VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

NÁVRH SENZOROVÉ SÍTĚ PRO MONITORING OSOB A VĚCÍ V BUDOVĚ

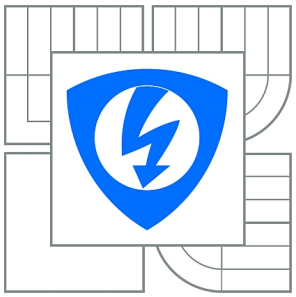
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE Bc. ZDENĚK ZÁDĚRA
AUTHOR

BRNO 2011

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF TELECOMMUNICATIONS

NÁVRH SENZOROVÉ SÍTĚ PRO MONITORING OSOB A VĚCÍ V BUDOVĚ

PROPOSAL OF WIRELESS SENSOR NETWORK FOR INDOOR MONITORING OF PEOPLE AND OBJECTS

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

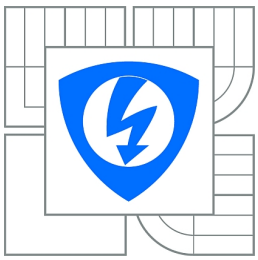
AUTOR PRÁCE — Bc. ZDENĚK ZÁDĚRA

AUTHOR

VEDOUCÍ PRÁCE — Ing. PATRIK MORÁVEK

SUPERVISOR

BRNO 2011



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Diplomová práce

magisterský navazující studijní obor
Telekomunikační a informační technika

Student:	Bc. Zdeněk Záděra	***ID:***	70256
Ročník:	2	***Akademický rok:***	2010/2011

NÁZEV TÉMATU:

Návrh senzorové sítě pro monitoring osob a věcí v budově

POKYNY PRO VYPRACOVÁNÍ:

Student se seznámí s problematikou monitoringu osob a věcí v budovách pomocí bezdrátových senzorových sítí. Cílem práce bude navrhnout senzorovou síť, která bude sloužit ke sledování pohybu osob a věcí v budově PA-118. Práce bude obsahovat rozbor požadavků aplikace, návrh lokalizační funkce a její implementace do senzorových uzlů.

DOPORUČENÁ LITERATURA:

[1] Born A.,Niemeyer F., Schwiede M., Bill R., On Distance Estimation based on Radio Propagation Models and Outlier Detection for Indoor Localization in Wireless Geosensor Networks, In International Conference on Indoor Positioning and indoor navigation - INIP2010, Zurich 2010

[2] Jia, T.,Buehrer R.M., Thompson, Cooperative Indoor localization, In International Conference on Indoor Positioning and indoor navigation - INIP2010, Zurich 2010

Termín zadání:	7.2.2011	***Termín odevzdání:***	26.5.2011

Vedoucí práce: Ing. Patrik Morávek

prof. Ing. Kamil Vrba, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

Abstrakt

Tato diplomová práce se zabývá návrhem bezdrátové senzorové sítě (WSN) pro monitorování pohybu osob a věcí v budově. Práce se věnuje problematice lokalizace a sledování objektů v senzorových sítích a implementaci lokalizačních algoritmů do senzorových uzlů. Dále obsahuje popis požadavků na senzorovou síť. Tyto požadavky pak tvoří základ pro samotný návrh. Hardwarová část sítě je tvořena senzorovými uzly IRIS firmy Crossbow. Práce obsahuje popis vlastností těchto uzlů. Další část práce tvoří návrh propagačního modelu a návrh lokalizačního algoritmu. V práci je též popsána komunikace v síti. Práce také obsahuje praktickou realizaci navržené sítě, lokalizačního systému a jeho otestování. V práci je přiloženo CD s plánem rozmístění senzorových uzlů v programu AutoCAD a se zdrojovými kódy vytvořených aplikací.

Klíčová slova

bezdátová senzorová síť, WSN, IRIS, Crossbow, ZigBee, IEEE 802.15.4

Abstract

This thesis describes the design of wireless sensor network (WSN) for monitoring of people and objects in a building. The work deals with issues of localization and tracking in sensor networks and algorithm implementation to sensor nodes. It also contains a description of the aplication requirements. These requirements form the basis for the proposal. The hardware part of the network consists of sensor nodes IRIS from Crossbow company. The work describes the properties of these nodes. Next part deals with of propagation model and design of the localization algorithm. The paper also describes the communication in the network. The thesis also includes a practical realization of the proposed network, the localization system and its testing. In the work is included a CD with the building schematic in AutoCAD and with source code of created applications.

Keywords

Wireless Sensor Network, WSN, IRIS, Crossbow, ZigBee, IEEE 802.15.4

ZÁDĚRA, Z. Návrh senzorové sítě pro monitoring osob a věcí v budově. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2011. 62s. Vedoucí diplomové práce Ing. Patrik Morávek.

Prohlášení

Prohlašuji, že svou diplomovou práci na téma Návrh senzorové sítě pro monitoring osob a věcí v budově jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení §11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení §152 trestního zákona č. 140/1961 Sb.

V Brně dne 22. května 2011 .....

Poděkování

Na tomto místě bych rád poděkoval svému vedoucímu Ing. Patriku Morávkovi za cenné připomínky a odborné rady, kterými přispěl k vypracování této diplomové práce.

Obsah

1 Úvod

Tato práce vznikla za účelem navrhnutí bezdrátovou senzorovou síť pro monitorování pohybu osob a věcí v budově. Bezdrátové senzorové sítě se začínají stále více používat díky své dostupnosti a velké oblasti použitelnosti.

Na světě vzniká spousta projektů na realizaci bezdrátových senzorových sítí za účelem monitorování pohybujících se objektů. Motivací pro tuto práci je například monitorování pacientů v nemocnici [19][10]. Každý pacient při příjmu do nemocnice dostane od personálu senzor například v podobě náramku na ruku. Pacienta přitom nijak neomezuje ani neobtěžuje. Tento senzor však plní hned dvě funkce. Informuje celou senzorovou síť o pozici pacienta a druhá jeho funkce je snímání zdravotního stavu pacienta. Pokud dojde u některého pacienta k zhoršení stavu, je o tom ihned informován personál nemocnice. Současně vidí na monitoru počítače, kde se tento pacient nachází. Tyto informace mohou zkrátit čas, který je velice důležitý.

Další reálné nasazení monitorovací senzorové sítě lze najít při záchranných akcích u hromadných nehod. Záchranné týmy často pracují u velkého počtu raněných na velké ploše. Při prvním zjišťování životních funkcí záchranáři umístí na raněné senzorové uzly, které budou informovat o jejich poloze i životních funkcích. Díky tomuto lze celou záchrannou akci řídit z jednoho místa a posílat záchranáře tam, kde jich bude opravdu zapotřebí. A zároveň mít kontrolu nad všemi raněnými a v případě potřeby k nim poslat pomoc.

Toto byly dva příklady praktického nasazení senzorových sítí pro monitoring pohybu osob. Dalšími příklady by například bylo využití v průmyslovém odvětví.

První kapitola této práce se zabývá obecným popisem bezdrátových senzorových sítí a jejich vlastnostmi. Uvádí nejčastěji používané topologie v bezdrátových senzorových sítích. Dále je zde uveden rozbor senzorového uzlu jako základního prvku sítě. Následně se seznámíme se standardem IEEE 802.15.4, na kterém je postavena komunikace v senzorových sítích. Tato kapitola dále přibližuje problematiku lokalizace a rozdělení jejích metod. Poslední částí kapitoly je popis sledování mobilních uzlů pohybujících se v určité oblasti.

Druhá kapitola se věnuje návrhu sítě. Rozebírá jednotlivé cíle, kterých má být při návrhu dosaženo. Před návrhem je třeba určit konkrétní požadavky na síť a způsob jejího použití. Tyto požadavky dále slouží k výběru hardwaru a architektury celého systému. V této kapitole je také popsán propagační model, který předpovídá chování sítě aniž bychom ji realizovali. Tato kapitola obsahuje také popis chování všech typů uzlů v síti pomocí vývojových diagramů.

Třetí kapitola popisuje praktickou realizaci senzorové sítě a systému lokalizace hledaného mobilního uzlu. V této kapitole budou popsány jednotlivé celky, které zpracovávají datové pakety ze senzorové sítě až po zobrazení v koncové aplikaci. Budou zde popsány implementace aplikací, potřebných pro fungování celého systému lokalizace. Dále zde bude uveden popis navržené databáze pro ukládání dat, získaných ze senzorové sítě.

Poslední kapitola obsahuje měření realizované sítě. Je v ní uveden způsob určování chybovosti určování pozice použitými algoritmy. Dále zde najdeme statistické vyhodnocení výsledků.

V závěru je uvedeno zhodnocení návrhu a k jakým výsledkům jsme při návrhu dospěli. Závěr také obsahuje popis úkolů, kterými lze do budoucna na práci navázat a postupně ji rozšířit.

2 Bezdrátové senzorové sítě

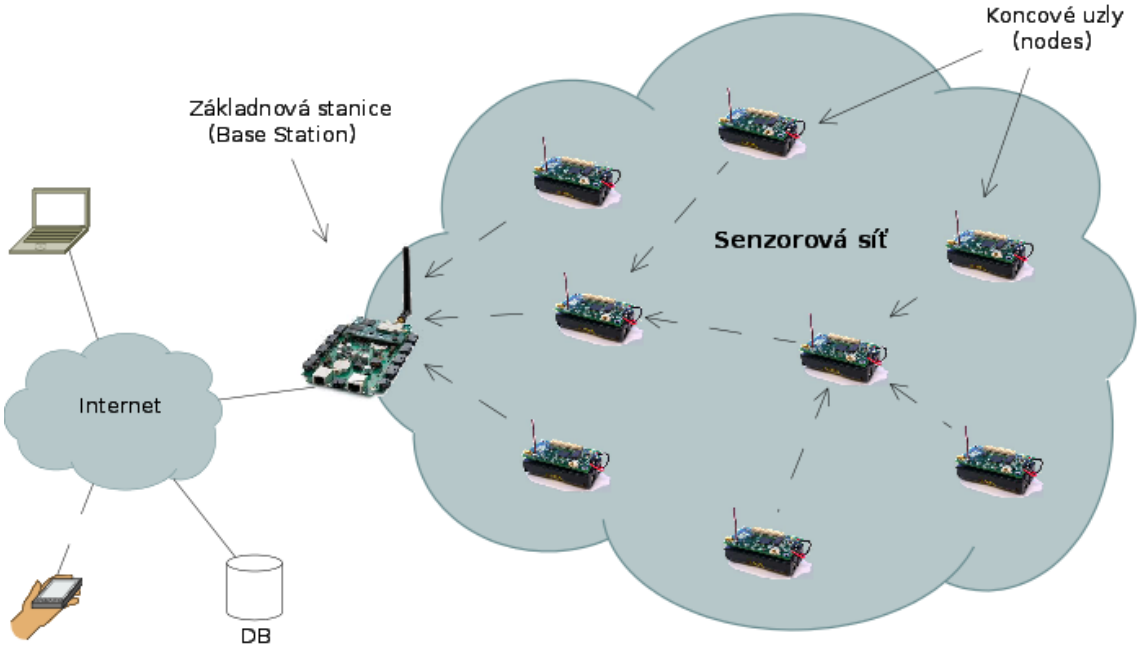
V této kapitole se seznámíme s pojmem bezdrátová senzorová síť (WSN). Tyto sítě jsou založeny na myšlence tzv. ad–hoc sítí, kdy každý uzel sítě je schopen komunikovat s kterýmkoli dalším uzlem, se kterým má přímě spojení. Tato komunikace není centrálně řízena. V počítačových sítích se takováto architektura nazývá peer–to–peer (rovný s rovným). Tato síť je velmi jednoduchá, rychlá a pro potřeby monitorování prostředí, což je hlavní využití WSN, je ideální.

2.1 Vlastnosti WSN

WSN vznikly za účelem monitorování, sledování či hlídání fyzikálních veličin (jako například teplota, zvuk, osvětlení, vlhkost, radiace...) v různých prostředích. Senzorová síť je složena z jednotlivých uzlů (angl. nodes), také se jim říká senzorové uzly nebo koncová zařízení. Jejich primární funkci je sběr dat pomocí vestavěného senzoru, který snímá monitorovanou fyzikální veličinu a poté tyto data posílat na výchozí bránu sítě. Každý uzel je schopen předávat zprávu okolním sousedům, kteří jsou v jeho rádiovém dosahu. Tímto způsobem jsou data přeposílána směrem na výchozí bránu celé této sítě, která se nazývá základnová stanice (angl. base station), která zpracovává data získaná ze senzorů a přeposílá je do vnějších sítí, například do sítě internet, či do datového úložiště. Strukturu bezdrátové senzorové sítě ukazuje Obrázek 1.

Senzorové sítě mají široké uplatnění od řízení domácností přes zjišťování polohy objektů, pohybujících se uvnitř sítě až po monitorování v rizikových oblastech, které by mohly být pro zdraví člověka nebezpečné. Níže jsou uvedenu příklady komerčních použití podle [6]:

1. **monitorování prostředí** (pohyb zvěře, záplavy, požáry),
2. **průmyslová diagnostika** (spotřebiče, továrny),
3. **hlídání infrastruktury** (energetické sítě, rozvod vody),
4. **vojenské účely** (zaměření vojáků na území),
5. **inteligentní budovy** (řízení vytápění, osvětlení).



Obrázek 1: Struktura bezdrátové senzorové sítě.

Hlavními prvky, které popisují senzorovou síť, jsou:

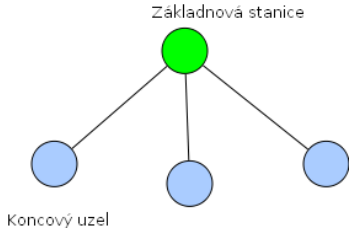
- **Síťová topologie** – graf konektivity, kde body jsou senzorové uzly a hrany jsou komunikační médium.
- **Senzor** – převádí fyzické veličiny (světlo,teplo,zvuk) na elektrické.
- **Senzorový uzel** – základní jednotka senzorové sítě. Skládá se ze senzoru, procesoru, paměti, bezdrátové komunikační jednotky a zdroje napětí.
- **Základnová stanice** – brána sítě, která komunikuje s okolními sítěmi.
- **Lokalizace** – zjištěni pozice uzlu v senzorové síti.
- **Tracking** – sledování, je to mechanizmus, při němž senzorová síť monitoruje polohu a vlastnosti objektů, které se nacházejí uvnitř této sítě.

Na následujících stránkách si postupně objasníme uvedené pojmy, které jsou důležité pro popis senzorové sítě i pro její návrh.

2.1.1 Topologie v sítích WSN

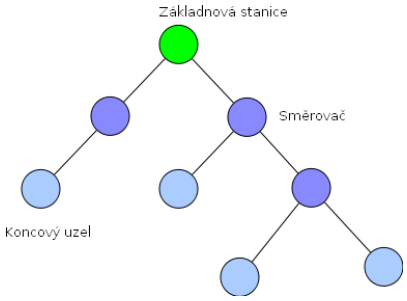
Topologie sítě je logické uspořádání jednotlivých uzlů, ze kterých se síť skládá, do hierarchické struktury. Toto uspořádání je obzvláště pro rozsáhlé sítě velmi přínosné, protože pomáhá definovat nejkratší cestu sítí od zdroje k příjemci.

Základní topologie, kterou je možné použít pro sítě s malým počtem uzlů, je topologie typu hvězda (Star). Každý koncový uzel vidí přímo na základnovou stanici a může data posílat přímo jí jako vidíme na Obrázku 2.



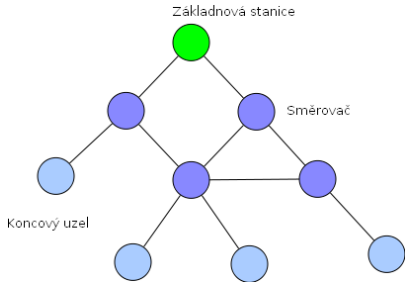
Obrázek 2: Topologie typu Star.

K pokrytí větší plochy je potřeba použít topologii stromovou (Tree), zobrazene na Obrázku 3. Tato komplexnější architektura určuje pro každý uzel jednu spojitou cestu, ale ostatní uzly jsou použity k přesměrování zprávy k základnové stanici. Tato topologie má ale nevýhodu v tom, že pokud přestane fungovat některý uzel blízko základnové stanice, celá větev pod ním nebude moci komunikovat se základnovou stanicí.



Obrázek 3: Topologie typu Tree.

Nevýhodu stromové struktury řeší síť typu mesh. Tato topologie přidává další možné cesty tak, že každý uzel má možnost posílat data více než jednou cestou. Takže, když nějaký uzel na cestě přestane fungovat, směrování se provádí přes uzly na alternativní cestě. Tato topologie však zvyšuje zpoždění sítě, protože data musí udělat více přeskoků než se dostanou do základnové stanice. Tento typ topologie zobrazuje Obrázek 4.



Obrázek 4: Topologie typu Mesh.

2.2 Senzorový uzel

Senzorový uzel je základní stavební jednotka senzorové sítě. Sbírá data a posílá do centrálního místa. Uzel se skládá z následujících částí.

- **Mikroprocesor**
  Mikroprocesor je hlavní částí Senzorového uzlu. Zpracovává data z pole senzorů, zpracovaná data odesílá rádiovému vysílači. Obsahuje protokolový stack, který definuje pravidla směrování paketů v síti. Podle těchto pravidel buď přeposílá pakety, nebo, pokud jsou určeny pro tento uzel, vyhodnocuje data uložená v datové části. Mikroprocesory jsou nenáročné na proudový odběr - většinu času jsou v spánkovém režimu, kdy odběr se pohybuje kolem desítek mikroampérů. Mikroprocesor ke své práci a výpočtům potřebuje externí paměť.

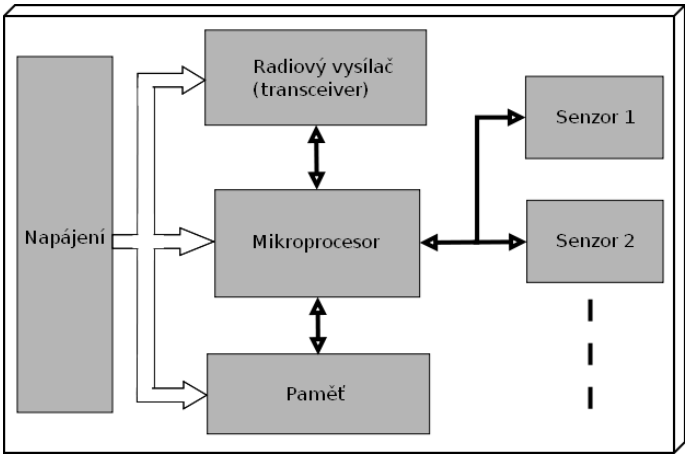
- **Externí paměť**
  Paměť slouží mikroprocesoru k ukládání mezivýsledků, dat, které nelze hned zpracovat či informace o uzlu a jeho poloze. Nejčastěji používané typy pamětí jsou typu: EEPROM - jsou to levné paměti, které jsou mazatelné elektrickým impulsem a FLASH - moderní typ pamětí, které mají vysokou kapacitu a nízkou spotřebu a dnes již i nízkou cenu.

- **Rádiový transceiver**
  Senzorové uzly musí obsahovat nějakou komunikační jednotku. Nejpoužívanější je rádiový vysílač-přijímač (tzv. transceiver), který vysílá signál na rádiových vlnách. Transceiver má nastavitelnou úroveň vysílaného signálu, což výrazně ovlivňuje spotřebu celého uzlu. Komunikační frekvence se používají v pásmu 868MHz, 900MHz a 2.4GHZ.

- **Pole senzorů**
  Senzor je elektronická součástka, která snímá fyzikální spojité veličiny a tyto veličiny převádí pomocí analogově-digitálního převodníku na digitální signály, které jsou dále zpracovány. Požadavky na senzory jsou: přesnost a nízká spotřeba, protože senzorové uzly jsou často vybaveny bateriovým napájením. Senzory se dělí podle fyzikálních veličin, které snímají:
  - teplotu (termistor),
  - světlo (fotorezistor),
  - vlhkost,
  - rychlost,

– zrychlení,
– tlak.

- **Napájení**
  Všechny předchozí části potřebují pro svoji práci napájení elektrickým napětím. Toto zprostředkovává napájecí část. Nejčastěji jsou jako zdroj napájení použity baterie z důvodu, aby uzel byl mobilní a při správně navržené komunikaci má životnost i několik let. Největší odběr má však vysílací část, která musí mít dostatečný vysílací výkon, aby mohla doručit zprávu okolním uzlům. Podle prostředí lze použít i alternativní zdroje napájení, jako například solární články. Pro zajištění dlouhé životnosti senzorového uzlu je třeba zajistit efektivní využití odběru energie z tohoto zdroje.



Obrázek 5: Důležitými součástmi bezdrátového uzlu jsou: mikroprocesor s externí pamětí, transceiver, senzory a napájení.

V následující tabulce je srovnání několika komerčních senzorových uzlů a jejich parametrů. Mezi nejdůležitější parametry uzlu patří použitý mikroprocesor, transceiver, interní a externí paměti.

Tabulka 1: Srovnání vlastností senzorových uzlů.

Senzor	**Mikroprocesor**	**Transceiver**	**Interní paměť**	**Externí paměť**
BTnode	Atmel ATmega128L	TI7 Chipcon CC1000	64 KB RAM	128 KB Flash 4 KB EEPROM
IMote	ARM9 7TDMI 12 MHz	Bluetooth	64 KB SRAM	512 KB
IMote 1.0	ARM 7TDMI 12-48 MHz	Bluetooth	64 KB SRAM	512 KB
Iris	Atmel ATmega1281	Atmel AT86RF230	8 KB RAM	128 KB
Mica	Atmel ATmega103	RFM TR1000	128 KB RAM	512 KB

2.2.1 Standard IEEE 802.15.4

Standard IEEE 802.15.4 [22] specifikuje fyzickou vrstvu a vrstvu přístupu na médium pro bezdrátové osobní sítě (WPAN). Struktura síťových vrstev vychází z modelu OSI. Důraz je kladen na velmi nízké náklady na komunikaci a nízkou spotřebu energie koncových uzlů. přenosovou rychlost může být až 250 kbit/s. IEEE 802.15.4 řeší prevenci kolizí pomocí metody CSMA/CA a integruje podporu pro zabezpečenou komunikaci.

- **Fyzická vrstva**
  Fyzická vrstva se stará o převod datového toku na elektrické signály a jejich vysílání pomocí rádiového transceiveru. Na této vrstvě se také provádí výběr výkonové úrovně signálu a výběr frekvenčního pásma. Na frekvenčním pásmu 868MHz se smí používat jen v Evropě a nabízí přenosový 1 kanál. Na frekvenčním pásmu 900MHz se smí používat jen v USA a nabízí 10 kanálu. A frekvenční pásmo se smí pro přenos používat po celém světě a nabízí 16 kanálů. Fyzická vrstvy využívám metodu rozprostřeného spektra (DSSS) k modulování datového toku na nosnou frekvenci.

- **MAC vrstva**
  Tato vrstva se zajišťuje o přenos MAC rámců pomocí fyzické vrstvy. Řídí přístup k fyzickému kanálu a stará se o signalizaci uvnitř sítě.

Obrázek 6: Rozdělení síťových vrstev, definovaných podle IEEE 802.25.4.

Na standardu IEEE 802.15.4 je založena technologie ZigBee, která definuje vrstvu síťovou a aplikační. ZigBee je určeno pro zařízení, která vyžadují dlouhou životnost baterií a nejsou náročná na rychlost přenosu dat. Technologie ZigBee definuje tři typy uzlů:

1. **PAN koordinátor** plní funkci směrovače pro přeposílání paketů k cíli,
2. **Plně funkční uzly** mohou se stát PAN koordinátorem a vytvářet sítě,
3. **Omezené funkční uzly** ke své funkci potřebují plně funkční uzly, samy o sobě nemohou v síti fungovat. Jejich výhoda je nízká spotřeba.

