μέθοδο και χρησιμοποιόντας την στην progress μέθοδο, μπορούμε να έχουμε οποιαδήποτε στιγμή επιθυμούμε, μέσω του σήματος KILL_SIG που έχουμε**

ορίσει, την πρόοδο ενός tag στον χώρο με τα μέχρι τώρα βήματα, χωρίς να έχει τελειώσει αυτό την κίνηση του/και χωρίς να απαιτείται κλείσιμο της τοπολογίας.

AvgTimePerTagBolt: **Το Bolt αυτό ακούει στο AvgTagStream από τον route (το id του αμέσως προηγούμενου bolt – RouteTracerBolt). Εδώ παίρνουμε το input από το RouteTrace αρχικοποιούμε ξανά ένα array of objects (array τύπου Trace) trace_final και το ταξινομούμε σε φθίνουσα σειρά ως προς τη χρονική στιγμή, και εσωτερικά ως προς τα tag id όπως προηγουμένως, ώστε να έχουμε τελικώς μέσα στη λίστα clusters των ιδίων tags μαζί, και τα όμοια tags να είναι ταξινομημένα αναμεταξύ τους ανάλογα τη χρονική στιγμή που τα παρατηρήσαμε.**

Έπειτα, για τα όμοια tags όπως τα έχουμε οργανωμένα στη σειρά και με φθίνουσα χρονική σειρά, βρίσκουμε τη χρονική διαφορά της κάθε εμφάνισης του tag από την επόμενη εμφάνιση του *ίδιου* tag. Αυτά τα time differences τα αθροίζουμε για κάθε χωριστό tag id , μέχρις ότου να συναντήσουμε διαφορετικό tag id στην επόμενη θέση της λίστας. Τότε βγάζουμε μέσω όρο των time differences διαιρώντας το σύνολο τους με τα hops που έγιναν για το συγκεκριμένο tag. Ύστερα κρατάμε αυτό το μέσο όρο για τη μετακίνηση του ενός tag και προχωράμε σε υπολογισμό για το επόμενο tag. Παράλληλα για να βρούμε σε ποιά περάσματα σημειώθηκαν οι μεγαλύτερες καθυστερήσεις (bottlenecks) κρατώντας τα time differences για κάθε hop του κάθε tag, έχοντας έτσι εποπτεία που καθυστερεί περισσότερο.Έπαναχρησιμοποιούμε (κάνοντας "abuse") την κλάση τύπου Trace και φτιάχνουμε έναν array από Trace αντικείμενα, όπου αντί για π.χ. antenna_code_1 έχουμε: :