2.3 Lokalizace

V senzorových sítích je velice důležité znát polohu jednotlivých uzlů, aby bylo možné dále zpracovat veličinu, změřenou tímto senzorem a správně na ni zareagovat. Tímto úklem se zabývá lokalizace. Hlavním úkolem lokalizace je určení fyzické pozice uzlu v senzorové síti. V této podkapitole se budeme zabývat jen decentralizovanými lokalizačními algoritmy. Pozice uzlu může být určena absolutně (například GPS - Global Position System) nebo relativně vůči ostatním senzorům či části sítě.

U menších sítí na monitorování například teploty v bytě stačí uzlům přiřadit statickou pozici a nemusíme se starat o lokalizaci. V případě rozsáhlých a dynamicky se měnících sítí musíme však tuto otázku řešit. Pro určování pozice uzlu v síti pomocí lokalizace máme několik metod.

První z nich je možnost, že každý uzel je vybaven GPS systémem, uzel je tedy neustále informován o své poloze díky tomuto systému. Tato myšlenka je však velmi drahá, obzvláště u rozsáhlých sítí a také odběr GPS přijímače by v případě napájení uzlu z baterií způsobil velmi rychlé ochromení sítě. GPS přijímače navíc nemohou být použity uvnitř budov, kde je zastíněn jejich signál a ukazují pozici nepřesně.

Dále se používají algoritmy, které počítají pozici uzlu ze znalosti polohy sousedů. V tomto případě rozlišujeme uzly, které znají svoji absolutní polohu (tzv. kotevní body) a uzly, které jsou schopny určit svou vlastní polohu relativně ke kotevním bodům. Metoda kotevních bodů velmi usnadňuje proces lokalizace, neboť uzly, které pomocí některého z lokalizačních algoritmů vypočítají svoji relativní polohu vůči kotevním bodům, mohou získanou relativní polohu převést na absolutní polohu a tuto poskytnout dalším uzlům, které se snaží lokalizovat. Kotevní uzly jsou tedy pro lokalizační algoritmy nezbytnou součástí, avšak přinášejí další výdaje na lokalizační hardware. Lokalizační algoritmy se rozdělují do dvou skupin:

1. **range-based** a
2. **range-free.**

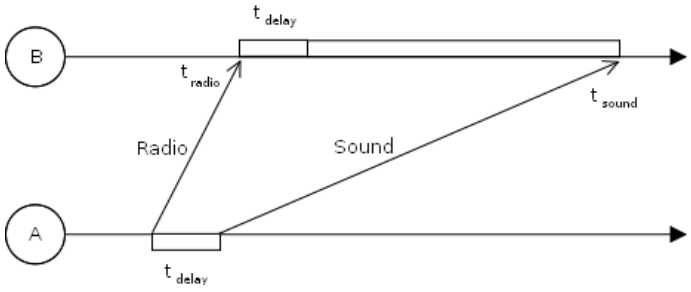
2.3.1 Algoritmy range-based

Při určování polohy je třeba změřit vzdálenost od kotevních uzlů. Tuto vzdálenost je možno měřit několika způsoby:

- **ze zpoždění signálu** - využívá techniku TOA (Time of Arrival), která měří dobu přenosu paketu od vysílače k příjemci. Tato technika je však drahá a limitované hardwarem. Použitelnější technika, nazvaná TDOA (Time Different of Arrival) [8] využívá k zjištění vzdálenosti rozdílné doby přenosu pomocí rádiových vln a pomocí zvukových vln. Princip je takový, že uzel A vyšle signál k uzlu B pomocí rádiových vln a po určitém čase pomocí zvukových vln jako je vidět na Obrázku 7. Uzel B přijme oba signály a zjistí čas, který uplynul mezi jejich přijetím. Ze získaného času lze vypočítat vzdálenost uzlu pomocí rovnice [8]

$$d = (s_{radio} - s_{sound}) \cdot (t_{sound} - t_{radio} - t_{delay}), \tag{1}$$

  kde s_{radio} a s_{sound} jsou rychlosti šíření daného signálu, t_{sound} a t_{radio} jsou časy přijetí zvukového a rádiového signálu a t_{delay} je doba o kterou je zpožděno vyslání zvukového signálu od rádiového.
  TDOA metoda dává dobré výsledky v prostorách, kde nejsou žádné překážky a nedochází k odrazům zvukových vln. To je nevýhoda při nasazování v místnostech kde se pohybují lidé. Další nevýhoda je použití přídavného hardwaru.



Obrázek 7: Metoda TDOA pro určení vzdálenosti uzlů.

- **z útlumu signálu** - technika, která měří hodnoru RSSI (Received Signal Strength Indication), která nese informaci o síle přijatého signálu. Je to nejjednodušší metoda určení vzdálenosti a často používaná vzhledem ke zkutečnosti že informace RSSI je dostupná v každém rádiovém přijímači a není tedy třeba přídavný hardware. Jednoduchost metody je ale vykoupena nepřesností a citlivostí na jakékoliv předměty v cestě přenosu.

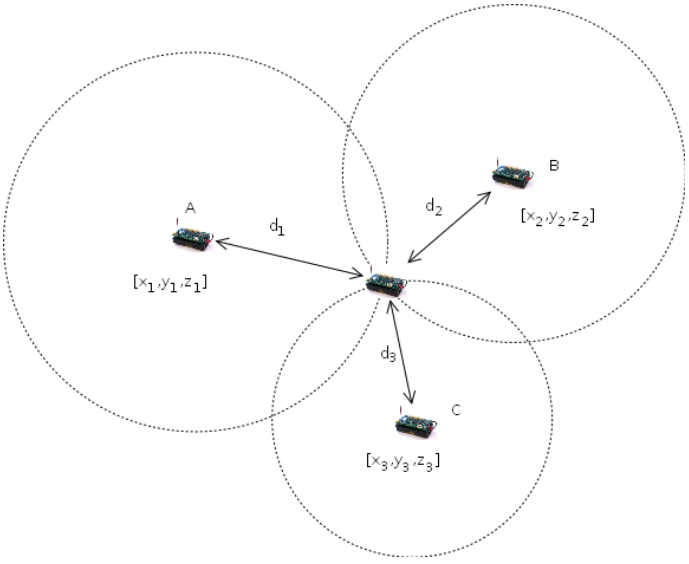
Jednoduchý algoritmus, který z takto získané vzdálenosti vypočítá souřadnice v trojrozměrném prostoru je založen na metodě nejmenších čtverců. Pro výpočet vzdálenosti bodu se souřadnicemi [x,z,y] od bodu se souřadnicemi $[x_1, y_1, z_1]$ lze použít rovnici

$$d^2 = (x1 - x)^2 + (y1 - y)^2 + (z1 - z)^2, \tag{2}$$

my však víme vzdálenost uzlů od sebe ale potřebuje určit souřadnice jednoho z nich. Na to však nestačí tuto rovnici použít jen jednou, ale potřebujeme tři tyto rovnice, do kterých dosadíme souřadnice a vzdálenosti od tří okolních uzlů, podle Obrázku 8.

$$(x1 - x)^2 + (y1 - y)^2 + (z1 - z)^2 = {d_1}^2, \tag{3}$$
$$(x2 - x)^2 + (y2 - y)^2 + (z2 - z)^2 = {d_2}^2, \tag{4}$$
$$(x3 - x)^2 + (y3 - y)^2 + (z3 - z)^2 = {d_3}^2. \tag{5}$$

Tyto rovnice můžeme řešit pomocí známých metod na výpočet soustavy rovnic (např. gaussova eliminační metoda, metoda LU rozkladu). Vektor neznámých [x,y,z] obsahuje hledané souřadnice uzlu.

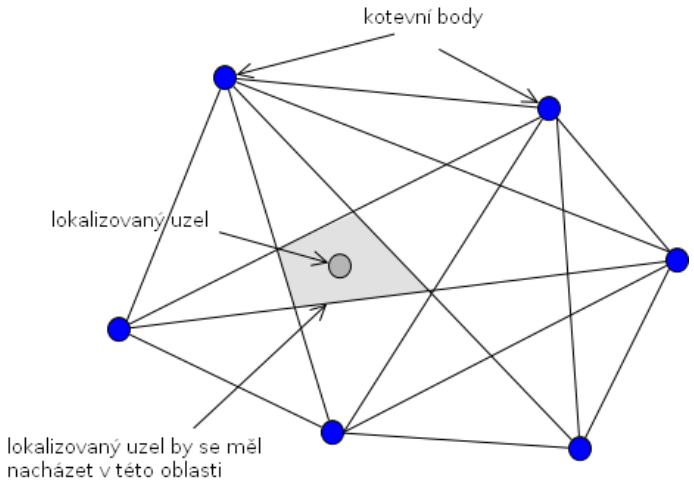


Obrázek 8: Lokalizační algoritmy typu range-based potřebují k vypočítání pozice uzlu vědět vzdálenost od okolních uzlů.

2.3.2 Algoritmy range-free

Tyto algoritmy nepotřebují měřit vzdálenost od kotevních uzlů. Pro odhad pozice je však třeba vědět: lokalizační rozsah sousedů a které uzly jsou v dosahu. Řadí se sem metody:

- **DV-HOP**, kdy každý kotevní uzel vyšle paket o své pozici a při každém směrování uzly se zvýší počítadlo v tomto paketu o 1. Každý uzel si tak spočítá počet skoků ke každému kotevnímu bodu a vytvoří si z nich tabulku.
- **APIT (Approximated Point In Triangle)** tato technika rozděluje síť na malé části, konkrétně trojúhelníky mezi kotevními body a poté se provádí test, zda lokalizovaný uzel se nachází uvnitř trojúhelníku.



Obrázek 9: Range-free metoda APIT pro určení pozice uzlu.

2.4 Tracking

Sledování a monitorování veličin je další z hlavních úkolu bezdrátové senzorové sítě. Sledování na úrovni objektů může být v praxi užitečné například pro sledování pozice dítěte, sledování pohybu pacienta v nemocnici či sledování pohybu zvířat v přírodě. V tomto případě je výhodné určovat absolutní lokaci sledovaných uzlů. Pokud se pohybuje uvnitř sítě nějaký uzel a vysílá rádiový signál, lze tento signál měřit a určit jeho polohu relativně k statickým uzlům v síti. Jak vidíme, sledování pohybujících se uzlů je úzce spjato s lokalizací a můžeme využít některý z lokalizačních algoritmů, uvedených výše.

Některé projekty sledování a lokalizace objektů ve vnitřních prostorách využívají podle [] ke zjišťování pozice infračervené (IR) bezdrátové sítě. Tato technika je limitována použitelným dosahem, avšak je používána a může mít své výhody.

V této podkapitole se budeme zabývat systémem RADAR, který je používaný právě pro sledování osob a věcí v budovách.

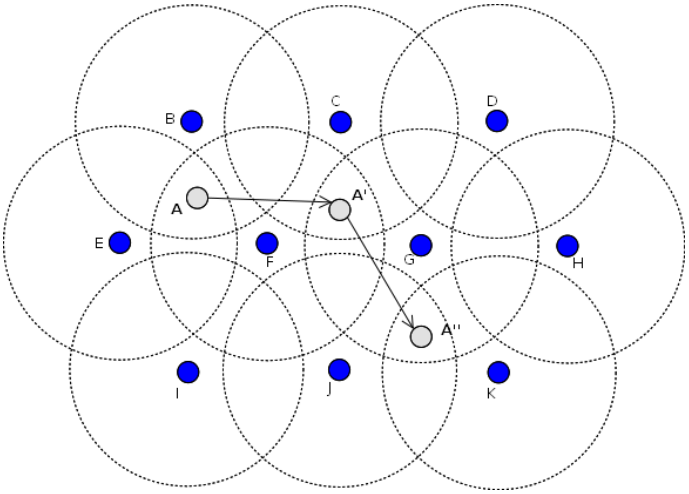
Lokační systém RADAR

Systém RADAR, používaný pro lokalizaci a sledování osob, pracuje v oblasti rádiových vln. Klíčová použití tohoto systému je sledování lokace objektu či osoby uvnitř senzorové sítě. RADAR operuje s hodnotou RSSI od několika uzlů k zjištění pozice osoby pomocí triangulace.

Princip je založen na umístění kotevních uzlů tak, že jejich rádiové dosahy se navzájem překrývají. Když se mobilní uzel pohybuje uvnitř sítě, může vyslat dotaz okolním uzlům, ty mu odpoví a pokud je v jejich rádiovém dosahu, tak signál příjme a vytvoří si odhad vlastní souřadnice. Na Obrázku 10 můžeme tento vidět metodu v praxi. Máme síť, kde se nachází pevné uzly se statickou pozicí (označené B až K) a v této síti se pohybuje mobilní uzel A. Uzel A se v počáteční

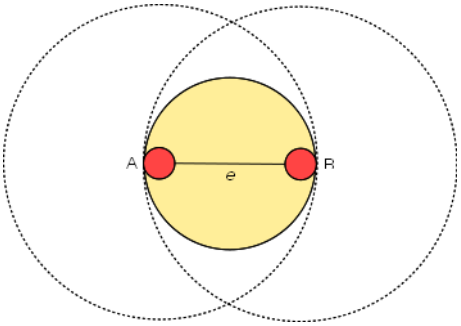
fázi nachází v rádiovém dosahu uzlů B, E a F, může tedy určit odhad své vlastní pozice na oblast, kde se rádiové dosahy těchto uzlů překrývají. Uzel A se nyní začne pohybovat, dostane se na pozici A' a po dotazu mu přijde odpověď od uzlů C, F a G, že se nachází v jejich rádiovém dosahu. Svou pozici tedy opět odhadne v oblasti, kde se rádiové dosahy překrývají. Poté se uzel dostane na pozici A" a opět si určí pozici v oblastí překrytí rádiových dosahů uzlů G, J, K.

Pozice není přesně určena absolutními souřadnicemi v souřadném systému, ale odhadem oblasti, kde se daný uzel nachází. Je třena zvážit, k jaké aplikaci se sledování mobilního uzlu použije a zda stačí přibližný odhad pozice, nebo zda je potřeba použít další přídavný hardware k určení přesné pozice.



Obrázek 10: Princip sledování pozice uzlu uvnitř senzorové sítě.

Další systém, popsaný v [9] je postaven na protokolu, který se stará o sledování pozice a pohybu mobilního uzlu. Celá sít je rozdělena do tzv clusterů. Pro ušetření energie tento systém používá předpovědi pozice uzlu, kterou si poté ověří zasláním dotazu. Dále je ušetřena energie tím, že v daném clusteru jsou některé uzly aktivní pro sběr informací o poloze uzlu a jiné jsou v sleep módu.



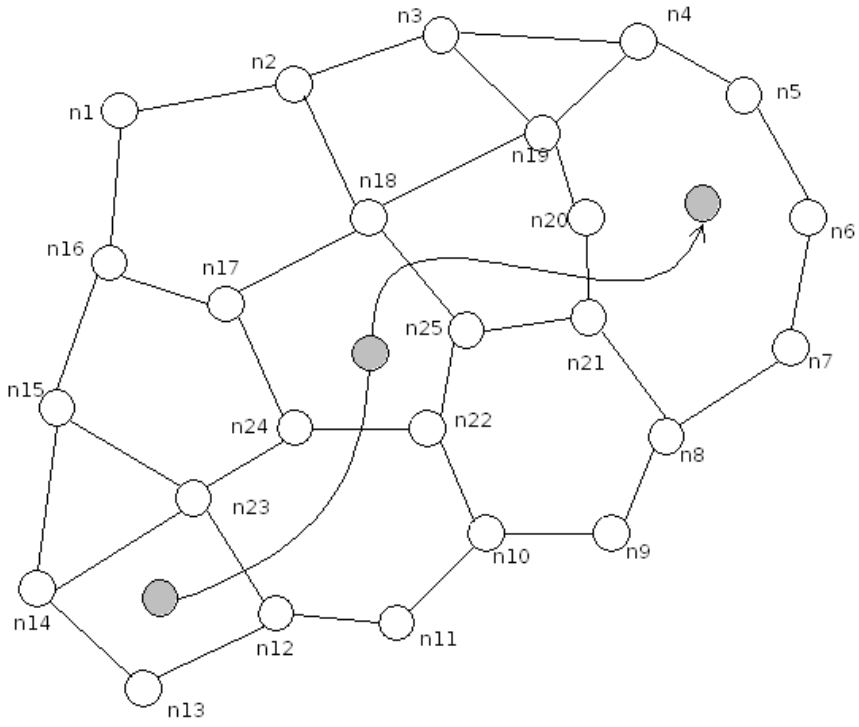
Obrázek 11: Urcení hrany Gabriel edge. Obrázek prevzat z [9]

K určení pozice používá tzv. Gabriel Graph [9], uvedený na Obrázku 11. Každý uzel zjistí své sousedy, jež má v rádiovém dosahu. Pokud uzel A je sousedem uzlu B, úsečka e je nazývána Gabriel edge. Takto můžeme postupně určit všechny spolu sousedící uzly v grafu a vytvořit planární graf.

V tomto grafu jsou definované jednotlivé spojité oblasti (clustery), definované vždy sousedícími uzly, které mají mezi sebou Gabriel edge.

Pokud některý z uzlů detekuje mobilní uzel, pošle informace o tomto uzlu postupně všem svým nejbližším sousedům. Tyto uzly postupně jak jsou probouzeny zjišťují přítomnost daného mobilního uzlu. Pokud jeho přítomnost nezjistí po stanovenou dobu, jsou opět uvedeny do stavu spánku. Podle uzlů, které detekují mobilní uzel ve svém okolí, můžeme zjistit, ve kterém kvadrantu se mobilní uzel nachází.

V tomto systému jsou používány určité protokoly k jednotlivým úkonům při sledování mobilního uzlu. Protokol request definuje tvar zprávy pro požadavek o pozici sledovaného uzlu. Na tento požadavek je zdroje odeslán paket typu reply. Po detekci mobilního uzlu, rozesílá detekující uzel paket typu wakeup všem svým sousedum. V neposlední rade je definován formát paketu typu deletion, který je vysílán uzlům, pokud se mobilní uzel vzdálí z oblasti, kterou obklopují.



Obrázek 12: Sledování mobilního uzlu v planárním grafu, reprezentujícím sít.

Velká výhoda tohoto systému je, že v jednu chvíli k určení pozice mobilního uzlu potřebuje vždy jen nejbližší uzel a jeho sousedy. Ostatní uzly mohou být ve stavu spánku a šetřit energii.

3 Návrh sítě pro monitorování osob a věcí

V této kapitole se budeme zabývat návrhem senzorové sítě. V podkapitolách budeme rozebírat postupně kroky, které je třeba při návrhu uvažovat.

Na začátku si musíme určit cíle, kterých chceme při návrhu sítě dosáhnout. Vlastnosti navrhované sítě závisí na prostředí, ve kterém bude použita. Proto musíme rozebrat požadavky prostředí. Bude třeba vytvořit propagační model, který bude popisovat způsob šíření signálu v předpokládaném prostoru a z tohoto je možno určit skutečné pokrytí signálem. Dalším cílem při návrhu je vybrat hardware, který bude použit při realizaci sítě. V příslušné kapitole si popíšeme hardware, který bude použit a jeho vlastnosti, které nás zajímají při návrhu budoucí sítě. Dále je potřeba navrhnout lokalizační algoritmy, které budou použity k monitorování a v neposlední řadě je také třeba určit architekturu sítě. Jednotlivými body se zabývají následující podkapitoly.

3.1 Cíle návrhu

Cílem této práce je navrhnout bezdrátovou senzorovou síť, která bude monitorovat pohyb osob a věcí v budově.

Tato síť by měla pomoci například při sledování návštěv a jejich pohybu. Pomocí údajů, získaných jednotlivými koncovými uzly bude monitorovat polohu jednotlivých mobilních uzlů, které budou mít sledované osoby u sebe.

Výstupem tohoto návrhu bude teoretická mapa rozmístění uzlů a návrh lokalizačního algoritmu. Dále si při návrhu určíme způsob komunikace v síti. Tento výstup by měl obsahovat vše důležité k tomu, aby bylo podle něj možné realizovat senzorovou síť v daném prostředí.

Při monitorování pohybu osob a objektů musíme mít dánu přesnost, s jakou chceme objekty lokalizovat. Naším cílem je určit pohyb osoby po budově a vyhodnocovat její polohu. Proto jsme si určili, že přesnost zjišťování pozice bude 5 metrů. Tato přesnost v sobě zahrnuje i odchylky při měření vzdáleností. Metoda, která bude použita pro zjišťování polohy objektů bude centralizovaná. To znamená že samotná lokalizace bude počítána pro všechny uzly v jednom centrálním místě.

Síť bude navržena pro nasazení do třetího patra budovy PA-118. Po realizovaní tohoto návrhu bude síť sloužit pro monitorování pohybu osob a věcí v daném patře. Poté se počítá s rozšířením sítě i do dalších podlaží budovy. Do budoucna se také počítá s přidáním dalších služeb, například monitorování teploty, či osvětlení.

3.2 Požadavky

Při budování senzorové sítě je potřeba brát v úvahu dodržování předem definovaných parametrů, které jsou od sítě očekávány. Senzorové sítě mají velkou škálu využití v mnoha oborech. Všechny tyto sítě mají však společné klíčové základní požadavky, které můžeme najít v literatuře [4]. Při návrhu sítě pro monitorování objektů uvnitř budovy potřebujeme zajistit následující požadavky:

1. **Přesnost vyhodnocení pozice**, na jakém místě se objekt nachází. Tato přesnost nesmí překročit hodnotu 5 metrů,
2. **Dostupnost informací o poloze objektů** pro externí aplikace,
3. **Jednoznačné určení pozice** ze získaných lokalizačních údajů.
4. **Kompatibilita lokalizačních údajů** a možnost integrování dalších údajů (např. zdravotní stav sledované osoby ap.),
5. **Práce v reálném čase**,
6. **Robustnost systému**, přesné určení pozice za různých teplot či při výpadku některých uzlů,
7. **Malá velikost mobilních uzlů**, aby je mohla osoba nosit u sebe,
8. **Efektivní využití energie** v mobilních uzlech. Mobilní uzel by měl na baterii vydržet alespoň 24 hodin.

Lokalizace objektů pomocí globálního poziční systému GPS je velmi rozšířené. Dnes již s přesností na jednotky metrů kdekoliv na světe. Tento systém však nelze použít v budovách kvůli zastínění signálu. Je třeba použít nebo vytvořit systém, který bude založený na shromažďování údajů, zjištěných lokálně. Tento systém by měl být nezávislý na konkrétním hardwaru, aby jej bylo možno používat kdekoliv. Použitý algoritmus by měl také umět odhadnout chybu měření.

V této fázi je třeba zhodnotit, zda je pro nás důležitá rychlost, či přesnost zjišťování pozice. Tato síť se neřadí mezi sítě se sběrem kritických dat, proto dáme přednost přesnosti sledované pozice než rychlosti jejího zjištění. Perioda snímání pozice uzlu se odvíjí od rychlosti sledovaného uzlu. V navrhované síti předpokládáme, že sledované uzly se budou pohybovat rychlostí chůze člověka (kolem 1,5 m/s). Pokud chceme dodržet přesnost 5 metrů, je třeba snímat polohu s periodou $\leq 3,3s$. Musíme však uvažovat i nepřesnost lokalizační metody, proto periodu sledování pozice zvolíme 2 sekundy.

3.3 Výběr hardwaru

3.3.1 Senzorový uzel

Pro výběr hardwaru je třeba určit princip, jakým bude získávat data ze svého okolí. My jsme se rozhodli pro komunikaci přes rádiové vlny. Na tomto poli máme na výběr z několika produktů, je třeba si tedy určit kritéria pro výběr:

- **Prostředí**, hardware musíme volit podle prostředí, ve kterém bude použit. V našem případě to bude použití uvnitř budovy, což rapidně snižuje jeho rádiový dosah a je s tím třeba počítat,
- **Přesnost**, tato vlastnost záleží kromě hardwaru také na použitém algoritmu. Čím více uzlů je použito pro lokalizaci, tím více se tato nepřesnost šíří,
- **Mobilita**, aby bylo možné uzly použít jako mobilní a pomocí nich sledovat pohyb osob, je nutné, aby neměly příliš velké rozměry a váhu.
- **Rádiové pásmo**, je vhodné vybrat přenosové pásmo takové, abychom omezili na minimum rušení komunikace jinými zařízeními,
- **Paměť**, pro ukládání naměřených dat i pro nahrání programového kódu je potřeba aby v senzorovém uzlu byla dostatečná kapacita paměti,
- **Spotřeba**, efektivní využití energie je velice důležité při výběru senzorového uzlu,
- **Cena**, při výběru samozřejmě hraje roli i cena hardwaru a jeho dostupnost.