1)το «hop» δηλαδή antenna_code_1 →antenna_code_2 ,

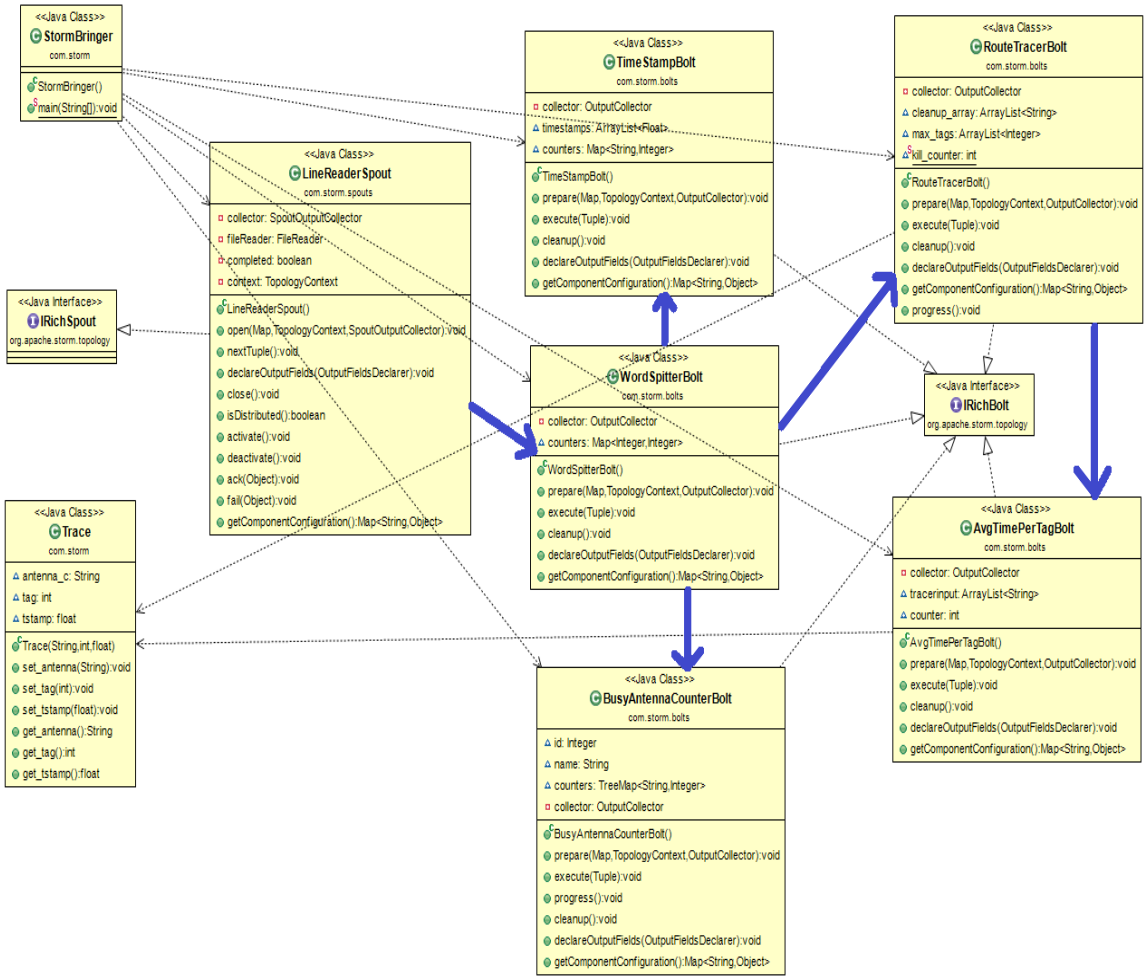
2)το tag όπως και πριν

3)το time difference, δηλαδή την καθυστέρηση που παρατηρήσαμε για να γίνει το hop από το antenna_1 στο antenna_2

Τέλος ταξινομούμε και εκτυπώνουμε με φθίνουσα σειρά τα bottlenecks για το κάθε συγκεκριμένο tag ξεχωριστά δείχνοντας τις καθυστερήσεις που σημειώθηκαν, και την «ακμή» του γραφήματος της τοπολογίας στην οποία σημειώθηκαν.

Παρακάτω παραθέτουμε τον χάρτη διεπικοινωνίας UML (πίνακας 5) των οντοτήτων του προγράμματος όπως εξήχθει από το Eclipse IDE.

Πίνακας 6: Το συνολικό UML που δείχνει την αρχιτεκτονική και την διεπικοινωνία εντός του συστήματος



4. ΣΥΜΠΕΡΑΣΜΑΤΑ

Στα πλαίσια της εργασίας που εκπονήθηκε, συγκεκριμένα όσον αφορά το προγραμματιστικό κομμάτι, σκοπός ήταν να υλοποιηθεί έναν κατανεμημένο σύστημα που να επεξεργάζεται δεδομένα από sensors, σημαντικού όγκου, και που βρίσκονται εν κινήσει καθώς και σε χρονικό διάστημα που ουσιαστικά είναι real-time. Για το λόγο αυτό χρησιμοποιήθηκε το ελεύθερου λογισμικού υπολογιστικό σύστημα Apache Storm. Με τη χρήση του επιτυγχάνεται η αξιόπιστη επεξεργασία streams δεδομένων σε πραγματικό χρόνο, όπως κάνει αντίστοιχα το Hadoop στο batch processing δεδομένων.