Pro užší výběr jsme zvolili senzorové uzly, uvedené v následující tabulce. Jsou zde uvedeny tři senzorové uzly s jejich porovnávanými vlastnostmi.

Tabulka 2: Porovnání nabízených senzorových uzlů.

Senzor	Mikroprocesor	Transceiver	Paměť	Spotřeba
BTnode	Atmel ATmega128L	Chipcon CC1000	64 KB RAM 128 KB Flash	31 mA[1)]/3 mA[2)]
Iris	Atmel ATmega1281	Atmel AT86RF230 250 kbit/s	8 KB RAM 128 KB Flash	25 mA[1)]/8 μA[2)]
Mica	Atmel ATmega103	RFM TR1000 50 kbit/s	128 KB RAM 512 KB Flash	18 mA[1)]/6 μA[2)]

[1)] Proudová spotřeba v aktivním režimu při vysílání signálu s maximální výkonem.
[2)] Proudová spotřeba v režimu spánku.

Jako senzorový uzel byl vybrán produkt IRIS od firmy Crossbow. Tento uzel má podle [24] dosah v budově minimálně 50 metrů. Jeho integrovaný RF transceiver podporuje standard IEEE 802.15.4 a pracuje v ISM pásmu od 2,4 do 2,48 GHz. Na fyzické vrstvě je použita technologie rozprostřeného spektra, která je odolná vůči rušení a podporuje bezpečnost dat. Přenosová rychlost dat u tohoto uzlu dosahuje až 250 kdps.

Tento uzel je podporován platformou MoteWorks. Tato platforma je založena na open–source operačním systému TinyOS. K tomu nabízí realizaci například ad–hoc sítí, programování za běhu (over–the–air–programming) a aplikaci pro serverovou část.

Výpočetní jednotku tvoří mikroprocesor Atmel ATmega1281. V tomto procesoru je nahrána platforma MoteWorks, která podporuje volání definovaných příkazů pro snadné programování aplikací. Pomocí konektorů je možno připojovat k tomuto uzlu široký výběr senzorů, či pomocí sériových rozhraní připojovat vlastní externí hardware.

Všechny tyto vlastnosti jsou shrnuty v tabulce 3.

Tabulka 3: Hlavní vlastnosti senzorovéhu uzlu IRIS, převzaté z [24].

Senzorový uzel IRIS (XM2110CA)		
Výpočetní jednotka		
Paměť interní	128KB Flash, 512KB Serial Flash	
Paměť externí	8KB RAM, 4KB EEPROM	
Komunikační rozhraní	UART,I2C,SPI	
Analogově–digitální převodník	10 bit ADC	8 kanálů
Proudový odběr	8 mA	Aktivní režim
	8 μA	Sleep režim
RF Transceiver		
Frekvenční pásmo	2405 MHz – 2480 MHz	programovatelné kanály
Přenosová rychlost	250 kbps	
Úroveň vysílaného signálu	-17 dBm – 3 dBm	
Rádiový dosah v budově	$\geq$ 50 m	
Odběr transceiveru	10 mA	při -17 dBm
	17 mA	při 3 dBm
Elektromechanické		
Napájení	2x AA baterie	
Velikost	58 x 32 x 7 mm	
Váha	18 gramů	

3.3.2 Základnová stanice

Jako základnová stanice bude použit rovněž senzorový uzel IRIS, který bude připojen do programovací brány MIB520. Tato brána vytvoří přes rozhraní USB spojení s počítačem, který bude ukládat a dále zpracovávat získaná data.

3.3.3 Centrální databáze

Jako centrální úložiště dat bude sloužit počítač, na kterém je nainstalovaný databázový systém. Tento počítač bude sloužit jako databázový server, který bude shromažďovat všechna naměřená data ze senzorové sítě. Odtud budou data přenášena do spolupracující aplikace, která je vytvářena v diplomové práci [15] souběžně s touto prací. Tato aplikace slouží k vizualizaci získaných dat,

v našem případě bude zobrazovat získané polohy sledovaných uzlů a díky tomu bude možné sledovat jejich pohyb po budově.

3.4 Propagační model

V této části si vytvoříme tzv. propagační (rádiový) model [23]. Je to teoretický model, který popisuje způsob šíření rádiového signálu. Cílem propagačního modelu je určit výkonnost šíření signálu v závislosti na prostředí a frekvenci. Pro plánování komunikační sítě je potřeba vědět, jakou sílu bude mít signál v určité vzdálenosti a jaký rádiový dosah budou mít jednotlivé uzly. Díky tomuto modelu můžeme předpokládat určité chování v daných podmínkách a využívat energie senzorových uzlů efektivně.

Ve vnitřních prostorách často vzniká spoustu elektromagnetického záření kolem zařízení. Také je používání bezdrátové technologie WiFi vytváří rádiové zarušení těchto prostor. To vše ruší komunikaci v bezdrátové síti. Je třeba si tento vliv na komunikační signál popsat matematickým modelem. Dále zde dochází k rušení signálu kvůli odraženým signálům, které interferují s původním signálem a tím ho znehodnocují. Tomuto jevu se říká multipath. Další zeslabení signálu je způsobeno zdmi, dveřmi, nábytkem i pohybem osob. Všechny tyto překážky můžou znehodnotit signál. Musíme je tedy zahrnout do propagačního modelu.

Pro vytváření propagačního modelu se používají různé deterministické modely. Jsou to site–specific a site–general. Používanější je model site–general, protože na rozdíl od druhého jmenovaného modelu nevyžaduje znalost vlastností budovy rozložení nábytku a použitých materiálech. Toto se může s časem měnit a bylo by třeba vytvářet vždy nový model.

Propagační model ITU site–general

Úroveň každého rádiového signálu, vycházejícího z vysílače se exponenciálně snižuje s přibývající vzdáleností od tohoto vysílače. Tento jev se nazývá ztráta ve volném prostoru (free space loss).

Pro frekvenční pásmo s vlnovou délkou λ lze velikost ztráty ve volném prostoru a ve vzdálenosti d od vysílače určit ze vztahu, dle [23]:

$$L_{total} = 20\log_{10}(f) + \eta\log_{10}(d) + Lf(n) - 28dB, \tag{6}$$

kde η je koeficient útlumu daného prostředí, f je frekvence v MHz, d je vzdálenost v metrech, $Lf(n)$ útlum způsobený podlahou a n je počet podlah, které se vyskytují v cestě signálu.

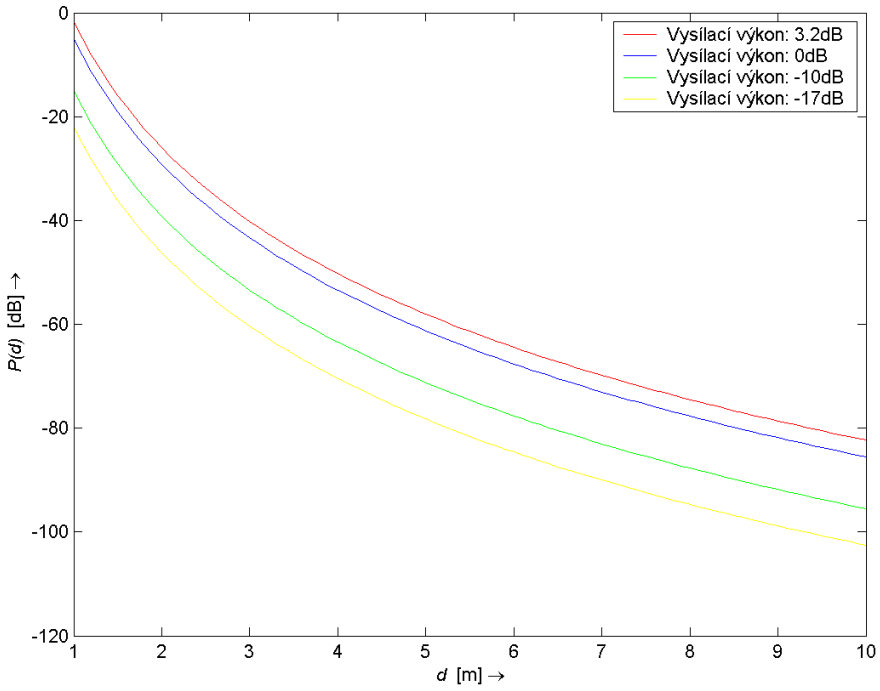
Pokud signál prochází překážkou, úroveň signálu se snižuje rychleji, než při šíření volným prostředím. Obecně lze rozdělit překážky do několika skupin. V literatuře [16] jsou uvedeny koeficienty η pro různé prostředí. Zde jsou vybrány některé z nich:

- $\eta = \mathbf{20}$, tento koeficient platí pro přímou viditelnost přijímače a vysílače.
- $\eta = \mathbf{30}$, tento koeficient odpovídá modelu kanceláře, kdy signál prochází jen vnitřními zdmi,
- $\eta = \mathbf{40}$, pokud signál prochází kolem rohů budovy a skrz obvodové zdi, je použit tento koeficient.

My použijeme koeficient $\eta = 35$, protože nejlépe vystihuje prostředí v jakém bude senzorovou síť používat.

Podle literatury [16] dále určíme neznámou v rovnici $Lf(n)$. Předpokládáme jeden odraz od podlahy, potom dostaneme hodnotu $Lf(n) = 15$ dB.

Máme tedy rovnici, která vyjadřuje velikost ztráty signálu v určité vzdálenosti. Pokud chceme vyjádřit výkon, který změříme v této vzdálenosti, odečteme tuto celkovou ztrátu od vysílacího výkonu. Tento výpočet použijeme, abychom si vytvořili lepší obrázek, jak se bude šířit signál v předpokládaných podmínkách. Obrázek 13 ukazuje velikost úrovně signálu v decibelech v různých vzdálenostech pro různé vysílací výkony. Tento obrázek byl vytvořen v prostředí MATLAB.



Obrázek 13: Grafická závislost výkonové úrovně signálu na vzdálenosti od zdroje.

Tato grafická závislost platí pro prostředí, které jsme si zvolili dosazením do předchozí rovnice. To znamená, že signál prochází zdmi a má v cestě nábytek a další překážky.

Dále si určíme zisk přenosu signálu na lince. Podle [5] lze zisk linky vyjádřit takto:

$$Zisk\ linky = EIRP - L_{total} + G_{Rx} - TH_{Rx}. \tag{7}$$

V této rovnici $EIRP$ znamená efektivní izotropní vysílací výkon v dBm, G_{Rx} je zisk přijímače a TH_{Rx} je minimální citlivost přijímače.

Neznámou EIRP v rovnici vypočítáme ze vztahu:

$$EIRP_{dB} = P_{TdB} + G_{TdB}. \tag{8}$$

Do rovnice (7) dosadíme hodnoty hardwaru, který jsme si vybrali. Tedy hodnoty senzorového uzlu IRIS. Samozřejmě, abychom dosáhli přenosu signálu, musíme mít zisk linky větší nebo roven 0. Vypočítáme největší dosažitelnou vzdálenost pro maximální vysílací výkon, tedy pro $P_{TdB} = 3.2$dB.

$$0 \geq 3,2dB + 2,5dBi - L_{total} + 2,5dBi - (-91dBm) \tag{9}$$

$$L_{total} \leq 99,2dB. \tag{10}$$

Tuto hodnotu dosadíme do rovnice (6) a vyjádříme jedinou neznámou, kterou je vzdálenost. Po výpočtu získáme vzdálenost $d \leq 14{,}79$m. Tímto jsme získali maximální dosažitelnou vzdálenost pro naše prostředí. My si pro rozmístění uzlů zvolíme rádiový dosah 10 metrů. Máme tedy rezervu a můžeme například snížit vysílací výkon a tím šetřit energii uzlů.

Rovnici 6 můžeme také vyjádřit v poměrném tvaru. V tomto případě potřebujeme v síti znát pozice aspoň dvou senzorových uzlů. Propagační model mezi těmito dvěma uzly pak můžeme brát jako referenční a podle něho upravovat propagační model pro ostatní uzly. V tomto případě by rovnice pro přijatý signál vypadala následovně:

$$Rssi(d) = Rssi(d_0) - 10\eta log(\frac{d}{d_0}), \tag{11}$$

kde η je stejný koeficient jako v předchozím případě, d_0 je předem známá vzdálenost a $Rssi(d_0)$ je síla signálu, změřená v této vzdálenosti.

Z uvedené rovnice vychází i následující vztah pro přepočet přijatého Rssi na vzdálenost:

$$d = d_0 \cdot \left(\frac{Rssi(d_0)}{Rssi}\right)^{\frac{1}{\eta}}. \tag{12}$$

Díky zjištěnému poklesu úrovně signálu na této referenční vzdálenosti můžeme dynamicky měnit propagační modely pro nejbližší uzly a započítávat tak do nich vzniklé útlumy v prostředí.

3.5 Energetický model

V této části si uvedeme energetický model sítě. Tento model popisuje spotřebu energie jednotlivými uzly a používá se pro určení jejich životnosti. Nejprve je třeba zjistit spotřebu energie jednotlivých uzlů. Pokud víme, jakou energii mohou poskytnout napájecí baterie, můžeme určit celkový čas, po který je možno tuto energii spotřebovávat.

Podle [21] se může uzel IRIS nacházet ve třech režimech. V těchto režimech je proudová spotřeba:

- **25 mA** v aktivním režimu a při vysílání signálu,
- **80** μ**A** v idle režimu, tedy když transceiver pouze naslouchá na rádiovém kanále. Pokud zjistí na kanále vysílání, teprve poté přejde do aktivního režimu.
- **8** μ**A** v režimu spánku, kdy transceiver je odpojen od napájení.

Dále budeme potřebovat napěťový rozsah, s kterým pracuje senzorový uzel. Tento rozsah je dán použitím dvou baterií typu AA a pohybuje se od 2,5V do 3,6V. Ve výpočtech budeme brát průměrné napětí U = 3V. K napájení budou použity alkalické baterie, jejichž energetická hodnota je E_{bat} =21600 Joule.

Pro komunikaci v síti budeme využívat Xmesh protokol, který je definovaný v literatuře [21]. Velikost zprávy v tomto protokolu je 55 bytů. K této datové části si připočteme ještě 6 bytů, které se přidají na linkové vrstvě, tedy velikost celého rámce bude 61B. Přenosová rychlost rádiového signálu je 250 kbps. Z těchto údajů lze spočítat, že doba přenosu jedné zprávy je $t_{tx} = 1,95ms$. Energie, spotřebovaná na přenos jedné zprávy bude $E_{tx} = 146\mu J$. Energie, spotřebovaná v režimu spánku po dobu 1 sekundy bude $E_{sleep} = 24\mu J$ a energie v idle režimu po dobu 1 sekundy bude $E_{idle} = 240\mu J$

Nyní si vytvoříme energetické modely pro jednotlivé typy uzlů, které budou použity v navržené síti. Začneme mobilnímm uzlem, který se každé 2 sekundy probudí z režimu spánku. Po probuzení pošle 10 zpráv a počká na odpověď po dobu 10ms. Po obdržení odpovědi opět přejde do režimu spánku. Spotřeba energie za tuto periodu bude:

$$E_{duty} = 2s \cdot E_{sleep} + 10 \cdot E_{tx} + 10ms \cdot E_{idle} + E_{tx} \tag{13}$$

$$E_{duty} = 48 \cdot 10^{-6} + 1,46 \cdot 10^{-3} + 2,40 \cdot 10^{-3} + 146 \cdot 10^{-6} = 1,51mJ, \tag{14}$$

a délka periody bude:

$$t_{duty} = 2s. + 10 \cdot 1,95ms + 10ms + 1,95ms = 2,041s. \tag{15}$$

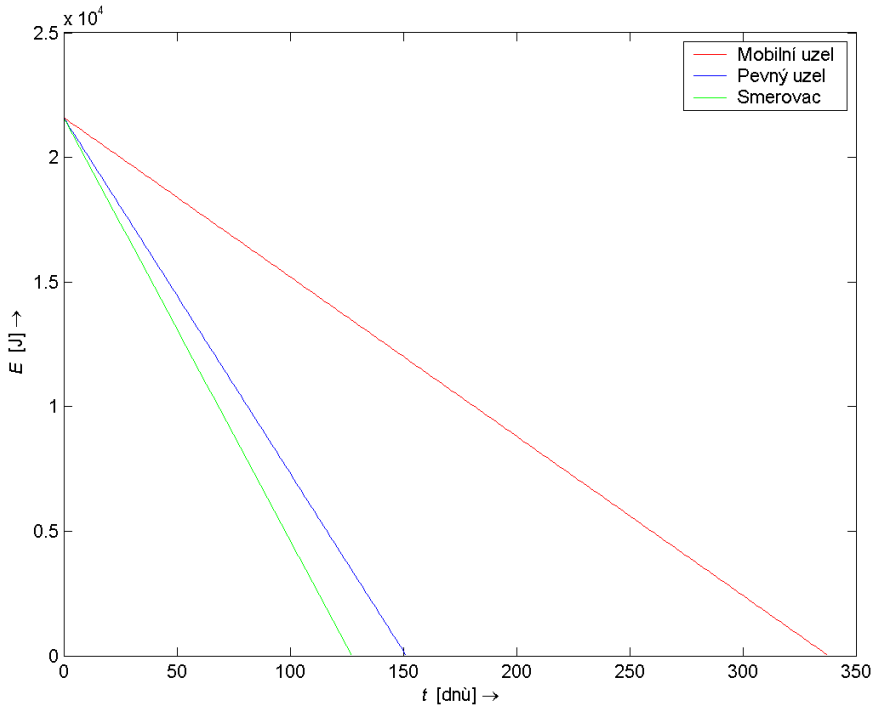
Nyní můžeme vypočítat celkovou dobu života mobilního uzlu. Energii baterie podělíme energií spotřebovanou v jedné periodě a poté vynásobíme délkou periody.

$$t_{mob} = \frac{E_{bat}}{E_{duty}} \cdot t_{duty} = \frac{21600}{1,51 \cdot 10^{-3}} \cdot 2,041 = 11 \; mesicu \; a \; 7 \; dni. \tag{16}$$

Stejným postupem dojdeme k výsledku pro kotevní uzel. Tento uzel stráví většinu času v idle režimu, kdy bude naslouchat na kanále. Při aktivní komunikaci se probudí a přijme signál. Poté zprávu pošle dále a zase přejde do režimu idle. Předpokládejme, že průměrně se bude v síti pohybovat v jednu chvíli 5 mobilních uzlů, které budou vysílat každé 2 sekundy informace o sobě. Pokud bychom rovnoměrně rozdělili tuto zátěž na všechny kotevní uzly, můžeme u každého kotevního uzlu předpokládat probuzení každou 1 sekundu. Spotřeba energie během jedné periody bude $E_{duty} = 2,23mJ$ a $t_{duty} = 1,023s$ Délka života pak bude: $t_{anchor} = 4$ měsíce a 8 dní.

Směrovač pak bude vykonávat stejnou práci jako kotevní uzel. U tohoto typu uzlu bude však navíc docházet k směrování zpráv od jiných uzlů, proto přidáme navíc energii, která bude odpovídat tomu, že tento uzel bude každou 1 sekundu vzbuzen a přepošle jednu zprávu. Pro dobu života uzlu typu směrovač pak dojdeme k hodnotě $t_{rout} = 3$ měsíce a 21 dní.

Na Obrázku 14 můžeme vidět průběh energie na bateriích jednotlivých uzlů. Tato grafická závislost je vytvořena v prostředí MATLAB. Z porovnání je vidět, jak se liší doba života mobilního uzlu, kotevního uzlu a směrovače.



Obrázek 14: Grafická závislost energie jednotlivých uzlů na čase.

Z vypočtených hodnot vidíme, že v uzlech by bylo několikrát do roka třeba vyměňovat baterie. Pokud to bude možné, bude výhodné kotevní uzly a směrovače napájet z elektrické sítě.

3.6 Rozmístění uzlů

Pilotní nasazení sítě se plánuje do třetího podlaží v budově PA-118. Toto umístění umožní monitorování pohybu osob a zkoumání vlivu rušení na tuto funkci. V příloze je nákres rozmístění uzlů v uvedeném patře. Rádiový dosah jednotlivých uzlů jsme určili na 10 metrů. Senzorové uzly jsou rozmístěny tak, aby se jednotlivé rádiové dosahy sousedních uzlů překrývaly. Druhá podmínka při umisťování uzlů byla, aby v každém místě se překrývaly rádiové dosahy minimálně tří uzlů. Tato podmínka je nutná pro provedení lokalizace mobilního uzlu, který se bude moci volně pohybovat po celém patře.

Jak již bylo řečeno, návrh se týká jen třetího patra. Do budoucna se však počítá s rozšířením sítě i do ostatních podlaží budovy.

3.7 Architektura sítě

Pod pojmem architektura sítě se rozumí její topologie, forma komunikace a služby, které síť nabízí. V následujících částech popíšeme návrh pro jednotlivé body.

3.7.1 Topologie

Použijeme topologii typu mesh, kdy každý uzel bude mít více možností k směrování datového paketu směrem k základnové stanici. Tato topologie zajišťuje, že síť bude provozuschopná i po výpadku jednoho z uzlů a po výměně nefunkčního uzlu bude síť opět plně schopna efektivního provozu.

Uzly sítě budou pracovat na standardu 802.15.4, jak bylo řečeno v části výběru hardware. Hierarchie jednotlivých typů uzlů bude následující. Signál z mobilních sledovaných uzlů bude snímán pomocí uzlů, které budou umístěny na kotevních místech podle mapy rozmístění. Tyto uzly budou data posílat směrem na směrovače, které budou přeposílat data v hierarchické struktuře výše směrem k základnové stanici. Celý tento proces je vidět na Obrázku 15. Uzly s funkcí směrování budou nejvíce vytížené vzhledem k tomu, že musí přeposílat zprávy od všech uzlů, které mají pod sebou, proto by bylo vhodné tyto uzly napájet z elektrické sítě.

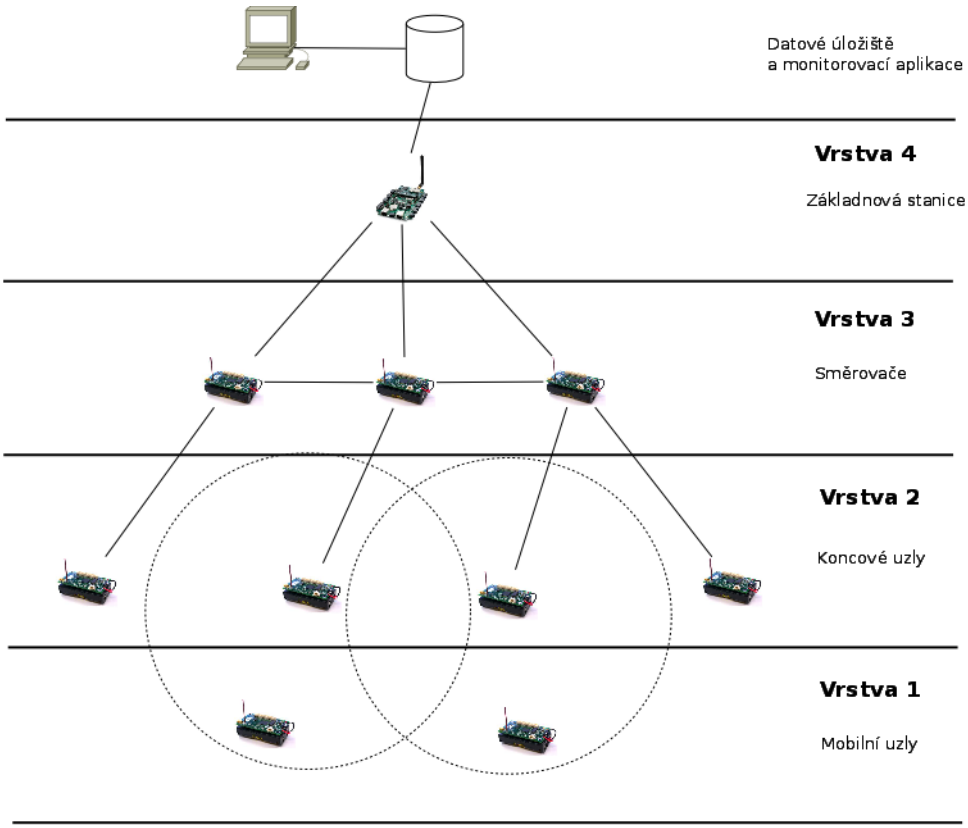
3.7.2 Komunikace v síti

Pro popis komunikace v síti použijeme diagramy chování jednotlivých uzlů. Celou síť si můžeme rozdělit na čtyři vrstvy.