Το Storm έχει πολλές χρήσεις, όπως realtime analytics, online machine learning, continuous computation, distributed RPC, ETL και άλλα, ενώ είναι εξαιρετικό από άποψη ταχύτητας καθώς όπως αναφέρεται στο http://storm-project.net/ ένα benchmark έδειξε ταχύτητες επεξεργασίας πάνω από 1 εκατομμύριο tuples το δευτερόλεπτο ανά κόμβο. Είναι έναν σύστημα που κλιμακώνει εύκολα, είναι fault-tolerant, και παρέχει εγγυήσεις ότι όλα τα δεδομένα θα επεξεργαστούν, ενώ είναι εύκολο στο set-up και την χρήση, υποστηρίζοντας κάθε JVM προγραμματιστική γλώσσα όπως Java, Python, Ruby, ενώ με μία custom βιβλιοθήκη 100 γραμμών κώδικα μπορεί να υποστηρίξει και όποια άλλη προγραμματιστική γλώσσα. Τέλος το Storm ενσωματώνεται εξαιρετικά με τις υπάρχουσες τεχνολογίες queuing και βάσεων δεδομένων, με μία τοπολογία Storm να καταναλώνει streams δεδομένων, να τα επεξεργάζεται με περίπλοκους τρόπους και να ανασυντάσει τα streams μεταξύ κάθε σταδίου των υπολογισμών, ανάλογα των απαιτήσεων.

ΠΙΝΑΚΑΣ ΟΡΟΛΟΓΙΑΣ

Ξενόγλωσσος όρος	**Ελληνικός όρος**
Beacon	Σήμα/φάρος
Context-aware	Πλαίσιο επίγνωσης
Context information	Πληροφορίες εντός πλαίσιου
Spatial locations	Χωρικές θέσεις
Network locations	Δικτυακές τοποθεσίες
Trunk Circuits	Κυκλώματα κορμού
Airtime	Χρόνος ομιλίας/σύνδεσης ανά χρήστη
Traffic telematics	Τηλεματική οδικών μεταφορών
Fleet management	Διαχείριση στόλου
Path-loss	Απώλεια διαδρομής
Line of sight	Γραμμή οπτικής επαφής
Multipath propagation	Διάδωση πολλαπλών διαδρομών
Dedicated signaling	Ειδική σηματοδοσία
Terminal-Based	Βασισμένο στο τερματικό
Terminal-assisted	Υποβοηθούμενο από το τερματικό
Network-based	Βασισμένο στο δίκτυο
Reference positions	Θέσεις/σημεία αναφοράς
Matching process	Διαδικασία ταιριάσματος
Empirical approach	Εμπειρική προσέγγιση
Observed System Context	Πλαίσιο παρατηρούμενου συστήματος
Recasting	Αναμόρφωση (σε τύπους μεταβλητώ)

ΣΥΝΤΜΗΣΕΙΣ – ΑΡΚΤΙΚΟΛΕΞΑ – ΑΚΡΩΝΥΜΙΑ

LBS	Location Based Services
GPS	Global Positioning System
IEEE	Institute of Electrical and Electronics Engineers
WLAN	Wireless Local Area Network
BSA	Basic Service Area
BSS	Basic Service Set
ESS	Extended Service Set
RSS	Received Signal Strength
SNR	Signal To Noise
GSM	Global System for Mobiles
ECEF	Earth-centered, Earth-fixed
NNSS	Nearest Neighbor In Signal Space
RFID	Radio Frequency Identification
IoT	Internet Of Things
BAP	Battery-assisted Passive
RF	Radio frequency
API	Application Programming Interface
SS7	Signaling System 7
FPGA	Field Programmable Gate Arrays
FPU	Floating Point Unit
GPU	Graphics Processing Unit
DSP	Digital Signal Processing
QoS	Quality Of Service
ABS	Anti-lock braking system
CEP	Complex Event Processing

ΑΝΑΦΟΡΕΣ

[1] M. Chatzidakis, M. Loukeris, K. Gerakos and S. Hadjiefthymiades, "E-PreS: Monitoring and Evaluation of Natural Hazard Preparedness At School Community", 2016.Küpper, A. (2005). *Location-based services*. Chichester, England: John Wiley.