První a nejnižší vrstva zahrnuje mobilní uzly, které bude nosit sledovaná osoba nebo se připevní k sledovanému objektu. Popis činnosti tohoto zařízení vidíme na diagramu na Obrázku 17a. Po zapnutí napájení se mobilní uzel ohlásí síti, jakmile se objeví v dosahu některého z kotevních uzlů. Zároveň jako první si nastaví časovač, který bude senzorový uzel probouzet periodicky v nastavenou dobu. Poté může senzorový uzel přejít do spánkového režimu. Po dobu stanovenou časovačem se uzel nachází v režimu spánku, čímž šetří energii. Po uplynutí této doby přejde uzel do aktivního režimu. Pošle zprávu se svým ID a čeká na potvrzení přijetí. Pokud nepřijde potvrzení předdefinované doby, mobilní uzel vyšle opět zprávu. Pokud přijme potvrzení, mobilní uzel se opět přepne do režimu spánku.

Na druhé vrstvě se nachází uzly s pevně danou pozicí. Tyto uzly budou rozmístěny po sledovaném patře tak, aby bylo celé celé patro pokryto signálem. Zároveň se budou rádiové dosahy jednotlivých uzlů překrývat, aby bylo možno určovat polohu jednotlivých mobilních uzlů. Pokud tedy přijmou zprávu od mobilního uzlu nejbližší kotevní uzly, oba tyto uzly přepošlou zprávu i se svými souřadnicemi do základnové stanice přes směrovače jak popisuje diagram na Obrázku 17b. Jakmile zprávu odešlou, potvrdí mobilnímu uzlu, že byla zpráva přeposlána a opět čekají na přijetí signálu.

Na třetí vrstvě jsou definovány směrovací uzly. Tyto zařízení slouže jako směrovače pro přeposílání zprávy od monitorovacích uzlů směrem k základnové stanici. Jejich nasazení má důvod



Obrázek 15: Jako topologie bude použita struktura mesh.

takový, aby nebyly zatěžovány přeposíláním jednotlivé uzly, ale aby všechna zátěž při přeposílání byla rozložena na tyto uzly.

Na nejvyšší čtvrté vrstvě bude základnová stanice s připojeným počítačem. Na této vrstvě se bude odehrávat ukládání zpracování údajů, které přicházejí ze sítě. Z těchto údajů bude možno na základě síly signálu od mobilního uzlu a souřadnic kotevních uzlů určit polohu mobilního uzlu.

3.7.3 Lokalizační algoritmus

Skutečnou pozici uzlu z naměřených vzdáleností získáme lokalizačním algoritmem. Tento algoritmus bude implementován v centrálním místě a bude počítat s daty, které přijdou ze senzorové sítě. Takovýto typ algoritmu se nazývá centralizovaný. Metoda, kterou použijeme se nazývá trilaterace. Tato metoda je založena na zjišťování vzdálenosti. Metody zjištění vzdálenosti byly popsány v první kapitole této práce. K určení polohy mobilního uzlu musíme znát jeho vzdálenost od tří okolních uzlů. My budeme zjišťovat tuto vzdálenost pomocí RSSI ukazatele, který převedeme na vzdálenost v metrech. Rovnice z první kapitoly si přepíšeme do maticového tvaru:

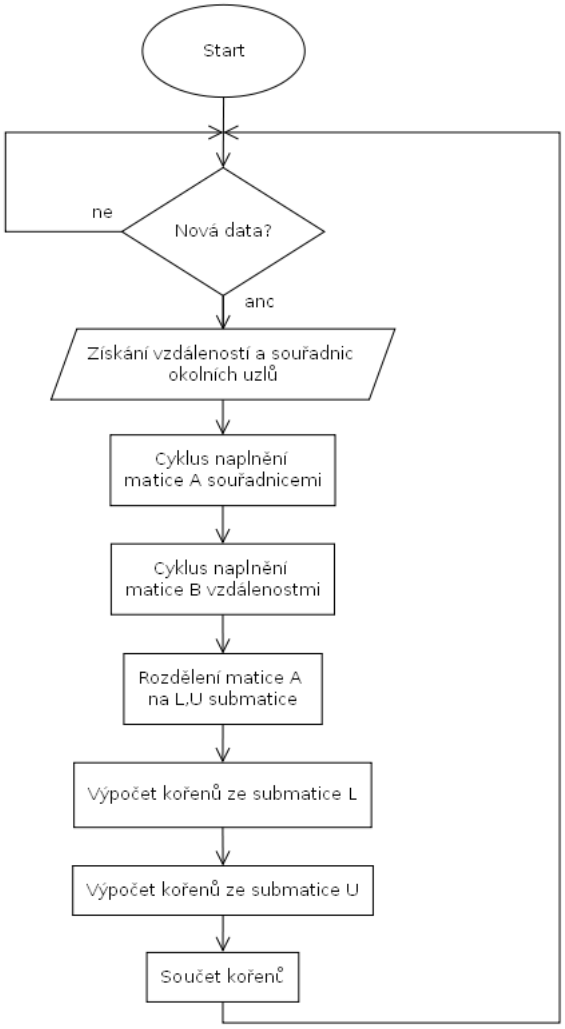
$$\begin{bmatrix} (x1-x)^2 & (y1-y)^2 & (z1-z)^2 \\ (x2-x)^2 & (y2-y)^2 & (z2-z)^2 \\ (x3-x)^2 & (y3-y)^2 & (z3-z)^2 \end{bmatrix} = \begin{bmatrix} {d_1}^2 \\ {d_2}^2 \\ {d_3}^2 \end{bmatrix}, \tag{17}$$

a hledáme řešení tohoto vztahu:

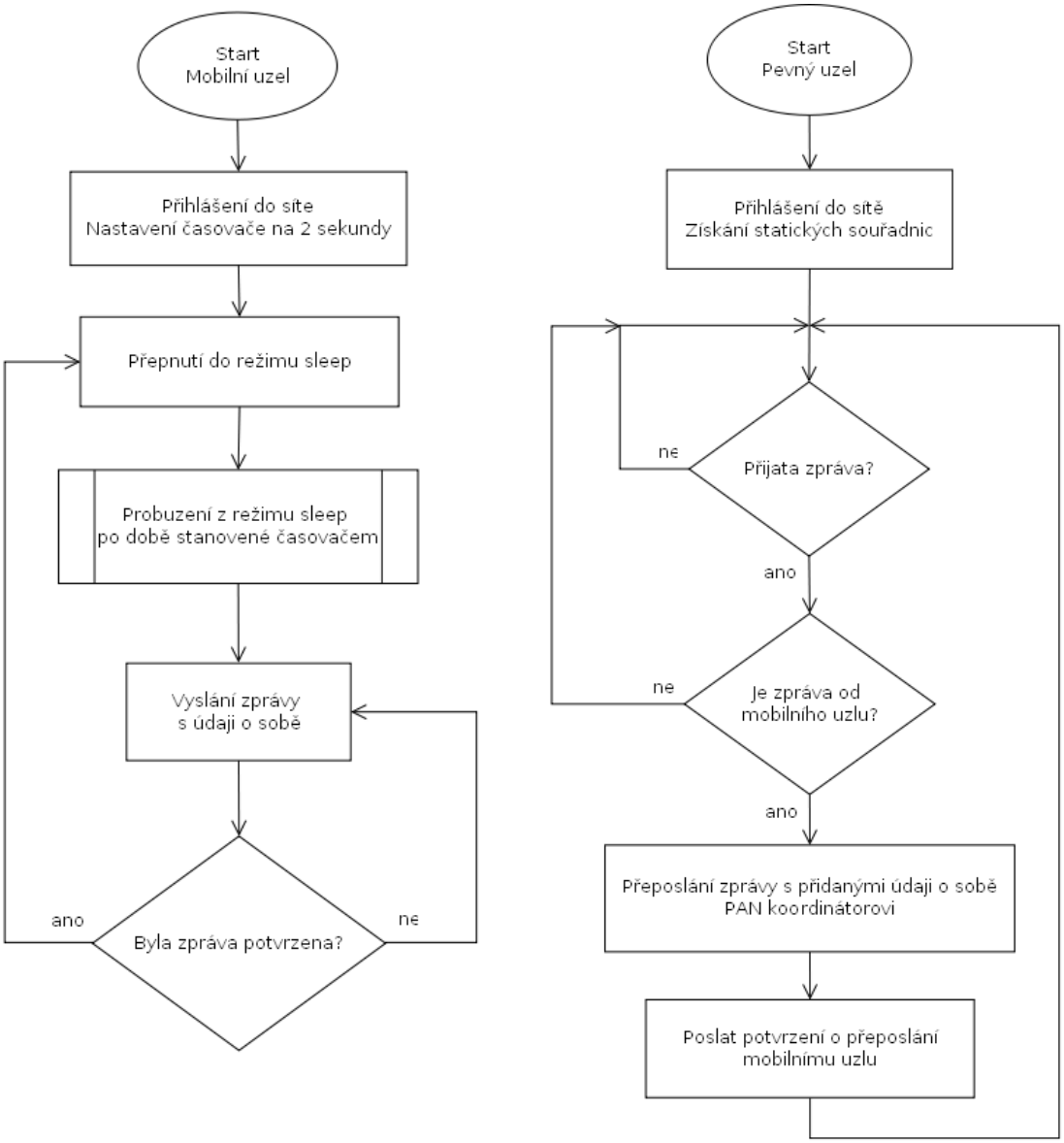
$$A \cdot x = b, \ kde: \tag{18}$$

$$A = \begin{bmatrix} (x1-x3) & (y1-y3) & (z1-z3) \\ (x2-x3) & (y2-y3) & (z2-z3) \end{bmatrix}, b = \begin{bmatrix} {d_1}^2 - {d_3}^2 - {x_1}^2 + {x_3}^2 - {y_1}^2 + {y_3}^2 \\ {d_2}^2 - {d_3}^2 - {x_2}^2 + {x_3}^2 - {y_2}^2 + {y_3}^2 \end{bmatrix}$$

Tento vztah budeme řešit pomocí LU faktorizace. Toto řešení soustavy rovnic má nejmenší náročnost prostorovou i časovou. Postup řešení je takový, že matice A se rozdělí na trojúhelník nad diagonálou (L) a na trojúhelník pod diagonálou (U). S každou touto částí se počítá zvlášť a následně se oba výsledky sečtou. Tímto rozložením se ušetří jednak potřebně místo k ukládání mezivýpočtů i čas celkového výpočtu se zkrátí.



Obrázek 16: Vývojový diagram lokalizačního algoritmu.



(a) Mobilní uzel (b) Kotevní uzel

Obrázek 17: Vývojové diagramy, popisující chování senzorových uzlů.

4 Praktická realizace

V této kapitole se budeme zabývat praktickou realizací uvedeného návrhu z předchozí kapitoly. Popíšeme si celou komunikaci od jednotlivých uzlů až po koncovou monitorovací aplikaci. V této kapitole také bude rozebrána implementace algoritmu lokalizace ve vybraném programovacím jazyce a postup vytváření databáze, která bude udržovat důležité informace v celém systému.

4.1 Koncepce monitorovacího systému

Praktická realizace monitorovacího systému bude složena z těchto částí:

- bezdrátové uzly,
- brána (k ní bude připojena základnová stanice),
- server, na kterém bude umístěna databáze,
- lokalizační algoritmus, který poběží na serveru a bude počítat lokalizaci z dat v databázi.

Všechny tyto části budou spolupracovat a jako výsledek vznikne tabulka, do které bude lokalizační algoritmus pravidelně ukládat vypočtené souřadnice mobilních uzlů. Tato tabulka bude poté dostupná pro monitorovací aplikaci.

Na tomto projektu se podílí svou diplomovou prací více studentů. Student Petr Žoldoš [15] vytváří výše uvedenou monitorovací aplikaci, která má za úkol zobrazovat v grafickém rozhraní data z databáze. Tato aplikace je přístupná přes webové rozhraní a po napojení na databázi vypisuje v tabulce všechny aktivní uzly. Vedle tabulky je do půdorysu zakresleno rozložení uzlů podle souřadnic, uvedených v tabulce.

Dále se na tvorbě senzorové sítě podílí svou diplomovou prací student Miroslav Botta [14], který se zabývá určením vhodného rádiového modelu. Tento rádiový model je poté použit pro výpočet nejvhodnější vzdálenosti ze síly signálu. Vypočítaná vzdálenost je poté vstupním parametrem lokalizačního algoritmu, jehož implementace bude popsána dále v této práci.

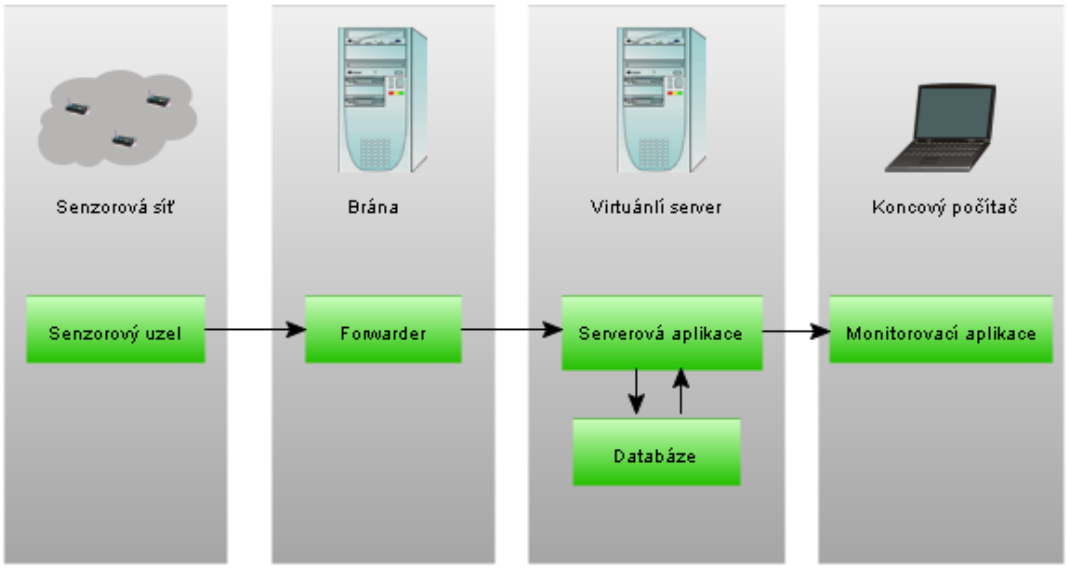
Na výchozí bráně sítě bude spuštěna aplikace, která neustále naslouchá na sériovém portu, pokud přijde paket ze sítě, tak tento paket pošle přes TCP spojení na server.

Aplikace je napsána v programovacím jazyce Java. Využívá třídu, která je vytvořena pro naslouchání na sériovém portu. Dále je zde vytvořena třída `TCPConnect`, která slouží k navázání TCP spojení, odeslání dat přes toto spojení a poté toto spojení uzavře. Tato třída se navíc spouští jako vláknová, tudíž aplikace může i při navazování spojení dále naslouchat na portu. Vždy, když je detekován nový paket na sériovém portu, je vytvořena nová instance třídy `TCPConnect` a tato třída je spuštěna v samostatném vlákně. Třída naváže spojení na se serverem a připojí se k portu 55550, na kterém naslouchá serverová část aplikace.

Na serveru běží aplikace, která implementuje serverovou obsluhu klientů. To znamená, že naslouchá na daném portu, pokud na tento port přijde TCP paket, tak z něj přečte datovou část a a takto získaná data zpracuje v nově vytvořeném vlákně. Po přijetí paketu je vytvořena nová instance třídy `ServerService`, která běží v samostatném vlákně a aplikace tak není blokována a může dále naslouchat na portu pro příjem dalšího paketů. Obslužná třída `ServerService` získá z datové části potřebné údaje (v našem případě je to ID zdrojového a cílového uzlu a síla signálu mezi nimi) a tyto údaje vloží pomocí SQL dotazu do databáze.

Senzorová síť

Senzorová síť má v tomto řetězci za úkol sesbírat data od lokalizovaných mobilních uzlů. Hledané uzly, které se nacházejí v této síti posílají pakety v pravidelných intervalech. Tyto pakety jsou přijaty nejbližšími kotevními uzly, které jsou v dosahu. Kotevní uzly pak přijaté údaje o mobilním uzlu uloží do datové části nově vytvořeného paketu a tento nový paket směřují směrem k základnové stanici. Na základnové stanici je nahrán program, který po přijmutí paketu ihned tento paket přeposílá po sériové lince do počítače, ke kterému je fyzicky připojen.



Obrázek 18: Popis funkce monitorovacího systému.

Výchozí brána

Výchozí brána, neboli Gateway, je počítač, který má k sobě připojenou základnovou stanici. Tato základnová stanice je vlastně další z uzlů sítě, který má ale ve své paměti nahraný program pro přeposílání veškeré příchozí komunikace po sériovém portu právě na naši Gateway. V počítači je nainstalována aplikace `Forwarder`. Jeho funkce byla naznačena výše. Tento program byl vytvořen v programovacím jazyce Java především kvůli rychlé obsluze příchozích paketů. Tato obsluha je dosažena pomocí vytváření nového vlákna pro každý paket, který je zpracováván, zatímco program může dál v nekonečné smyčce naslouchat na portu. Po přijetí dalších dat se opět vytvoří nové vlákno a program se ihned vrátí k naslouchání na portu. Vlákno zatím obslouží daný paket a poté se samo ukončí. Takto může pracovat paralelně několik vláken a přitom je každý paket obsloužen, viz Obrázek 20.

Server

Server je hlavní úložiště získaných dat. Na tomto místě je vytvořena databáze, která ukládá a uchovává důležitá data ze senzorové sítě. Databáze je vytvořena v prostředí PostgreSQL verze 8.4.7. Dále na serveru běží aplikace, která naslouchá na síťovém portu. Pokud přijde na daný port paket od výchozí brány (Gateway), tak z datové části tohoto paketu získá důležité údaje a tyto údaje uloží do databáze. Poté se provede lokalizace, která získá z databáze potřebné údaje, z těchto údajů spočte souřadnice lokalizovaného mobilního uzlu a tyto údaje zapíše opět do databáze. Tento výsledek bude poté dostupný pro monitorovací aplikaci.

Monitorovací aplikace

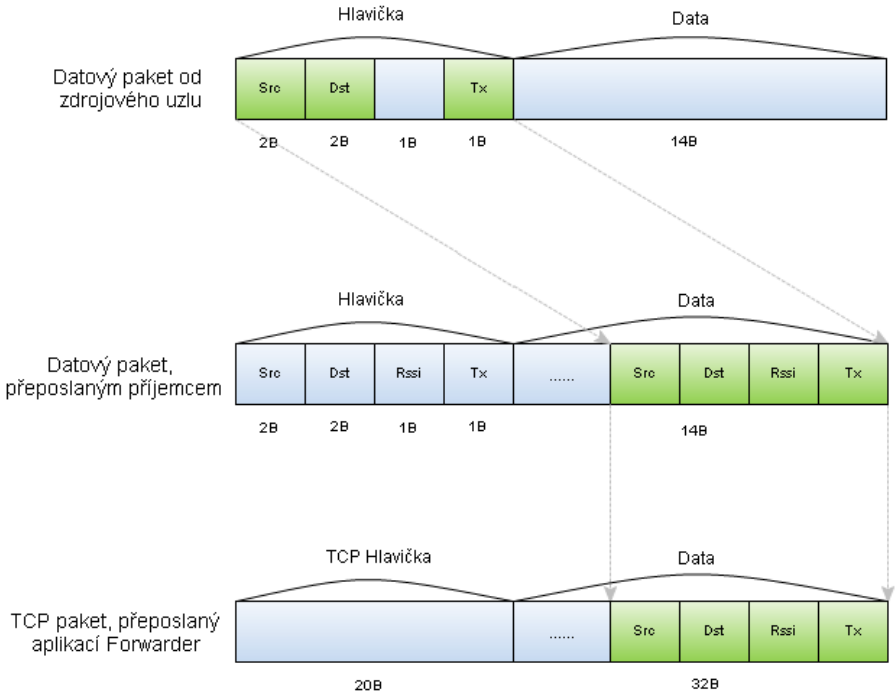
Monitorovací aplikace je součástí jiné diplomové práce, jak již bylo uvedeno výše. V celém procesu vystupuje jako koncová aplikace, která graficky zobrazuje data z databáze. Tím pomáhá k jednodušší orientaci v zjištěných datech.

4.2 Přenos dat

Senzorové uzly si mezi sebou vyměňují dva druhy paketů. První druh paketu je od mobilního uzlu a tento paket je přeposílán kotevními uzly směrem k výchozí bráně sítě. Při přeposílání musíme zajistit, aby se uchovaly údaje od zdrojového senzoru, od kterého paket přišel. Proto při přeposílání kotevní uzel data z původního paketu uloží do datové části nově vzniklého paketu a tento paket poté směřuje k výchozí bráně. Do datové části se uloží informace o zdroji paketu, tedy o mobilním uzlu, poté je v datové části informace o příjemci paketu, a informace o síle signálu mezi nimi, jak je vidět na Obrázku 19. Tato data je potřeba uchovat, abychom z nich následně mohli počítat lokalizaci.

Druhý typ paketu si vyměňují kotevní uzly mezi sebou. Opět je potřeba uchovat stejné údaje pro pozdější vyhodnocení, proto se údaje z hlavičky původního paketu uloží do datové části nově vzniklého. Tento paket se postupně přeposílá až k serveru, kde jsou z datové části vyjmuty, potřebné údaje použity pro výpočet vzdálenosti, pro výpočet rádiového modelu a poté pro samotnou lokalizaci.

Oba typy paketů mají stejnou strukturu, liší se jen tím, od jakého zdroje pocházejí a podle tohoto údaje jsou poté použity. Jestliže se jedná o paket od kotevního uzlu, tak na straně serveru se uloží do tabulky `RouterTab` (viz dále) a je použit na určení rádiového modelu. Jestliže je zdrojem paketu mobilní uzel, pak se tento paket použije k určení lokace tohoto uzlu. Přenos dat jednotlivými částmi paketu můžeme vidět na Obrázku 19.



Obrázek 19: Podoba datového paketu v různých částech systému.

4.3 Implementace potřebných aplikací

Pro fungování celé komunikace, která byla naznačena v předchozí sekci, jsou nutné aplikace, které se budou starat o odchytávání a přeposílání, či obsluhu datových paketů. Tyto aplikace byly implementovány v programovacím jazyce Java. Programovací jazyk Java má pro nás několik výhod, které můžeme plně využívat. První výhoda je platformní nezávislost. Znamená to, že aplikace takto vytvořené mohou běžet na jakékoliv platformě, aniž by se musely nějak modifikovat, nebo překompilovat. Tuto výhodu využíváme tím, že na výchozí bráně je nainstalován operační systém Windows a server běží nad operačním systémem Ubuntu. Další výhoda Javy v našem případě je možnost vytváření vícevláknových aplikací. Tato vlastnost nám velice pomáhá při zpracování datových paketů aplikacemi. Při přijetí paketu dojde k vytvoření nového vlákna, ve kterém bude postupně obsloužen daný paket. Aplikace však ihned může přejít do stavu naslouchání, aby nedošlo k nezaregistrování nového paketu. O vlákno se dále nestará. Toto vlákno dokončí podle instrukcí obsluhu paketu a poté se samo ukončí. Tímto je docíleno obsluhy všech paketů, které jsou přijaty a zároveň program může nepřetržitě naslouchat, zda přišel nový paket.

Tímto postupem byly vytvořeny aplikace `Forwarder` a `Server`. `Forwarder` je aplikace, která běží na výchozí bráně (viz výše). Server je spuštěn na virtuálním serveru, kde je připojen k lokální databázi a pracuje s daty z této databáze.

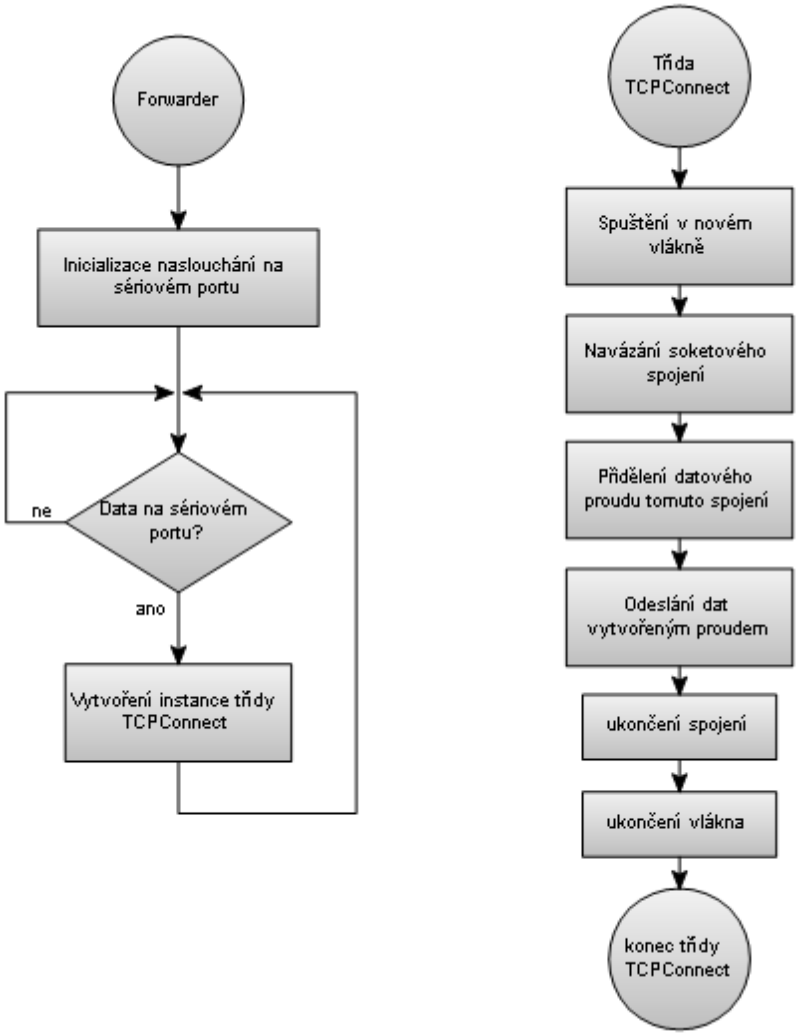
Připojení aplikace k databázi

Při implementaci serverové části aplikace bylo třeba vyřešit otázku spojení s databází. V programovacím jazyce Java existuje k tomuto účelu několik knihoven. Nejrozšířenější je knihovna JDBC, která implementuje samotné spojení do několika funkcí. Nejprve je třeba pomocí funkce `forName()` zaregistrovat databázový ovladač do systému, aby jej bylo možné dále používat. poté můžeme provést samotné napojení na databázi pomocí funkce `getConnection()`, která má tři vstupní parametry. Jsou to adresa databáze včetně použitého protokolu, přístupové jméno a heslo do databáze. Tato funkce vytvoří spojení mezi naší aplikací v Javě a lokální databází. Pomocí ukazatele na toto vzniklé spojení můžeme do databáze odesílat dotazy a přijímat výsledek dotazu. Poslání dotazu provedeme zavoláním funkce `executeQuery()` a jako parametr této funkci předáme odesílaný dotaz. Návratová hodnota funkce nám vrací odpověď databáze na dotaz. Zde je uveden příklad inicializace připojení k databázi včetně odchytávání výjimek:

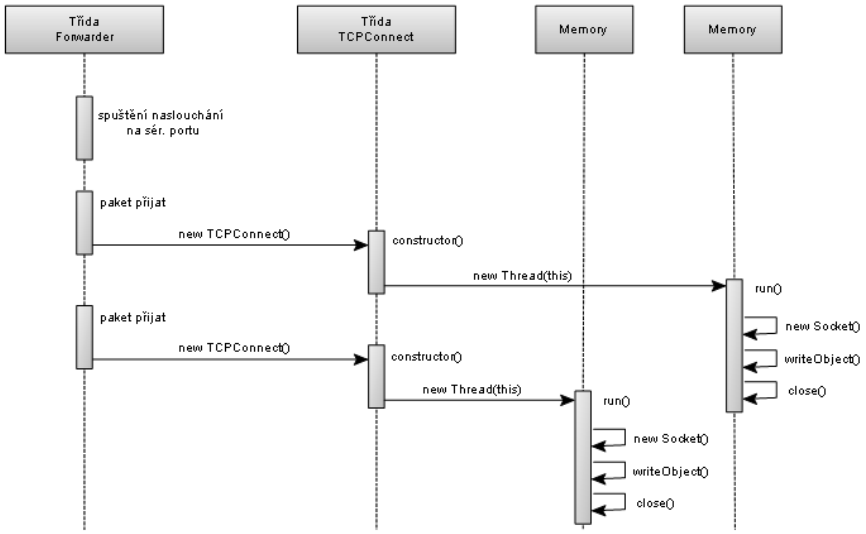
```
try {
  Class.forName("org.postgresql.Driver");
} catch (ClassNotFoundException e) {
  System.out.println("Can't find PostgreSQL JDBC Driver.");
}

try {
  connection = DriverManager.getConnection(host, name, pass);
} catch (SQLException e) {
  System.out.println("Connection Failed! Check output console");
}

Statement stmt = connection.createStatement();
stmt.executeQuery("SELECT * FROM table");
```

4.3.1 Forwarder

Tato aplikace je spuštěna na výchozí bráně a slouží k přeposílání paketů, které přijme ze sériového portu od základnové stanice. Jakmile přijme takový paket, spustí samotné vlákno a v tomto vlákně obslouží přijatý paket. Spustí se instance třídy `TCPConnect`, která vytvoří spojení TCP. Toto soketové spojení je realizováno vytvořením instance třídy Socket voláním new `Socket()`. Tomuto konstruktoru je třeba předat ip adresu, v našem případě je to ip adresa serveru, na který data přeposíláme. Jako druhý parametr je potřeba port, na kterém server naslouchá. Tímto je navázáno spojení. Na toto spojení se naváže proud, přes který poté budeme posílat data. Tento proud bude objektový, to znamená, že lze přes něj přenášet jakékoli objekty. Proud přiřadíme opět vytvořením instance třídy `ObjectOutputStream()` a jako parametr jí předáme ukazatel na naše soketové spojení. Tímto proudem odešleme data pomocí metody `writeObject()`. Jakmile jsou data odeslána, můžeme spojení zavřít metodou `close()` a ukončit vlákno, ve kterém běží tato obsluha paketu. Celý proces můžeme vidět na obrázcích 20 a 21. Aplikace zatím naslouchá na sériovém portu, po příchodu dalšího paketu celá obsluha provede znovu pro tento paket.



Obrázek 20: Vývojový diagram aplikace Forwarder.



Obrázek 21: Sekvenční diagram aplikace Forwarder.

4.3.2 Serverová část

V předchozí sekci byla popsána aplikace, která pouze přeposílá data na server. V této sekci si popíšeme samotnou serverovou část, která tyto pakety zpracovává a poté počítá pomocí lokalizačního algoritmu souřadnice mobilního uzlu, ze kterého příslušný paket přišel.

Vývojový diagram aplikace můžeme vidět na Obrázku 22. Aplikaci musíme nejdříve říci, jakým způsobem bude zachytávat příchozí pakety. K tomuto účelu jsou již vytvořeny třídy pro práci se serverovým odchytáváním. K tomu použijeme metodu `ServerSocket()` z třídy java.net. Touto metodou vytvoříme soket, který běží v pozadí aplikace a neustále naslouchá na portu, který dostane jako argument. Návratová hodnota této metody je odkaz na tento soket, abychom mohli později přistupovat k jednotlivým paketů.

Po tomto volání musíme ještě připojit naši aplikaci k databázi. Toto spojení bude aktivní po celou dobu běhu aplikace bude ukončeno až po ukončení aplikace. Navázání spojení s databází bylo uvedeno dříve. Po úspěšném spojení si musíme do proměnné uložit odkaz na toto spojení, protože ho budeme potřebovat vždy, když budeme posílat dotaz do databáze, tedy vždy po přijetí paketu ze sítě.

Následuje už jen spuštění samotné smyčky programu. V této smyčce bude program čekat na přijetí síťového paketu pomocí metody `accept()`. Jakmile je tento paket přijat, metoda `accept()` vrátí ukazatel na tento paket vytvoří se nová instance třídy `ServerService`. Po zavolání konstruktoru této třídy je také vytvořena nová instance třídy `Estimation`, ve které je proveden výběr nejvhodnějších kotevních uzlů a pomocí jejich rádiového modelu spočítána vzdálenost. Třída `Estimation` však není předmětem této práce, ale diplomové práce výše uvedeného studenta Miroslava Botty. Poté se aplikace opět vrátí zpět k naslouchání na portu.

Třída `ServerService` obsahuje konstruktor, který je volán při vytváření instance této třídy. V tomto konstruktoru se tato třída spustí v samotném vlákně, aby aplikace mohla dále odchytávat pakety ze sítě. V samostatném vlákně je získána datová část paketu a uložena do bytového pole. Poté z tohoto bytového pole získáme jednotlivé údaje. Jsou to adresa odesílatele, adresa příjemce a hodnota RSSI. Tuto hodnotu musíme přepočítat pomocí vzorce $RSSI[dB] = -3 * (RSSI - 1)$, abychom získali hodnotu v jednotkách dB.

Takto získané údaje vložíme do databáze. Před samotným vložením musíme určit, zda je paket vyslán od mobilního uzlu a je tedy určen pro lokalizaci, nebo zda je tento paket posílán mezi dvěma

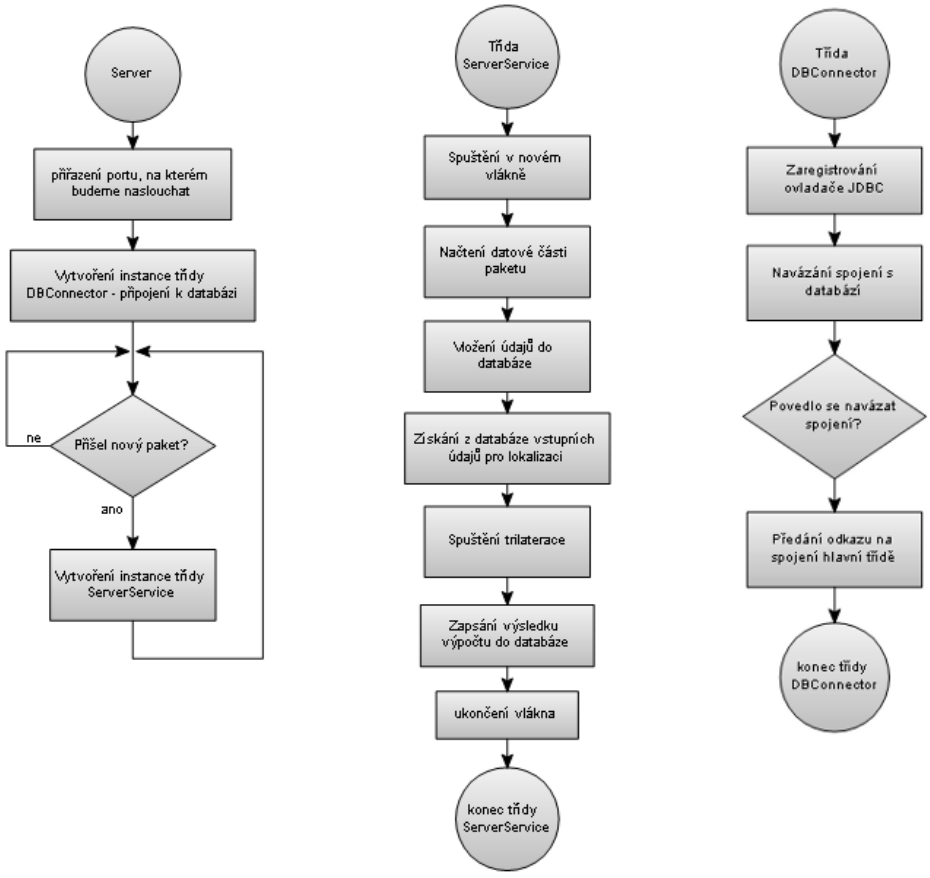
kotevními uzly. Takovýto paket slouží pro určení rádiového modelu prostředí. Podle druhu paketu vložíme data do odpovídající tabulky. Tyto tabulky a databáze budou popsány dále.

Po vložení do databáze může začít samotný cyklus lokalizace. Napřed musíme získat vstupní data z databáze. Provedeme tedy dotaz, který nám vrátí souřadnice všech kotevních uzlů, které mají rádiový dosah k mobilnímu uzlu a vzdálenost těchto uzlů od mobilního uzlu. Pokud je počet řádků vrácených hodnot menší než 3 (tzn. počet kotevních uzlů v dosahu je menší než 3), tak nemůžeme počítat lokalizaci a vlákno rovnou ukončíme. V opačném případě můžeme zavolat metodu `multilateration()` s parametry, které jsme získali posledním dotazem.

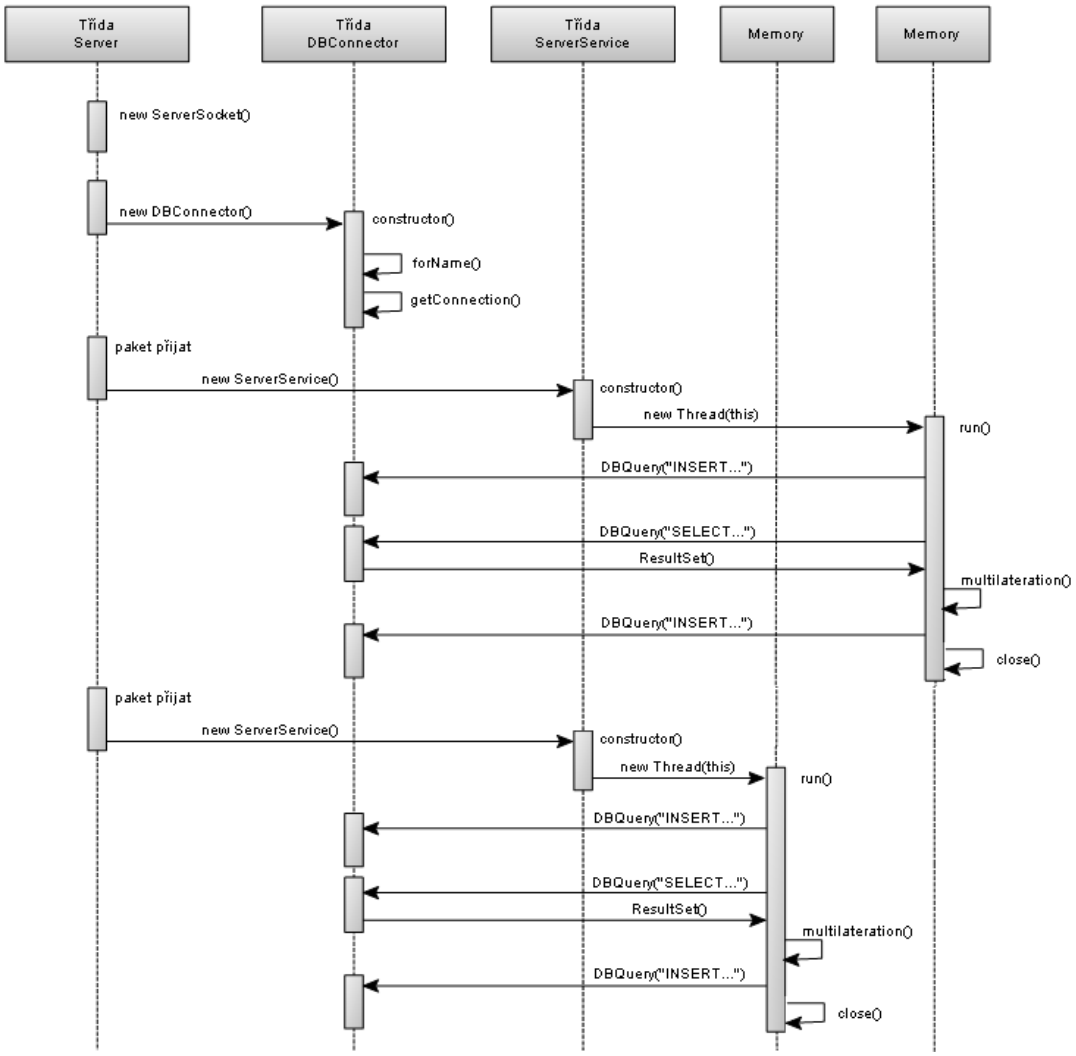
Tato metoda vytvoří samotné matice, potřebné pro výpočet. Postup vytvoření matic je popsaný výše v teoretické části. A s těmito maticemi již počítá jako s rovnicí. Výsledný vektor souřadnic metoda `multilateration()` vrátí jako matici.

Data z takto získané matice můžeme nakonec uložit do databáze do tabulky `MobileNodes`. Do této tabulky bude lokalizační algoritmus zapisovat všechny své výsledky a tabulka bude poté přístupná pro koncovou monitorovací aplikaci ke grafickému zveřejnění výsledků.

Po tomto úkonu už jen můžeme ukončit běh vlákna, které sloužilo k obsluze přijatého paketu. Aplikace mezitím spustila další vlákna, která mezitím vykonávají stejnou činnost, jako bylo popsáno výše. Celý proces vykonávání jednotlivých funkci je opět možno vidět na sekvenčním diagramu na Obrázku 23.



Obrázek 22: Vývojový diagram aplikace Server.



Obrázek 23: Sekvenční diagram aplikace Server.

4.3.3 Trilaterační algoritmus

Tento algoritmus je jedním z nejpoužívanějších zástupců lokalizačních algoritmů. Je založen na jednoduchém principu, kdy k určení polohy požadovaného uzlu potřebujeme znát polohy tří okolních uzlů. Po zjištění vzdálenosti tohoto uzlu od jednotlivých kotevních uzlů můžeme vytvořit soustavu rovnic (viz výše). Tuto soustavu rovnic můžeme převést na soustavu matic a počítat pomocí matematických metod (např. LU rozklad, Gaussova eliminace, prosté násobení transponovanou maticí).

Samotná implementace lokalizačního algoritmu zahrnuje tyto body:

- Získání vstupích dat(pozice,vzdálenosti) z databáze,
- sestavení matic z těchto hodnot,
- výpočet souřadnic z maticového zápisu,
- zapsání výsledných souřadnic do databáze.

Jako první tedy sestavíme dotaz nad databází, který nám získá z tabulek potřebné údaje. Při lokalizaci jsou pro nás důležité pozice všech tří okolních uzlů a jejich vzdálenost od hledaného uzlu. Pokud tyto údaje jsou dostupné v databázi a dotaz nám je vrátí, můžeme počítat. V opačném případě, pokud dotaz nevrátí dostatek údajů, tak ukončíme provádění a počkáme na příchod dalšího datového paketu na server a jeho zapsání do databáze.

Pokud jsem tedy získali dotazem vstupní údaje, tak tyto data uspořádáme do maticového zápisu. Vytvoříme maticovou rovnici, kdy nalevo bude matice, složená ze souřadnic uzlů. Tato matice bude násobena vektorem výsledných souřadnic, který je v této rovnici neznámou. Na pravé straně rovnice bude vektor vzdáleností od jednotlivých uzlů.

Jako třetí krok je třeba tuto rovnici vyřešit. My jsme použili k výpočtu metodu LU rozkladu, kdy matici na levé straně rozložíme na trojúhelníky L a U. Výpočet s těmito trojúhelníky je podstatně rychlejší a méně náročná na úložný prostor. Po těchto dvou výpočtech máme dva výsledné vektory, ze kterých vytvoříme průměr a získáme jeden výsledný vektor. K této operaci můžeme využít v programovacím jazyce Java knihovnu Jama, která byla vytvořena pro práci s maticemi. Obsahuje mimo jiné metodu `LUDecomposition()`, která vytvoří dvě trojúhelníkové matice a nad nimi pak zavoláme metodu `solve()`, která nám vrátí výsledný vektor hledaných souřadnic.

Poté stačí odeslat do databáze dotaz pro vložení výsledných souřadnic do správné tabulky v databázi. Tato tabulka bude obsahovat ID hledaného uzlu, které je udržováno po celou dobu výpočtu v paměti aplikace. Po výpočtu se tedy zapíše na nový řádek v tabulce ID hledaného uzlu, souřadnice v X-ové,Y-ové,Z-ové ose a Timestamp pro zaznamenání času vložení.

4.3.4 Weighted Centroid

Weighted centroid algoritmus (WCA) [19] je založena na jednoduché metodě váhování, přitom je efektivní. Princip je takový, že výsledná pozice je vypočítána z pozic jednotlivých kotevních uzlů pomocí váhovácího koeficientu. Tento váhovací koeficient je vypočítán pro každý kotevní uzel zvlášť a to tak, že získáme obrácenou hodnotu jeho změřené vzdálenosti od hledanéhu uzlu. Tuto vzdálenost ještě umocníme koeficientem. Tímto vytvoříme jednotlivé koeficienty, které udávají poměr vzdáleností všech uzlů. Když poté těmito koeficienty jednotlivé souřadnice vynásobíme, získáme souřadnici mobilního uzlu.

Implementace tohoto algoritmu v programovacím jazyce Java se skládá z těchto částí:

- Získání vstupích dat(pozice,vzdálenosti) z databáze,
- vypočítání vah pro všechny kotevní uzly v dosahu,
- váhování všech souřadnic pomocí získaných koeficientů,
- zapsání výsledných souřadnic do databáze.

Napřed tedy opět provedeme dotaz do databáze a tím získáme seznam všech uzlů, které jsou v dosahu. Údaje, které použijeme, jsou pozice všech uzlů a jejich RSSI k hledanému mobilnímu uzlu.

Z těchto údajů můžeme vypočítat jednotlivé váhy podle vzorce:

$$w_i = \frac{1}{d_i^g}, \tag{19}$$

kde w_i je váhovací koeficient pro jednotlivé kotevní uzly, d_i je změřená vzdálenost i-tého kotevního uzlu od hledaného uzlu a g je tzv. koeficient špičatosti [19]. Prakticky udává, které uzly budou mít větší váhu. Pokud je tento koeficient malý (1,2,3), nebudou se váhy mezi jednotlivými kotevními uzly příliš lišit. Pokud bude koeficient špičatosti větší (5 a více), bude mít mnohem větší váhu nejbližší uzel. Pokud by byl příliš velký (10 a více), tak je tímto zanášena do lokalizace chyba, protože by algoritmus příliš upřednostňoval nejbližší uzel. Po několika pokusných měřeních jsme určili jako nejlepší koeficient špičatosti $g = 5$. Vypočítané váhovací koeficienty dosadíme do vzorce pro získání výsledných souřadnic:

$$x = \left(\sum_{1}^{N-1} w_i \cdot s_i \right) / \sum_{1}^{N-1} w_i, \tag{20}$$

kde x je vektor souřadnic mobilního uzlu, w_i jsou koeficienty získané ze vzorce (19) a s_i je souřadnice i-tého kotevního uzlu.

Takto získáme vektor souřadnic a stejným způsobem jako u minulého algoritmu zapíšeme do databáze do tabulky MobileNodes.