[2] A. Küpper, *Location-based services*. Chichester, England: John Wiley, 2005.

[3] G. Kul, T. Özyer and B. Tavli, "IEEE 802.11 WLAN based Real Time Indoor Positioning: Literature Survey and Experimental Investigations", *Procedia Computer Science*, vol. 34, pp. 157-164, 2014.Li, J. (2012). *Characterization of WLAN Location Fingerprinting Systems*.

[4] J. Li, "Characterization of WLAN Location Fingerprinting Systems", School Of Informatics, University Of Edinburgh, 2012.

[5] J. Tong-Seng Quah and L. Lim, "Location Cluster with Nearest Neightbors in Signal Space: An Implementation in Mobile Service Discovery and Tracking", 2011.

[6] B. Yuntian, W. Suqin, W. Hongren and K. Zhang, "Overview of RFID-Based Indoor Positioning Technology", School Of Mathematical and Geospatial Sciences, RMIT University, 2012.

[7] M. Bakhouya, *Proceedings of the 2nd international workshop on Agent-oriented software engineering challenges for ubiquitous and pervasive computing*. New York, NY: ACM, 2008, pp. 35-40.

[8] R. Antunes de Rocha, "Middleware for Location-based Services", Laboratory for Advanced Collaboration, Pontificia Universidade Catolica de Rio de Janeiro, 2004.

[9] "Stream processing", *En.wikipedia.org*, 2016. [Online]. Available: https://en.wikipedia.org/wiki/Stream_processing. [Accessed: 05- Oct- 2016].

[10] "Real-Time Stream Processing as Game Changer in a Big Data World with Hadoop and Data Warehouse", *InfoQ*, 2016. [Online]. Available: https://www.infoq.com/articles/stream-processing-hadoop/. [Accessed: 10- Oct- 2016].

[11] C. Kolski and J. Vanderdonckt, *Computer-Aided Design of User Interfaces III*. Dordrecht: Springer Netherlands, 2002, pp. 13-33.

[12] J. Gama and M. Gaber, *Learning from data streams*. Berlin: Springer, 2007, pp. 25-39.

[13] "Real-time computing", *En.wikipedia.org*, 2016. [Online]. Available: https://en.wikipedia.org/wiki/Real-time_computing. [Accessed: 15- Oct- 2016].

[14] "Spark and Storm face new competition for real-time Hadoop processing", *InfoWorld*, 2016. [Online]. Available: http://www.infoworld.com/article/2932294/hadoop/spark-and-storm-watch-out-here-comes-project-apex-for-hadoop.html. [Accessed: 12- Oct- 2016].

[15] "Extract, Transform, and Load Big Data with Apache Hadoop", Intel, 2013.

[16] "Big Data, Fast Data, Smart Data", *WIRED*, 2016. [Online]. Available: https://www.wired.com/insights/2013/04/big-data-fast-data-smart-data/. [Accessed: 16- Oct- 2016].

[17] "Streaming Analytics and the Internet Of Things: Transportation and Logistics", 2015.

[18] P. Pugh-Jones, "Streaming Analytics & IoT", SAS FORUM, United Kingdom, 2015.

[19] "Stream Processing Everywhere – What to Use? | MapR", *Mapr.com*, 2016. [Online]. Available: https://www.mapr.com/blog/stream-processing-everywhere-what-use. [Accessed: 20- Oct- 2016].

[20] "Tutorial", *Storm.apache.org*, 2016. [Online]. Available: http://storm.apache.org/releases/current/Tutorial.html. [Accessed: 08- Oct- 2016].

[21] 2016. [Online]. Available: https://eclipse.org/ide/. [Accessed: 16- Oct- 2016].

[22] J. Schiller and A. Voisard, *Location-based services*. San Francisco, CA: Morgan Kaufmann Publishers, 2004.