4.4 Databáze

Pro správnou funkci naší senzorové sítě jsou nejdůležitější údaje a data, která pravidelně tato síť sbírá. Každou chvíli je potřeba zpracovat nové údaje a poskytnout koncovým uživatelům a aplikacím. Proto je potřeba databázový systém pro jejich ukládání. Pro tento účel byla vytvořena v databázovém systému PostgreSQL. Databáze je umístěna na virtuálním serveru, kde je spuštěna také serverová část aplikace. Tato aplikace tak může k databázi přistupovat lokálně, což přispívá k bezpečnosti komunikace.

Databáze se skládá celkem ze čtyř tabulek. Jsou to tabulky:

1. AnchorNodes,
2. MobileTab,
3. RouterTab,
4. MobileNodes.

V následující části si postupně vysvětlíme význam jednotlivých tabulek.

Tabulka `AnchorNodes` obsahuje pozice kotevních uzlů, které jsou potřebné k výpočtům údajů pro ostatní tabulky. Pro uložení pozice jsou v tabulce určeny tři sloupce `PosX`, `PosY`, `PosZ`. Pozice je tedy určována v trojrozměrné soustavě. Kromě pozice je v tabulce sloupec `NodeID`, který obsahuje ID uzlu, ke kterému patří dané souřadnice. Poslední sloupec `TimeStamp` obsahuje časovou známku pro zaznamenání data a času, kdy byl do této tabulky příslušný uzel přidán. Do této tabulky je potřeba zadat souřadnice všech kotevních uzlů, které se v síti nacházejí, ještě před samotným spuštěním celého systému. Pokud v této tabulce nebude některý uzel uveden, tak lokalizační algoritmus ani ostatní tabulky nebudou moci s tímto uzlem počítat, i když bude fyzicky v síti. Při vložení řádku do tabulky se automaticky zaznamená aktuální datum a čas do sloupce `TimeStamp`. Pokud dojde k přesunu fyzického uzlu na jinou pozici, tak je třeba v této tabulce zaznamenat novou pozici tohoto uzlu.

V případě, že je přijat ze senzorové sítě nový paket, tak aplikace na serveru tento paket uloží do tabulky `MobileTab`, nebo do tabulky `RouterTab`. Toto rozhodování probíhá na základě toho, zda paket přišel z mobilního uzlu, pak jsou data z tohoto paketu uložena do tabulku `MobileTab`. Pokud má paket jako zdroj vysílání uveden kotevní uzel, jeho data se uloží do tabulku `RouterTab`.

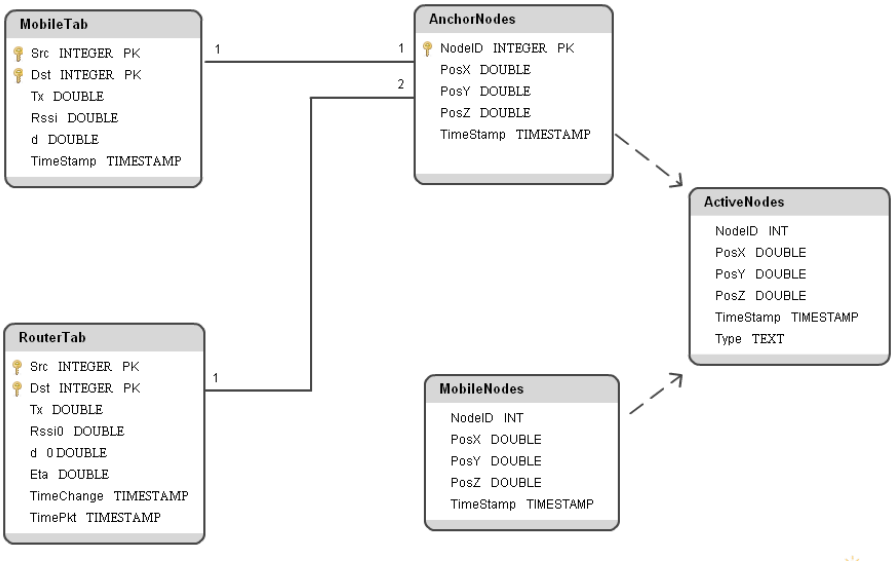
`MobileTab` tedy obsahuje údaje od mobilního, tedy lokalizovaného uzlu. Obsahuje sloupce `Src` a `Dst`, do kterých je zaznamenán zdrojový a cílový uzel vyslaného paketu. Tyto dva sloupce tvoří primární klíč tabulky. Další důležitý sloupec, nazvaný `Rssi`, obsahuje sílu signálu, s jakou byl tento paket přijat. Z tohoto údaje se dále bude počítat vzdálenost podle vzorce 12, uvedeného v teoretické části. Tato vzdálenost bude zapsána do sloupce `d`. Z této vzdálenosti bude lokalizační algoritmus počítat pozici hledaného uzlu. Hodnota `Tx` v této tabulce udává vysílací výkon uzlu, uvedeného ve sloupci `Src`.

Tabulka `RouterTab` byla navržena pro uchovávání propagačních modelů mezi dvojicemi kotevních uzlů. Tyto modely jsou vytvářeny pravidelně z údajů v paketech, které si posílají kotevní uzly mezi sebou navzájem. Proto budou data z paketu, který byl odeslán z kotevního uzlu zaznamenána do této tabulku. Tento model je popsán údaji ze sloupců `Rssi0`, `d0` a sloupcem `Eta` (viz kapitola o propagačním modelu). V této tabulce jsou ještě sloupce `Src` a `Dst`. Stejně jako v tabulce `MobileTab`, tvoří tyto dva sloupce primární klíč tabulky `RouterTab`.

Ukládání dat do dvou posledně jmenovaných tabulek probíhá následovně:

1. pokud záznam se stejným primárním klíčem v tabulce neexistuje, tak je řádek v tabulce vytvořen,
2. pokud záznam se stejným primárním klíčem v tabulce existuje, dojde k jeho nahrazení.

Poslední tabulka, `MobileNodes`, obsahuje výsledné souřadnice mobilních uzlů z lokalizačního algoritmu. Tyto údaje jsou obsaženy ve sloupcích `PosX`, `PosY`, `PosZ`. Dále v tabulce je sloupec `TimeStamp`, kde je opět zaznamenána časová známka, kdy byl do této tabulky zapsán výsledek. Tento údaj je důležitý z důvodu zobrazení pohybu mobilního uzlu. Aplikace má tedy jak funkci lokalizační, tak i funkci trasování objektu v čase.



Obrázek 24: ER diagram návrhu databáze.

4.4.1 Vazby mezi tabulkami

Na tabulku `AnchorNodes` navazují ostatní tabulky, proto si popíšeme vztahy mezi jednotlivými tabulkami. Tabulka `MobileTab` má k tabulce `AnchorNodes` vztah 1:1, protože vždy bude v tabulce `MobileTab` právě jeden záznam odkazovat na právě jeden záznam v tabulce `AnchorNodes`. Tato vazba ukazuje na kotevní uzel, který je uveden jako příjemce paketu v `MobileTab`.

Další vazba je mezi `RouterTab` a `AnchorNodes`. Tato vazba je 1:2, jelikož v tabulce `RouterTab` jsou uloženy pakety, které si vyměňují kotevní routery. Tyto pakety obsahují jako zdroj i jako příjemce kotevní uzel. Pro každé dva komunikující uzly bude v tabulce vždy jen jeden řádek. Proto jednomu záznamu odpovídají dva záznamy v tabulce `AnchorNodes`.

Tabulka `MobileNodes` nemá s ostatními tabulkami žádnou vazbu, jelikož jsou do ní ukládány výsledné souřadnice lokalizačního výpočtu. Tato tabulka pouze uchovává výsledky tohoto výpočtu a zprostředkuje tyto údaje koncové aplikaci. V našem případě je to aplikace na zobrazení pozice jednotlivých uzlů.

Sloupec `d0` v tabulce `RouterTab` obsahuje údaj o vzdálenosti kotevních uzlů od sebe. Tento údaj není potřeba měnit pravidelně protože kotevní uzly jsou od sebe vzdáleny pořád stejně. Výjimkou je událost, kdy dojde ke změně souřadnic, nebo k vymazání kotevního uzlu z tabulky `AnchorNodes`. Pro tento případ jsme vytvořili v databázi pravidlo, které jako reakci na uvedené události smaže tabulku `RouterTab`. Tato tabulka se opět postupně naplní aktuálními daty včetně nově propočítaného `d0`.

4.4.2 Databázové funkce a pohledy

Pro snadnější vkládání údajů z paketu do databáze byly vytvořeny v databázovém systému funkce. Tyto funkce jsou volány serverovou částí programu s parametry. Tyto parametry obsahují data, která se mají uložit do databáze. Funkce se vykoná v databázovém systému, spočte všechny potřebné údaje a zapíše údaje do tabulek. Výpočet je potřeba provádět vždy při vkládání údaje `Rssi0`, `Eta` a `d0`. `Rssi` se vždy průměruje s předchozím údajem v tabulce, `Eta` se spočte podle vzorce:

$$Eta = \frac{log(\frac{Rssi}{P_{Tx}})}{log(\frac{1}{d0})}, \tag{21}$$

kde P_{Tx} je vysílací výkon daného uzlu. Poslední údaj, `d0`, bude počítán pomocí pythagorovy věty, tedy pomocí následujícího vztahu:

$$d0 = \sqrt{(PosX_1 - PosX_2)^2 + (PosY_1 - PosY_2)^2 + (PosZ_1 - PosZ_2)^2}, \tag{22}$$

kde $Pos_1 - Pos_2$ je vždy rozdíl souřadnic uzlů, mezi kterými zjišťujeme vzdálenost, v jednom směru. Tyto výpočty provádějí uváděné funkce a to vždy při vkládání paketu do tabulek. funkce mají následující tvary:

- mobilnipaket(integer, integer, double precision),
- routerpaket(integer, integer, double precision).

Obě funkce mají stejné argumenty, kterými jsou: ID vysílacího uzlu, ID přijímacího uzlu, přijaté RSSI. Funkce mobilepaket() vkládá údaje do tabulky. Pokud daný řádek v tabulce existuje, stačí udělat jen update případných sloupců. V opačném případě je potřeba tento řádek do tabulky vložit. Následuje tělo funkce mobilepaket():

```
-- pokud řádek v tabulce existuje, tak jen přepíšeme hodnoty --
UPDATE "MobileTab"
  SET "Rssi"=($3+(SELECT "Rssi"
                  FROM "MobileTab"
                  WHERE "Src"=$1 AND "Dst"=$2
                  LIMIT 1))/2,
      "TimeStamp"=now()
  WHERE "Src"=$1 AND "Dst"=$2;

-- pokud neexistuje, tak ho vytvoříme --
INSERT INTO "MobileTab"("Src","Dst","Rssi","Tx")
  SELECT $1,$2,$3,3.2
    WHERE NOT EXISTS(SELECT 1 FROM "MobileTab" WHERE "Src"=$1 AND "Dst"=$2);
```

Vidíme, že jako první se zavolá dotaz UPDATE, který se snaží nahradit jen potřebné údaje. Tento příkaz se však neprovede, pokud nenajde žádný odpovídající řádek. Poté se provede příkaz INSERT jedině v tom případě, pokud tabulka tento řádek neobsahuje. Primárním klíčem každého řádku je dvojice sloupců `Src` a `Dst`.

Funkce routerpaket() pracuje s tabulkou `RouterTab`. Tato funkce se opět napřed pokouší nahradit pomocí dotazu UPDATE jen potřebné sloupce, pokud daný řádek v tabulce neexistuje, tak jej vytvoří pomocí dotazu INSERT. Primární klíč je opět dvojice sloupců `Src` a `Dst`. V této funkci je však při vkládání údajů třeba některé přepočítat. Jedná se o údaje:

- Rssi - průměrování,
- Eta - počítána podle vzorce, uvedeného výše,
- d0 - podle pythagorovy věty ze souřadnic v tabulce `AnchorNodes`.

Zde je tělo této funkce:

```
-- pokud řádek  v tabulce existuje, tak jen přepíšeme hodnoty --
UPDATE "RouterTab"
  SET "Rssi0"=($3+(SELECT avg("Rssi0")
                   FROM "RouterTab"
                   WHERE ("Src"=$1 AND "Dst"=$2)
                     OR ("Src"=$2 AND "Dst"=$1)))/2,
      "TimePkt"=now(),
      "Eta"=log(abs(($3)/(3.2)))/log((1)/("d0"))
  WHERE ("Src"=$1 AND "Dst"=$2) OR ("Src"=$2 AND "Dst"=$1);

-- pokud neexistuje, tak ho vytvoříme --
INSERT INTO "RouterTab" ("Rssi0","d0","Eta","Src","Dst","Tx")
  SELECT $3,
         -- výpočet d0 --
        (SELECT sqrt(power(max("PosX")-min("PosX"),2) +
                power(max("PosY")-min("PosY"),2) +
                power(max("PosZ")-min("PosZ"),2))
          FROM "AnchorNodes" WHERE "NodeID"=$1 OR "NodeID"=$2),
         -- výpočet Eta --
         log((($3)+100)/(103.2))/log((1)/(
          SELECT sqrt(power(max("PosX")-min("PosX"),2) +
                 power(max("PosY")-min("PosY"),2) +
                 power(max("PosZ")-min("PosZ"),2))
          FROM "AnchorNodes" WHERE "NodeID"=$1 OR "NodeID"=$2)),
         $1,
         $2,
         3.2
    WHERE NOT EXISTS(SELECT 1 FROM "RouterTab" WHERE
              ("Src"=$1 AND "Dst"=$2) OR ("Src"=$2 AND "Dst"=$1));
```

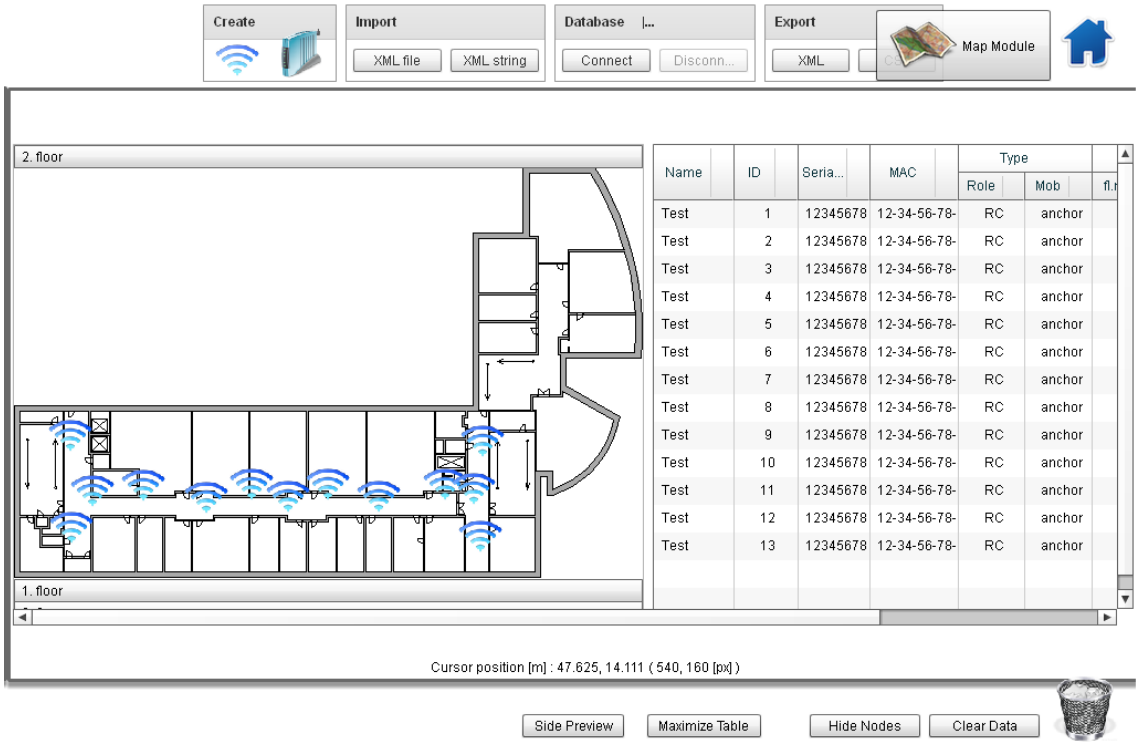
V databázi je kromě tabulek také vytvořen pohled. Pohledy je výhodné tvořit, pokud provádíme často dotaz SELECT nad více tabulkami a dotaz by tak byl příliš složitý. Tento pohled je nazván ActiveNodes. Obsahuje dotaz, který získává seznam všech kotevních uzlů a k tomuto seznamu přidává i seznam všech mobilních uzlů, které jsou v danou chvíli aktivní. To znamená, že z tabulky MobileNodes získá uzly, které byly s daným NodeID přidány do tabulku jako poslední a zároveň nejsou starší, než daný interval. Tím získáme uzly, které opravdu v senzorové síti aktuálně vysílají a získáme tím i jejich nejaktuálnější polohu. Tento pohled byl vytvořen pro koncovou aplikaci, která tyto údaje zobrazuje. Následuje výpis těla tohoto pohledu:

```
-- vrátí seznam všech kotevních uzlů z tabulky AnchorNodes --
SELECT "AnchorNodes"."NodeID",
       "AnchorNodes"."PosX" AS "PositionX",
       "AnchorNodes"."PosY" AS "PositionY",
       "AnchorNodes"."PosZ" AS "PositionZ",
       "AnchorNodes"."TimeStamp" AS "Timestamp",
       'Anchor' AS "Type"
    FROM "AnchorNodes"

UNION
```

```
-- získání seznamu uzlů s posledním výskytem pro každé ID uzlu --
SELECT t."NodeID", t."PositionX", t."PositionY", t."PositionZ",
       t."Timestamp", 'Mobile' AS "Type"
  FROM ( SELECT row_number() OVER (PARTITION BY "MobileNodes"."NodeID"
                  ORDER BY "MobileNodes"."TimeStamp" DESC) AS rowno,
                "MobileNodes"."NodeID",
                "MobileNodes"."PosX" AS "PositionX",
                "MobileNodes"."PosY" AS "PositionY",
                "MobileNodes"."PosZ" AS "PositionZ",
                "MobileNodes"."TimeStamp" AS "Timestamp"
           FROM "MobileNodes") t
  WHERE t.rowno = 1
```

4.5 Aplikace pro monitorovaní

Jak bylo výše uvedeno, souběžně s touto prací vzniká i aplikace, která zobrazuje data z databáze v grafické podoby. Tato aplikace je dostupná přes webové rozhraní a po připojení k databázi se nám objeví podobné rozmístění uzlů jako je na Obrázku 25. Vedle půdorysu patra s umístěním senzorových uzlů je i tabulka, ve které jsou vidět detailní informace k jednotlivým uzlům.



Obrázek 25: Ukázka monitorovací aplikace s rozmístěním kotevních uzlů.

5 Testování

Po vytvoření naší senzorové sítě je potřeba zjistit, s jakou přesností jsou určovány souřadnice a zda je navržený systém použitelný v praxi. Proto si musíme určit nějaký systém měření, kterým ověříme chování sítě při určování pozice.

5.1 Metody zjišťování chybovosti

Měření jsme rozdělili na dvě části. První z nich jsme proměřili v místnosti, kde jsme rozestavili kotevní uzly v rozích této místnosti a postupně na několika místech jsme provedli lokalizaci polohy algoritmem a její uložení v databázi. Pro každou pozici jsme takto zaznamenali pět určených pozic v rozestupu 1 minuty, abychom získali data z delší časové oblasti. Tyto souřadnice jsme uložili do databáze a porovnali se skutečnými souřadnicemi, které jsme zadali do databáze ručně. Takto získaná data byla převedena do podoby tabulek a poté z nich byl vytvořen graf pro lepší představu.

Půdorys laboratoře a přilehlé chodby můžeme vidět na Obrázku 26. V laboratoři bylo provedeno měření na osmi místech s pěti opakováními na každé pozici. Na chodbě pak bylo provedeno měření na deseti pozicích. Data byla poté uspořádána do tabulek 4 až 11 v příloze a poté zanesena do uvedeného půdorysu.

Druhá část měření byla provedena ve třetím podlaží budovy. Kotevní uzly byly rozestaveny podle návrhu, avšak jen na chodbě tohoto podlaží. V laboratořích nebyly kotevní uzly rozmístěny. Poté bylo provedeno na několika místech po chodbě statické měření, kdy jsme mobilní hledaný uzel umístili na jedno místo a poté provedli několik měření. Poté jsme mobilní uzel posunuli na další pozici a opět opakovali měření. Takto jsme prošli celou chodbu. Naměřené hodnoty jsou vidět na Obrázku 27. Data jsme uspořádali do tabulek 12 a 13 v příloze.

Po tomto měření jsme provedli testování, při kterém jsme nechali systém sledovat pohybující se mobilní uzel. Rozmístění kotevních uzlů zůstalo stejné. Poté jsme několikrát prošli chodbou v obou směrech se sledovaným mobilním uzlem. Lokalizované pozice se mezi tím stále ukládaly do databáze. Po skončení měření jsme data analyzovali a graficky znázornili.

5.2 Vyhodnocení

Pro zkoumání naměřených dat je vhodné udělat statistické testy, které nám dají větší přehled o chybovosti.

Jako první si určíme průměrnou hodnotu pro každý typ měření zvlášť a určíme odchylku od skutečné hodnoty. Střední hodnota je nejznámější hodnota ve statistice, je definována jako vážený průměr daného rozdělení. Tuto hodnotu vypočítáme podle vztahu:

$$E(x) = \frac{1}{d} \sum_{i=0}^{d} x_i, \tag{23}$$

kde $E(x)$ je střední hodnota pro soubor hodnot x, d počet naměřených hodnot a x_i jsou naměřená data. My dosadíme za x_i jednotlivé odchylky při určování vzdáleností a za d dosadíme celkový počet měření.

Pro statistické vyhodnocení naměřených hodnot nám však nestačí pouze střední hodnota. Potřebujeme ještě údaj, který nám řekne, jaké mohou být hraniční hodnoty chybovosti. Společně se střední hodnotou se proto používá i směrodatná odchylka, která nám dává obrázek o tom, jak moc se mohou chybové vzdalovat od střední hodnoty. Pokud je nějaká hodnota vzdálena o více, než je směrodatná odchylka, můžeme tuto hodnotu považovat za statisticky bezvýznamnou. Naměřená data mají normální rozdělení, to znamená, že nejvíce hodnot se pohybuje kolem střední hodnoty a čím více se chybové údaje vzdalují této hodnoty, je jejich výskyt nižší.

směrodatnou odchylku budeme počítat podle vzorce:

$$\sigma = \sqrt{\frac{1}{d}\sum_{i=0}^{d}(x_i - E(x))^2}, \tag{24}$$

kde σ je směrodatná odchylka a $E(x)$ střední hodnota, vypočítaná podle vzorce 23. Za x_i budeme opět dosazovat absolutní chybu měření z tabulky.

Nyní můžeme tedy dosadit hodnoty z tabulek naměřených dat. Vypočítáme střední hodnoty pro všechny měřená místa. Tyto střední hodnoty můžeme poté porovnat.

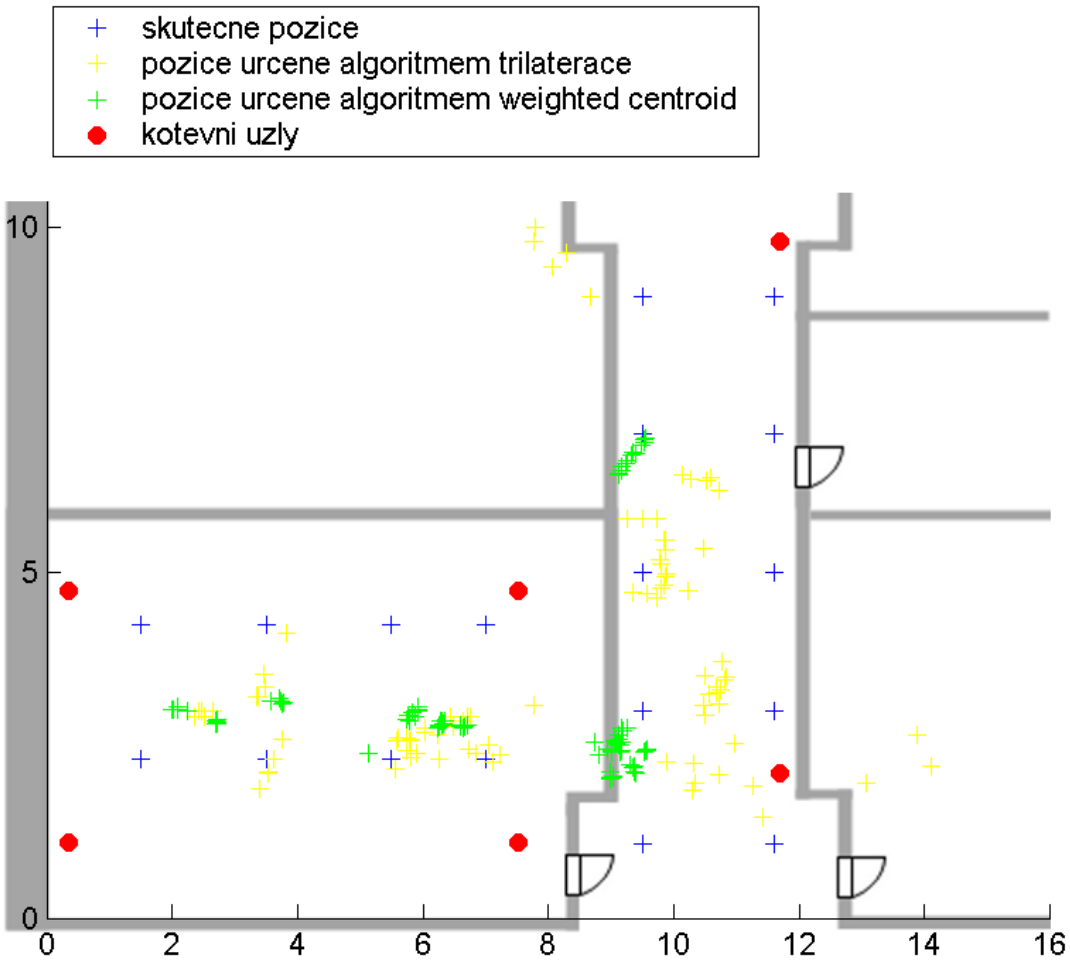
Po dosazení a výpočtu dostaneme tyto hodnoty:

Kotevní uzly rozmístěny jen v laboratoři a před laboratoří				
Místo měření	Měření v laboratoři		Měření na chodbě	
Metoda lokalizace	trilaterace	WCA	trilaterace	WCA
Střední hodnota chyby [m]	1,76	1,35	2,21	2,61
Směrodatná odchylka [m]	0,5	0,7	1,05	1,19
Rozmezí chyby [m]	1,26-2,26	0,65-2,05	1,16-3,26	1,42-3,80
Kotevní uzly rozmístěny po celém patře				
Místo měření	Měření po celé délce chodby			
Metoda lokalizace	trilaterace		WCA	
Střední hodnota chyby [m]	17,34		6,33	
Směrodatná odchylka [m]	14,36		4,88	
Rozmezí chyby [m]	2,98-31,69		1,45-11,2	

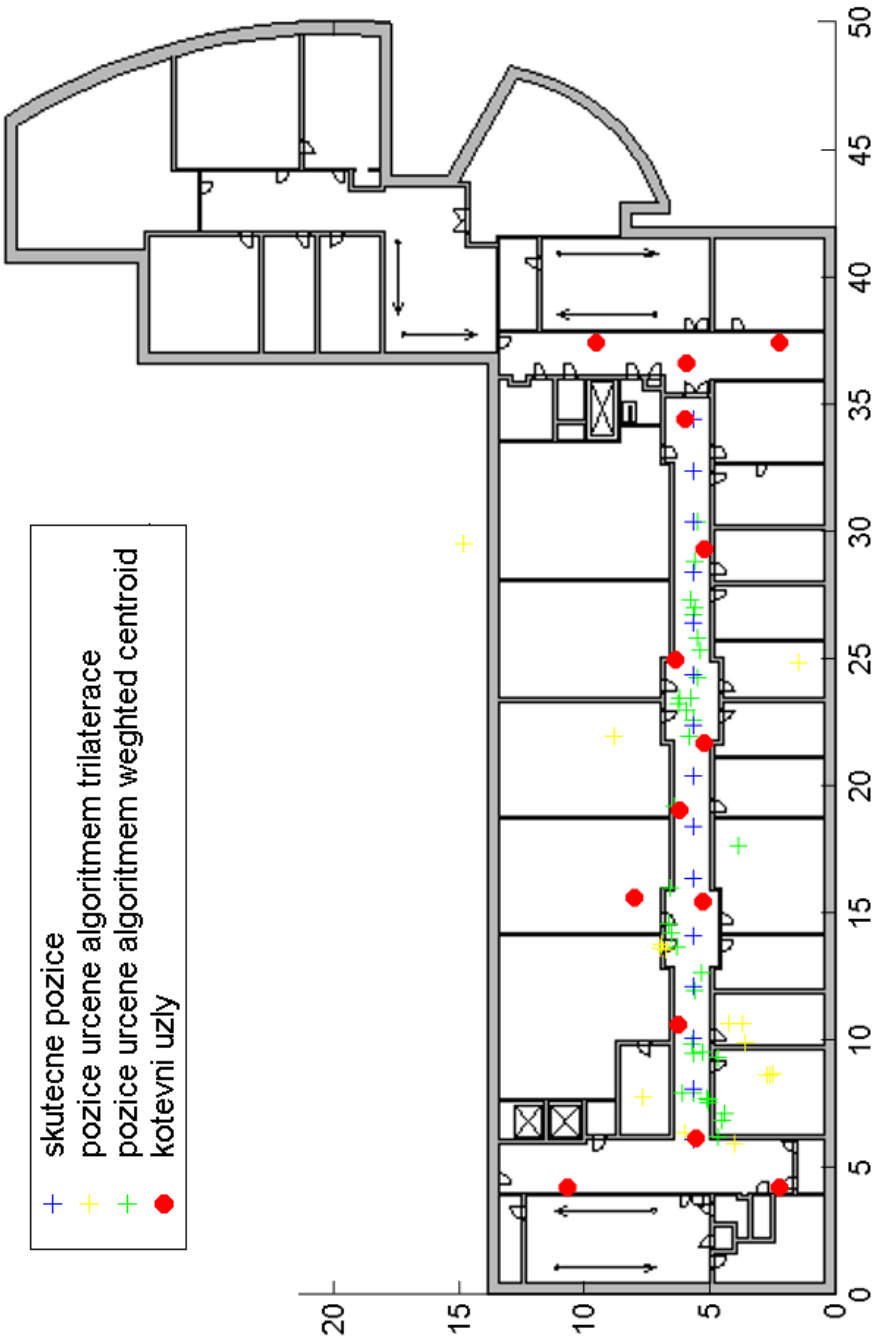
Pokud se podíváme na měření v laboratoři a přilehlé části, tak vidíme, že oba algoritmy se nijak významně neliší. V místnosti má menší střední chybu algoritmus weighted centroid, kdežto na chodbě, kde dochází k více odrazům signálu a většímu útlumu signálu z místnosti má tento algoritmus větší střední hodnotu i směrodatnou odchylku. Maximální statistická chyba byla v tomto případě 3,80 metru. Měření v místnosti, kde nedochází k velkým útlumům v podobě stěn a dveří má maximální chybovost dokonce 2,26 metru. V tomto případě měly kotevní uzly mezi sebou vzdálenost průměrně 6 metrů.

V druhém případě, kdy byly kotevní uzly rozmístěny po celé délce chodby a dalších místech ve třetím podlaží, se oba algoritmy výrazně lišily. Je to dáno jejich metodikou určování pozice. Algoritmu trilaterace, který měl velkou střední hodnotu chybovosti a jeho maximální chyba byla až 31,69 metru určuje pozici ze vzdálenosti od tří okolních uzlů a hledá pozici, odkud má k těmto uzlům změřenou vzdálenost. Weighted centroid naproti tomu váhuje a průměruje jednotlivé pozice kotevních uzlů, můžeme proto dopředu vědět, že výsledná pozice bude ležet v oblasti ohraničené kotevními uzly a nikde jinde. Střední hodnota chybovosti tohoto algoritmu je 6,33 metru a jeho maximální chyba je 11,2 metru. Jak si však můžeme všimnout na Obrázku 27, jsou zde hluchá místa, která způsobila tuto výchylku. Přesnost určení pozice můžeme snížit hustším pokrytím prostoru senzorovými uzly. Tento fakt nám potvrzuje i měření v laboratoři, kdy kotevní uzly byly rozmístěny na menší ploše.

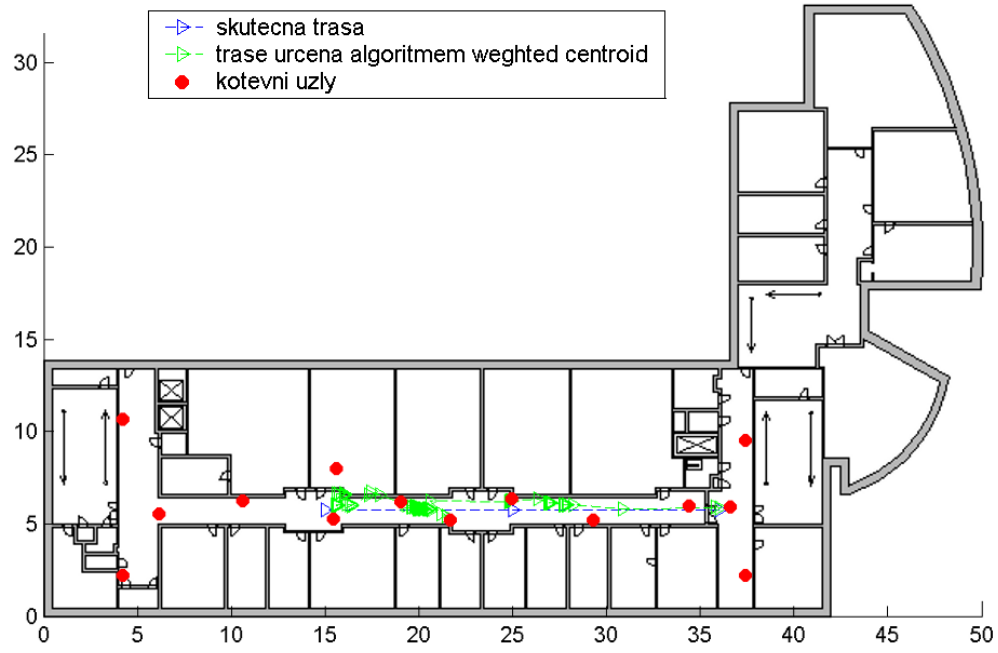
Poslední měření bylo provedeno pro zjištění, zda je navržený systém použitelný v praxi pro účel, ke kterému byl navrhnut. Způsob měření byl popsán výše. Toto měření bylo prováděno jen s algoritmem weighted centroid, který vyšel z předchozích měření nejlépe. Výsledné grafické znázornění naměřených dat můžeme vidět na Obrázku 28a a Obrázku 28b. Z těchto obrázků můžeme vidět, že systém téměř kopíroval trasu, kterou se pohyboval člověk s mobilním uzlem v ruce.



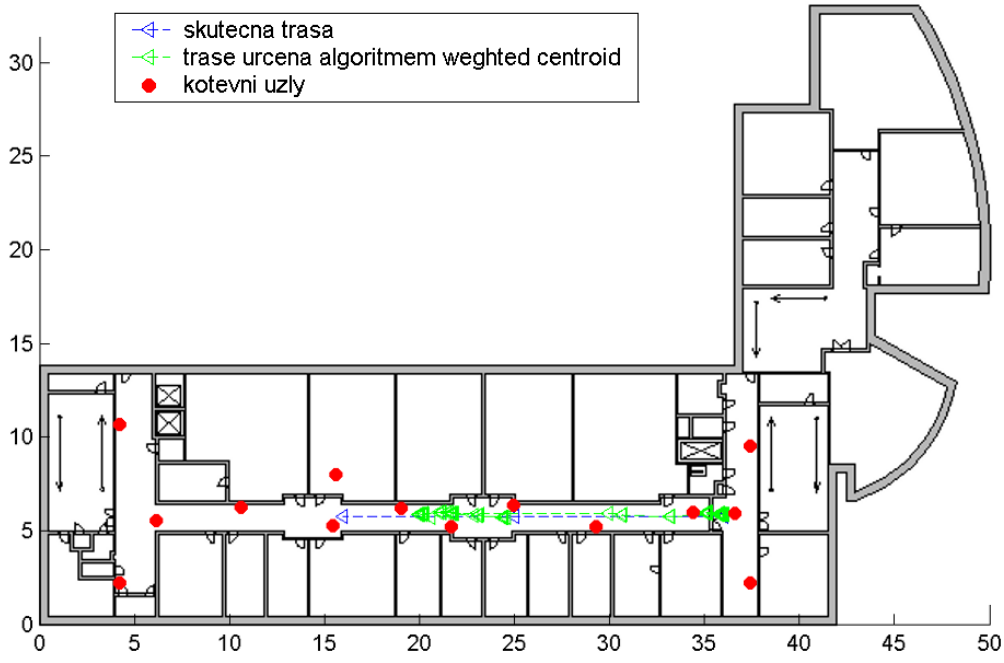
Obrázek 26: Půdorys laboratoře a přilehlé chodby s vyznačenými naměřenými daty. Modré číslice značí jednotlivé pozice, na které byl postupně umisťován mobilní uzel. Žluté a zelené číslice označují pozice, určené oběma porovnávanými algoritmy. Červeně jsou označeny pozice kotevních uzlů.



Obrázek 27: Půdorys 3. podlaží budovy PA-118 vyznačenými naměřenými daty. Modré číslice značí jednotlivé pozice, na které byl postupně umisťován mobilní uzel. Žluté a zelené číslice označují pozice, určené oběma porovnávanými algoritmy. Červeně jsou označeny pozice kotevních uzlů.



(a)



(b)

Obrázek 28: Půdorys 3. podlaží se znázorněním lokalizace pohybujícího se objektu. Modře je znázorněna trasa pohybující se osoby s mobilním uzlem a zeleně je znázorněna trasa, určená algoritmem weighted centroid.

6 Závěr

Cílem této diplomové práce bylo navrhnout bezdrátovou senzorovou síť, která bude sloužit ke sledování pohybu osob a věcí v budově. Tohoto cíle bylo dosaženo. Práce dále obsahuje rozbor požadavků, kladených na navrhovanou síť, výběr lokalizačních metod, realizaci navržené sítě a způsob jejího testování.

V první kapitole jsme se seznámili se základními vlastnostmi a aplikacemi bezdrátových senzorových sítí. Byly zde popsány různé existující metody lokalizace a monitorování v senzorových sítích.

V následující kapitole jsme přešli k návrhu vlastní sítě. Tato síť se skládá z kotevních uzlů, které mají pevně dány souřadnice, hledaných mobilních uzlů, které se k těmto kotevním uzlům lokalizují a základnové stanice. Dále byl vybrán hardware, pomocí kterého bude senzorová síť realizována. Porovnávali jsme několik senzorových modulů, ze kterých byl vybrán produkt **IRIS** od firmy Crossbow.

V této kapitole byla také navržena architektura sítě. Tato architektura zahrnuje topologii typu **mesh**. Pro rádiovou komunikaci v síti byl zvolen standard **Xmesh**. Služba, kterou síť zajišťuje, je monitorování pozice pohybujících se objektů, které budou mít na sobě připevněn sledovaný mobilní uzel sítě.

Dále byl v práci určen propagační model prostředí, ve kterém se se senzorová síť nachází. Tento statický propagační model byl poté převeden do dynamické podoby prací Miroslava Botty [14], který svou diplomovou prací spolupracoval na tomto projektu.

Praktická část práce se věnuje jednotlivým částem senzorové sítě, které se zapojují do procesu přenosu dat a lokalizace. Tyto samostatné části jsou:

- senzorová síť,
- výchozí brána sítě,
- server a
- koncová aplikace.

Senzorová síť se skládá ze senzorových uzlů již uvedeného hardwaru. Tyto uzly sbírají data a posílají na výchozí bránu. Výchozí brána je počítač, ke kterému je připojena základnová stanice sítě. Na tomto počítači byla vytvořena aplikace Forwarder, která data ze senzorové sítě přeposílá v podobě TCP paketu na server. Na serveru byla vytvořena serverová část aplikace, která tyto pakety zpracovává. Každý paket uloží do databáze a poté spustí lokalizační algoritmus. V této práci byly porovnávány dva lokalizační algoritmy. První z nich byl algoritmus trilaterace, kdy poloha hledaného uzlu byla vypočítána ze znalosti polohy nejbližších tří kotevních uzlů. Druhou metodou lokalizace byl algoritmus weighted centroid. Tato metoda spočívá v přiřazení váhy každému kotevnímu uzlu, podle jeho změřené vzdálenosti od sledovaného uzlu. Z takto váhovaných souřadnic se poté udělá průměr, který udává výsledné souřadnice hledaného uzlu.

Uvedené dva algoritmy byly srovnávány v poslední části práce. Byly provedeny dvě různá měření. První měření bylo prováděno jen v jedné místnosti. V tomto případě nebyl výrazný rozdíl mezi porovnávanými algoritmy a chybovost určení pozice se pohybovala v rozmezí 0,65 až 3,80 metru. Další typ měření byl proveden po rozmístění kotevních uzlů na větší ploše. V tomto případě vykazoval menší chybovost algoritmus weighted centroid. Tato chybovost však byla větší, než v prvním případě. Konkrétně rozmezí chybného určení pozice bylo pro tento algoritmus 1,45 až 11,2 metru. Z těchto měření vyplývá, že chybovost určování pozice je možné snížit hustším pokrýváním sledované oblasti kotevními uzly. Algoritmus **Weighted Centroid vykazoval lepší vlastnosti** během srovnávání a proto byl vybrán pro lokalizaci v navržené síti.

Uvedený algoritmus byl použit v posledním případě při sledování pohybujícího se objektu. Tento algoritmus téměř bezchybně kopíroval trasu pohybující se osoby se sledovaným mobilním uzlem.

6.1 Výhled do budoucna

Jak už bylo uvedeno, snížení chybovosti určování pozice spočívá v umisťování kotevních uzlů v menších vzdálenostech. Tím dojde k hustějšímu pokrytí sledované oblasti kotevními uzly a lokalizační algoritmus bude moci určovat pozice s menší odchylkou. Dále se počítá s rozšířením senzorové sítě do dalších podlaží budovy. Rozšířením monitorovací sítě do dalších podlaží se také dosáhne lepších výsledků díky tomu, že sledovaný uzel bude mít dosah k více kotevním uzlům. Dalšími možnostmi pro vylepšení jsou:

- pro rádiovou komunikaci použít některý z rozšířenějších standardů, jako je ZigBee či 6LoWPAN,
- možnost přenášet další data, jako jsou například informace o teplotě v okolí,
- rozšíření databáze o další tabulky pro ukládání těchto informací,
- použití jiného hardware, který bude mít přesnější metody pro určování vzdálenosti,
- častější přeposílání informací mezi uzly, čímž by systém mohl reagovat rychleji na nastalé změny.

7 Literatura

[1] Born A. Niemeyer F. SCchweide M. Bill R., *On Distance Estimation based on Radio Propagation Models and Outlier Detection for Indoor Localization in Wireless Geosensor Networks, In International Conference on Indoor Positioning and indoor navigation*, IPIN2010, Zurich 2010.

[2] Lujia Wang, Chao Hu, Longqiang Tian and Max Q.-H Meng, *A Novel 5-Dimensions RF Signal Strength Indoor Localization Method Based on Multipath Propagation*, IPIN2010, Zurich 2010.

[3] Jia T. Beuhrer R.M., *Cooperative Indoor localization, In International Conference on Indoor Positioning and indoor navigation*, IPIN 2010, Zurich 2010.

[4] Andrei Papliatseyeu, Aleksandar Matic, Venet Osmani, Oscar Mayora-Ibarra, *Indoor positioning using off-the-shelf FM radio devices*, IPIN2010, Zurich 2010.

[5] Alexander Born, Member IEEE, Frank Niemeyer, Mario Schwiede and Ralf Bill, *On Distance Estimation based on Radio Propagation Models and Outlier Detection for Indoor Localization in Wireless Geosensor Networks*, IPIN2010, Zurich 2010.

[6] Zhao F. Gubias L., *Wireless Sensor Networks: An information approach*, 2004 Elsevier Inc. ISBN-13: 978-1-55680-914-3.

[7] Dressler F., *Self-Organisation in sensor and actor networks*, 2007 John Willey and Sons Ltd. ISBN: 978-0-470-02820-9.

[8] Stojmenovic I., *Handbook of sensor networks : algorithms and architectures*, 2005 John Willey and Sons Inc. ISBN-13: 978-0-471-68472-5.

[9] Hua-Wen Tsai CHih-Ping CHu Tzung-Shi CHen, *Mobile object tracking in wireless sensor networks*, Elsevier B.V. 2007.

[10] Jara, A.J. Zamora, M.A. Skarmeta, A.F.G., *HWSN6: Hospital Wireless Sensor Networks Based on 6LoWPAN Technology: Mobility and Fault Tolerance Management.*

[11] G. Virone, A. Wood, L. Selavo, Q. Cao, L. Fang, T. Doan, Z. He, R. Stoleru, S. Lin, J.A. Stankovic, *An Advanced Wireless Sensor Network for Health Monitoring* , Department of Computer Science, University of Virginia.

[12] Tsenka Stoyanova, Fotis Kerasiotis, Aggeliki Prayati, George Papadopoulos, *Evaluation of impact factors on RSS accuracy for localization and tracking applications in sensor networks*, Springer Science+Business Media, LLC 2009.

[13] Vajsar P., *Webové rozhraní pro monitoring senzorového pole.* diplomová práce, Vysoké učení technické v Brně. Fakulta elektrotechniky a komunikačních technologií, 2010. 60 s., 20 s. příloh. Vedoucí práce Ing. Patrik Morávek.

[14] Botta, M., *Optimalizace odhadu vzdálenosti v bezdrátové ad-hoc síti.* diplomová práce, Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2011. Vedoucí diplomové práce Ing. Milan Šimek, Ph.D.

[15] Žoldoš, Petr, *Aplikácia pre zobrazenie modelu bezdrôtovej siete.* diplomová práce, Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikácií, 2011. 57 s. Vedoucí práce Ing. Pavel Vajsar.

[16] ITU-R Recommendations, *Propagation Data and Prediction Methods for the Planning of Indoor Radiocommunication Systems and Radio Local Area Networks in the Frequency range 900MHz to 100GHz*, ITU-R P.1238-2, Geneva, 2001.

[17] Aly I. El-Osery, Member, IEEE, Wael Abd-Almageed, Member, IEEE, and Moustafa Youssef, Member, IEEE, *Calibration-Free RF-Based Localization Algorithm for Sensor Actuator Networks using Particle Filters.*

[18] Kay Roemer and Friedemann Mattern, *The Design Space of Wireless Sensor Networks*, Institute for Pervasive Computing ETH Zurich.

[19] Jan Blumenthal, Ralf Grossmann, Frank Golatowski, Dirk Timmermann, *Weighted Centroid Localization in Zigbee-based Sensor Networks.*, 2007 University of Rostock.

[20] Jiří Rybička, *LaTeX pro začátečníky.*, 1995 Brno, nakladatelství Konvoj, ISBN: 80-7302-049-1.

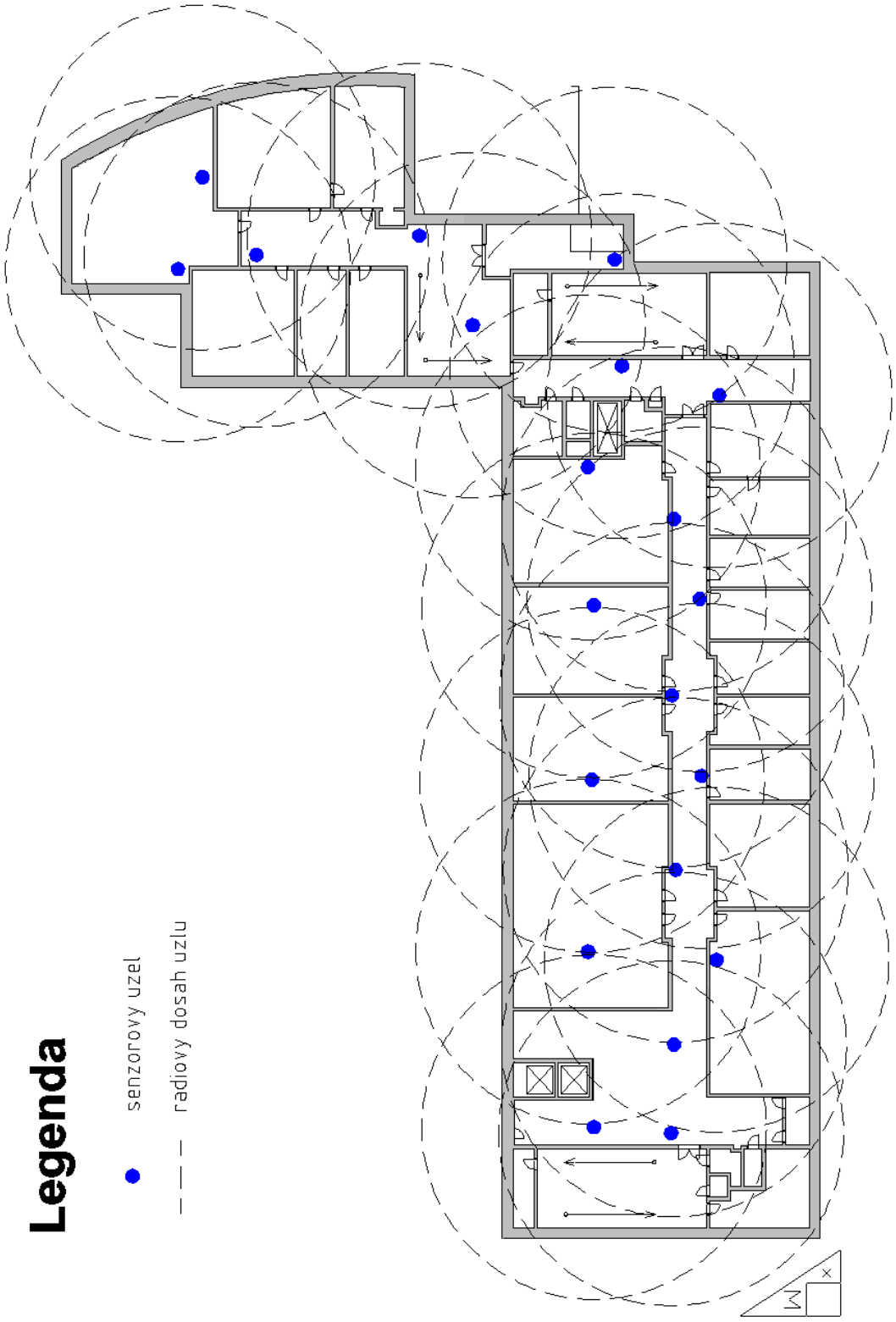
[21] *XMesh User's Manual*,Revision D, April 2007 2005-2007 Crossbow Technology, Inc.

[22] *Standard IEEE 802.25.4 na serveru IEEE.org.* dostupné z WWW: `http://standards.ieee.org/about/get/802/802.15.html`

[23] John S. Seybold, Ph.D., *Introduction to RF Propagation*, 2005 by John Wiley & Sons, Inc. ISBN-13 978-0-471-65596-1.

[24] *Manuál senzorového uzlu IRIS firmy Crossbow.* dostupné z WWW: `http://www.dinesgroup.org/projects/images/pdf_files/iris_datasheet.pdf`

Přílohy



Obrázek 29: Mapa pokrytí signálem.

Tabulka 4: Měření v laboratoři, skutečné, určené souřadnice a jejich odchylka pro algoritmus trilaterace.

Skutečné souřadnice			Určené souřadnice			Odchylka souřadnic			Absolutní odchylka
RealX	RealY	RealZ	PosX	PosY	PosZ	FailX	FailY	FailZ	FailAbs
Pozice 1									
1.5	2.3	3.3	3.459	3.543	3.300	1.959	1.243	0	2.321
1.5	2.3	3.3	3.375	3.200	3.300	1.875	0.900	0	2.080
1.5	2.3	3.3	3.490	3.338	3.300	1.990	1.038	0	2.244
1.5	2.3	3.3	3.831	4.120	3.300	2.331	1.820	0	2.957
1.5	2.3	3.3	3.353	3.201	3.300	1.853	0.901	0	2.061
Pozice 2									
3.5	2.3	3.3	5.614	2.601	3.300	2.114	0.301	0	2.135
3.5	2.3	3.3	5.814	2.594	3.300	2.314	0.294	0	2.333
3.5	2.3	3.3	5.896	2.582	3.300	2.396	0.282	0	2.413
3.5	2.3	3.3	5.895	2.597	3.300	2.395	0.297	0	2.414
3.5	2.3	3.3	5.841	2.543	3.300	2.341	0.243	0	2.354
Pozice 3									
5.5	2.3	3.3	7.111	2.268	3.300	1.611	0.032	0	1.611
5.5	2.3	3.3	6.845	2.382	3.300	1.345	0.082	0	1.348
5.5	2.3	3.3	6.726	2.444	3.300	1.226	0.144	0	1.235
5.5	2.3	3.3	7.041	2.511	3.300	1.541	0.211	0	1.556
5.5	2.3	3.3	7.222	2.364	3.300	1.722	0.064	0	1.723
Pozice 4									
7	2.3	3.3	6.328	2.778	3.300	0.672	0.478	0	0.825
7	2.3	3.3	6.240	2.663	3.300	0.760	0.363	0	0.843
7	2.3	3.3	6.477	2.766	3.300	0.523	0.466	0	0.700
7	2.3	3.3	6.037	2.761	3.300	0.963	0.461	0	1.067
7	2.3	3.3	6.036	2.687	3.300	0.964	0.387	0	1.038

Tabulka 5: Měření v laboratoři, skutečné, určené souřadnice a jejich odchylka(pokračování).

Skutečné souřadnice			Určené souřadnice			Odchylka souřadnic			Absolutní odchylka
RealX	RealY	RealZ	PosX	PosY	PosZ	FailX	FailY	FailZ	FailAbs
Pozice 5									
7	4.25	3.3	6.753	2.915	3.300	0.247	1.335	0	1.358
7	4.25	3.3	7.764	3.074	3.300	0.764	1.176	0	1.402
7	4.25	3.3	6.440	2.915	3.300	0.560	1.335	0	1.448
7	4.25	3.3	6.703	2.915	3.300	0.297	1.335	0	1.368
7	4.25	3.3	6.651	2.852	3.300	0.349	1.398	0	1.441
Pozice 6									
5.5	4.25	3.3	5.755	2.418	3.300	0.255	1.832	0	1.850
5.5	4.25	3.3	6.263	2.303	3.300	0.763	1.947	0	2.091
5.5	4.25	3.3	5.546	2.164	3.300	0.046	2.086	0	2.086
5.5	4.25	3.3	5.889	2.390	3.300	0.389	1.860	0	1.900
5.5	4.25	3.3	5.799	2.326	3.300	0.299	1.924	0	1.947
Pozice 7									
3.5	4.25	3.3	3.761	2.581	3.300	0.261	1.669	0	1.689
3.5	4.25	3.3	3.401	1.868	3.300	0.099	2.382	0	2.384
3.5	4.25	3.3	3.627	2.315	3.300	0.127	1.935	0	1.939
3.5	4.25	3.3	3.528	2.120	3.300	0.028	2.130	0	2.131
3.5	4.25	3.3	3.519	2.102	3.300	0.019	2.148	0	2.148
Pozice 8									
1.5	4.25	3.3	2.430	2.986	3.300	0.930	1.264	0	1.569
1.5	4.25	3.3	2.636	2.989	3.300	1.136	1.261	0	1.697
1.5	4.25	3.3	2.361	2.915	3.300	0.861	1.335	0	1.589
1.5	4.25	3.3	2.517	2.915	3.300	1.017	1.335	0	1.678
1.5	4.25	3.3	2.459	2.989	3.300	0.959	1.261	0	1.584

Tabulka 6: Měření na chodbě, skutečné, určené souřadnice a jejich odchylka pro algoritmus trilaterace.

Skutečné souřadnice			Určené souřadnice			Odchylka souřadnic			Absolutní odchylka
RealX	RealY	RealZ	PosX	PosY	PosZ	FailX	FailY	FailZ	FailAbs
Pozice 1									
9.5	1.07	3.3	13.880	2.658	3.300	4.380	1.588	0	4.659
9.5	1.07	3.3	14.114	2.214	3.300	4.614	1.144	0	4.754
9.5	1.07	3.3	11.259	1.916	3.300	1.759	0.846	0	1.952
9.5	1.07	3.3	13.078	1.962	3.300	3.578	0.892	0	3.687
9.5	1.07	3.3	11.417	1.469	3.300	1.917	0.399	0	1.958
Pozice 2									
9.5	3	3.3	10.985	2.533	3.300	1.485	0.467	0	1.557
9.5	3	3.3	10.809	3.449	3.300	1.309	0.449	0	1.384
9.5	3	3.3	10.666	3.341	3.300	1.166	0.341	0	1.215
9.5	3	3.3	10.487	2.938	3.300	0.987	0.062	0	0.989
9.5	3	3.3	10.472	3.086	3.300	0.972	0.086	0	0.976
Pozice 3									
9.5	5	3.3	10.835	3.475	3.300	1.335	1.525	0	2.027
9.5	5	3.3	10.734	3.097	3.300	1.234	1.903	0	2.268
9.5	5	3.3	10.675	3.262	3.300	1.175	1.738	0	2.098
9.5	5	3.3	10.737	3.298	3.300	1.237	1.702	0	2.104
9.5	5	3.3	10.565	3.249	3.300	1.065	1.751	0	2.050
Pozice 4									
9.5	7	3.3	9.499	5.795	3.300	0.001	1.205	0	1.205
9.5	7	3.3	9.266	5.783	3.300	0.234	1.217	0	1.239
9.5	7	3.3	9.265	5.787	3.300	0.235	1.213	0	1.235
9.5	7	3.3	9.851	5.483	3.300	0.351	1.517	0	1.557
9.5	7	3.3	9.727	5.787	3.300	0.227	1.213	0	1.234
Pozice 5									
9.5	9	3.3	9.873	5.484	3.300	0.373	3.516	0	3.535
9.5	9	3.3	9.857	5.340	3.300	0.357	3.660	0	3.678
9.5	9	3.3	9.769	5.179	3.300	0.269	3.821	0	3.831
9.5	9	3.3	9.808	5.121	3.300	0.308	3.879	0	3.891
9.5	9	3.3	10.471	5.361	3.300	0.971	3.639	0	3.766

Tabulka 7: Měření na chodbě, skutečné, určené souřadnice a jejich odchylka pro algoritmus trilaterace.

Skutečné souřadnice			Určené souřadnice			Odchylka souřadnic			Absolutní odchylka
RealX	RealY	RealZ	PosX	PosY	PosZ	FailX	FailY	FailZ	FailAbs
Pozice 6									
11.6	1.07	3.3	9.882	2.268	3.300	1.718	1.198	0	2.094
11.6	1.07	3.3	10.304	1.846	3.300	1.296	0.776	0	1.511
11.6	1.07	3.3	10.721	2.068	3.300	0.879	0.998	0	1.330
11.6	1.07	3.3	10.308	2.242	3.300	1.292	1.172	0	1.744
11.6	1.07	3.3	10.341	1.949	3.300	1.259	0.879	0	1.535
Pozice 7									
11.6	3	3.3	10.215	4.736	3.300	1.385	1.736	0	2.221
11.6	3	3.3	9.789	4.771	3.300	1.811	1.771	0	2.533
11.6	3	3.3	10.771	3.720	3.300	0.829	0.720	0	1.098
11.6	3	3.3	10.494	3.503	3.300	1.106	0.503	0	1.215
11.6	3	3.3	9.738	4.633	3.300	1.862	1.633	0	2.476
Pozice 8									
11.6	5	3.3	9.567	4.698	3.300	2.033	0.302	0	2.056
11.6	5	3.3	9.845	4.824	3.300	1.755	0.176	0	1.764
11.6	5	3.3	9.351	4.710	3.300	2.249	0.290	0	2.268
11.6	5	3.3	9.894	4.987	3.300	1.706	0.013	0	1.706
11.6	5	3.3	9.855	4.950	3.300	1.745	0.050	0	1.745
Pozice 9									
11.6	7	3.3	10.530	6.334	3.300	1.070	0.666	0	1.261
11.6	7	3.3	10.717	6.194	3.300	0.883	0.806	0	1.196
11.6	7	3.3	10.278	6.360	3.300	1.322	0.640	0	1.469
11.6	7	3.3	10.604	6.385	3.300	0.996	0.615	0	1.171
11.6	7	3.3	10.153	6.410	3.300	1.447	0.590	0	1.563
Pozice 10									
11.6	9	3.3	8.302	9.632	3.300	3.298	0.632	0	3.358
11.6	9	3.3	7.790	9.999	3.300	3.810	0.999	0	3.939
11.6	9	3.3	8.056	9.427	3.300	3.544	0.427	0	3.570
11.6	9	3.3	7.776	9.798	3.300	3.824	0.798	0	3.907
11.6	9	3.3	8.678	8.999	3.300	2.922	0.001	0	2.922

Tabulka 8: Měření na v laboratoři, skutečné, určené souřadnice a jejich odchylka pro algoritmus weighted centroid.

Skutečné souřadnice		Určené souřadnice		Odchylka souřadnic		Absolutní odchylka
RealX	RealY	PosX	PosY	FailX	FailY	FailAbs
Pozice 1						
1.5	2.3	2.03	3.02	0.53	0.72	0.89
1.5	2.3	2	3.01	0.5	0.71	0.86
1.5	2.3	2.08	3.01	0.58	0.71	0.91
1.5	2.3	2.08	3.05	0.58	0.75	0.95
1.5	2.3	2.24	3	0.74	0.7	1.02
Pozice 2						
3.5	2.3	3.58	3.14	0.08	0.84	0.84
3.5	2.3	3.78	3.13	0.28	0.83	0.87
3.5	2.3	3.74	3.14	0.24	0.84	0.87
3.5	2.3	3.7	3.18	0.2	0.88	0.91
3.5	2.3	3.76	3.1	0.26	0.8	0.84
Pozice 3						
5.5	2.3	5.14	2.38	0.36	0.08	0.37
5.5	2.3	5.78	2.86	0.28	0.56	0.62
5.5	2.3	5.77	2.93	0.27	0.63	0.68
5.5	2.3	5.73	2.87	0.23	0.57	0.61
5.5	2.3	5.88	2.91	0.38	0.61	0.72
Pozice 4						
7	2.3	6.6	2.77	0.4	0.47	0.62
7	2.3	6.7	2.8	0.3	0.5	0.59
7	2.3	6.59	2.79	0.41	0.49	0.64
7	2.3	6.65	2.76	0.35	0.46	0.57
7	2.3	6.67	2.78	0.33	0.48	0.58

Tabulka 9: Měření na chodbě, skutečné, určené souřadnice a jejich odchylka pro algoritmus weighted centroid.

Skutečné souřadnice		Určené souřadnice		Odchylka souřadnic		Absolutní odchylka
RealX	RealY	PosX	PosY	FailX	FailY	FailAbs
Pozice 5						
7	4.25	6.33	2.82	0.67	1.43	1.57
7	4.25	6.35	2.86	0.65	1.39	1.53
7	4.25	6.29	2.85	0.71	1.4	1.57
7	4.25	6.29	2.85	0.71	1.4	1.57
7	4.25	6.31	2.85	0.69	1.4	1.56
Pozice 6						
5.5	4.25	6.27	2.8	0.77	1.45	1.64
5.5	4.25	6.3	2.8	0.8	1.45	1.66
5.5	4.25	6.3	2.79	0.8	1.46	1.67
5.5	4.25	6.25	2.78	0.75	1.47	1.65
5.5	4.25	6.24	2.75	0.74	1.5	1.67
Pozice 7						
3.5	4.25	5.93	3.06	2.43	1.19	2.7
3.5	4.25	5.92	3	2.42	1.25	2.72
3.5	4.25	5.92	3.02	2.42	1.23	2.71
3.5	4.25	5.92	3.01	2.42	1.24	2.72
3.5	4.25	5.84	2.97	2.34	1.28	2.67
Pozice 8						
1.5	4.25	2.71	2.84	1.21	1.41	1.86
1.5	4.25	2.69	2.85	1.19	1.4	1.84
1.5	4.25	2.72	2.88	1.22	1.37	1.84
1.5	4.25	2.69	2.82	1.19	1.43	1.86
1.5	4.25	2.71	2.82	1.21	1.43	1.87

Tabulka 10: Měření na chodbě, skutečné, určené souřadnice a jejich odchylka pro algoritmus weighted centroid.

Skutečné souřadnice		Určené souřadnice		Odchylka souřadnic		Absolutní odchylka
RealX	RealY	PosX	PosY	FailX	FailY	FailAbs
Pozice 1						
9.5	1.07	9.34	2.17	0.16	1.1	1.11
9.5	1.07	9.3	2.23	0.2	1.16	1.17
9.5	1.07	9.37	2.21	0.13	1.14	1.14
9.5	1.07	9.38	2.11	0.12	1.04	1.04
9.5	1.07	9.4	2.09	0.1	1.02	1.02
Pozice 2						
9.5	3	9.14	2.5	0.36	0.5	0.62
9.5	3	9.05	2.57	0.45	0.43	0.62
9.5	3	8.8	2.36	0.7	0.64	0.95
9.5	3	9.02	2.4	0.48	0.6	0.77
9.5	3	9	2.42	0.5	0.58	0.77
Pozice 3						
9.5	5	9.55	2.42	0.05	2.58	2.59
9.5	5	9.58	2.44	0.08	2.56	2.56
9.5	5	9.53	2.4	0.03	2.6	2.6
9.5	5	9.57	2.42	0.07	2.58	2.58
9.5	5	9.55	2.42	0.05	2.58	2.58
Pozice 4						
9.5	7	9.13	2.64	0.37	4.36	4.38
9.5	7	9.2	2.71	0.3	4.29	4.3
9.5	7	9.17	2.76	0.33	4.24	4.26
9.5	7	9.25	2.76	0.25	4.24	4.25
9.5	7	9.2	2.71	0.3	4.29	4.3
Pozice 5						
9.5	9	9.39	6.73	0.11	2.27	2.28
9.5	9	9.18	6.48	0.32	2.52	2.54
9.5	9	9.13	6.44	0.37	2.56	2.59
9.5	9	9.12	6.42	0.38	2.58	2.61
9.5	9	9.03	2.05	0.47	6.95	6.97

Tabulka 11: Měření na chodbě, skutečné, určené souřadnice a jejich odchylka pro algoritmus weighted centroid.

Skutečné souřadnice		Určené souřadnice		Odchylka souřadnic		Absolutní odchylka
RealX	RealY	PosX	PosY	FailX	FailY	FailAbs
Pozice 6						
11.6	1.07	8.99	2.01	2.61	0.94	2.77
11.6	1.07	8.98	2.03	2.62	0.96	2.79
11.6	1.07	9.04	2.03	2.56	0.96	2.73
11.6	1.07	8.99	2.12	2.61	1.05	2.81
11.6	1.07	9.13	2.51	2.47	1.44	2.86
Pozice 7						
11.6	3	9.17	2.55	2.43	0.45	2.47
11.6	3	9.13	2.58	2.47	0.42	2.5
11.6	3	9.08	2.52	2.52	0.48	2.57
11.6	3	9.11	2.55	2.49	0.45	2.53
11.6	3	8.75	2.54	2.85	0.46	2.89
Pozice 8						
11.6	5	8.94	2.44	2.66	2.56	3.69
11.6	5	9.16	2.42	2.44	2.58	3.55
11.6	5	9.14	2.42	2.46	2.58	3.57
11.6	5	9.15	2.41	2.45	2.59	3.57
11.6	5	9.24	6.58	2.36	1.58	2.84
Pozice 9						
11.6	7	9.26	6.61	2.34	0.39	2.37
11.6	7	9.32	6.7	2.28	0.3	2.3
11.6	7	9.35	6.74	2.25	0.26	2.26
11.6	7	9.17	6.53	2.43	0.47	2.47
11.6	7	9.48	6.85	2.12	0.15	2.12
Pozice 10						
11.6	9	9.52	6.89	2.08	2.11	2.96
11.6	9	9.56	6.95	2.04	2.05	2.89
11.6	9	9.55	6.95	2.05	2.05	2.9
11.6	9	9.53	6.92	2.07	2.08	2.94

Tabulka 12: Měření po celé décle chodby ve 3. podlaží budovy PA-118.

Skutečné souřadnice		Určené souřadnice		Odchylka souřadnic		Absolutní odchylka
RealX	RealY	PosX	PosY	FailX	FailY	FailAbs
6.09	5.64	11.97	5.62	5.89	0.02	15.36
6.09	5.64	7.91	6.06	1.82	0.42	2.17
6.09	5.64	7.9	5.63	1.82	0.01	3.02
8.09	5.64	6.22	4.68	1.87	0.96	1.26
8.09	5.64	7.09	4.41	0.99	1.23	0.21
8.09	5.64	6.85	4.5	1.23	1.14	0.1
10.09	5.64	9.83	5.68	0.25	0.04	0.23
10.09	5.64	9.51	5.25	0.57	0.39	0.2
10.09	5.64	9.47	5.64	0.62	0	0.79
12.09	5.64	7.71	5.11	4.38	0.53	8.45
12.09	5.64	7.55	5.04	4.53	0.6	8.75
12.09	5.64	7.65	5.03	4.44	0.61	8.41
14.09	5.64	14.56	6.61	0.47	0.97	0.35
14.09	5.64	14.23	6.51	0.14	0.87	0.38
14.09	5.64	13.69	6.32	0.39	0.68	0.25
16.38	5.64	19.22	6.39	2.84	0.75	3.67
16.38	5.64	21.94	5.79	5.57	0.15	13.74
18.38	5.64	12.64	5.31	5.73	0.33	13.67
18.38	5.64	23.43	6.17	5.05	0.53	10.63
20.38	5.64	9.3	4.66	11.08	0.98	33.67
20.38	5.64	17.63	3.87	2.75	1.77	1.37
22.38	5.64	26.71	5.67	4.33	0.03	9.89
22.38	5.64	27.29	5.76	4.91	0.12	11.55

Tabulka 13: Měření po celé décle chodby ve 3. podlaží budovy PA-118 - dokončení.

Skutečné souřadnice		Určené souřadnice		Odchylka souřadnic		Absolutní odchylka
RealX	RealY	PosX	PosY	FailX	FailY	FailAbs
24.38	5.64	22.95	5.93	1.43	0.29	1.66
24.38	5.64	25.31	5.36	0.93	0.28	0.84
26.38	5.64	23.22	6.23	3.15	0.59	4.85
26.38	5.64	27.01	5.62	0.64	0.02	0.78
28.38	5.64	23.43	5.73	4.94	0.09	11.73
28.38	5.64	22.61	5.63	5.76	0.01	14.93
30.38	5.64	22.61	5.63	7.76	0.01	22.92
30.38	5.64	22.61	5.63	7.76	0.01	22.92
30.38	5.64	22.61	5.63	7.76	0.01	22.92
32.38	5.64	22.61	5.63	9.76	0.01	31.96
32.38	5.64	24.23	5.49	8.14	0.15	23.95
32.38	5.64	25.79	5.46	6.59	0.18	17.43
34.38	5.64	28.83	5.6	5.55	0.04	14.05
34.38	5.64	30.35	5.47	4.02	0.17	8.48