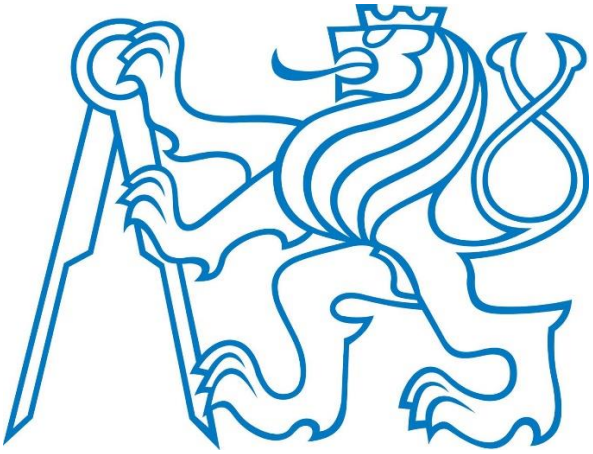
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Fakulta elektrotechnická

Katedra telekomunikační techniky



Systém domácí automatizace s využitím konceptu Internetu věcí

Household Automation Systems Using Internet of Things Paradigm

Bakalářská práce

Studijní program: Elektronika a komunikace

Vedoucí práce: doc. Ing. Leoš Boháč, Ph.D.

Filip Bělohlávek

Praha
květen 2020

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Bělohlávek** Jméno: **Filip** Osobní číslo: **474228**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra telekomunikační techniky**

Studijní program: **Elektronika a komunikace**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Systém domácí automatizace s využitím konceptu Internetu věcí

Název bakalářské práce anglicky:

Household Automation Systems Using Internet of Things Paradigm

Pokyny pro vypracování:

Vytvořte návrh architektury systému domácí automatizace založené na konceptu Internetu věcí. Identifikujte a navrhněte aplikace, které by mohl v takovém prostředí prakticky využít běžný uživatel. Následně jednu z těchto aplikací prakticky realizujte a otestujte její vlastnosti.

Seznam doporučené literatury:

[1] SPIVEY, Dwight. Home automation for dummies. Hoboken, NJ: John Wiley, [2015]. –For dummies. ISBN 1-11894-926-9.
[2] BREWER, Dennis C. Home automation made easy: do it yourself know how using UPB, Insteon, X10 and Z-Wave. Indianapolis, IN: Que, [2013]. ISBN 0-78975-124-0.

Jméno a pracoviště vedoucí(ho) bakalářské práce:

doc. Ing. Leoš Boháč, Ph.D., katedra telekomunikační techniky FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **08.01.2020** Termín odevzdání bakalářské práce: **22.05.2020**

Platnost zadání bakalářské práce: **30.09.2021**

doc. Ing. Leoš Boháč, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Mgr. Petr Páta, Ph.D. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. Datum převzetí zadání

Podpis studenta

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně, a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze, dne 20. 5. 2020 ……

Filip Bělohlávek

Poděkování

Chtěl bych poděkovat doc. Ing. Leoši Boháčovi, Ph.D. za jeho vedení a užitečné rady a poznámky, ze kterých jsem si odnesl mnoho cenných zkušeností. Dále bych chtěl poděkovat všem, kteří mě během psaní této práce podporovali.

Abstrakt

Práce pojednává zejména o IoT systémech pro domácí automatizaci. Za hlavní problém je považována nejednotnost architektur systémů domácí automatizace. Jako řešení tohoto problému je v práci abstraktně navržená architektura systémů domácí automatizace, snažící se o sjednocení dosavadních architektur do jednoho funkčního celku. Navržená architektura definuje základní principy pro návrh dílčích vrstev systému, od samotných koncových zařízení, přes návrh brány a cloudu až po mobilní aplikace. V další části se práce věnuje identifikaci aplikací v prostředí domácí automatizace. Identifikovány jsou aplikace s vysokým podílem automatizace. Následně je jedna z těchto aplikací podrobněji rozebrána a je navrženo vlastní řešení aplikace za použití SoC ESP8266. Navržené zařízení je poté otestováno a je zhodnocena jeho funkčnost.

Klíčová slova

Internet of Things, domácí automatizace, chytrá domácnost, architektura IoT, systém lokalizace osob v domácnosti

Summary

This thesis deals mainly with IoT systems for home automation. The main problem is considered to be the disunity of architectures of these systems. As a solution to this problem, an abstract architecture trying to unify existing architectures into one functional complex is designed in this thesis. Proposed architecture defines basic principles for designing individual layers of such system, from end devices through gateway and cloud design as far to mobile applications. The following part of the thesis focuses on identifying possible applications in an environment of home automation. Applications with high share of automation are identified. One of those applications is then studied more thoroughly and an own design of this application is proposed using the ESP8266 SoC. Designed device is then tested and its functionality is evaluated.

Index Terms

Internet of Things, home automation, smart home, IoT architecture, indoor positioning system

Obsah

Seznam obrázků a tabulek

Seznam zkratek

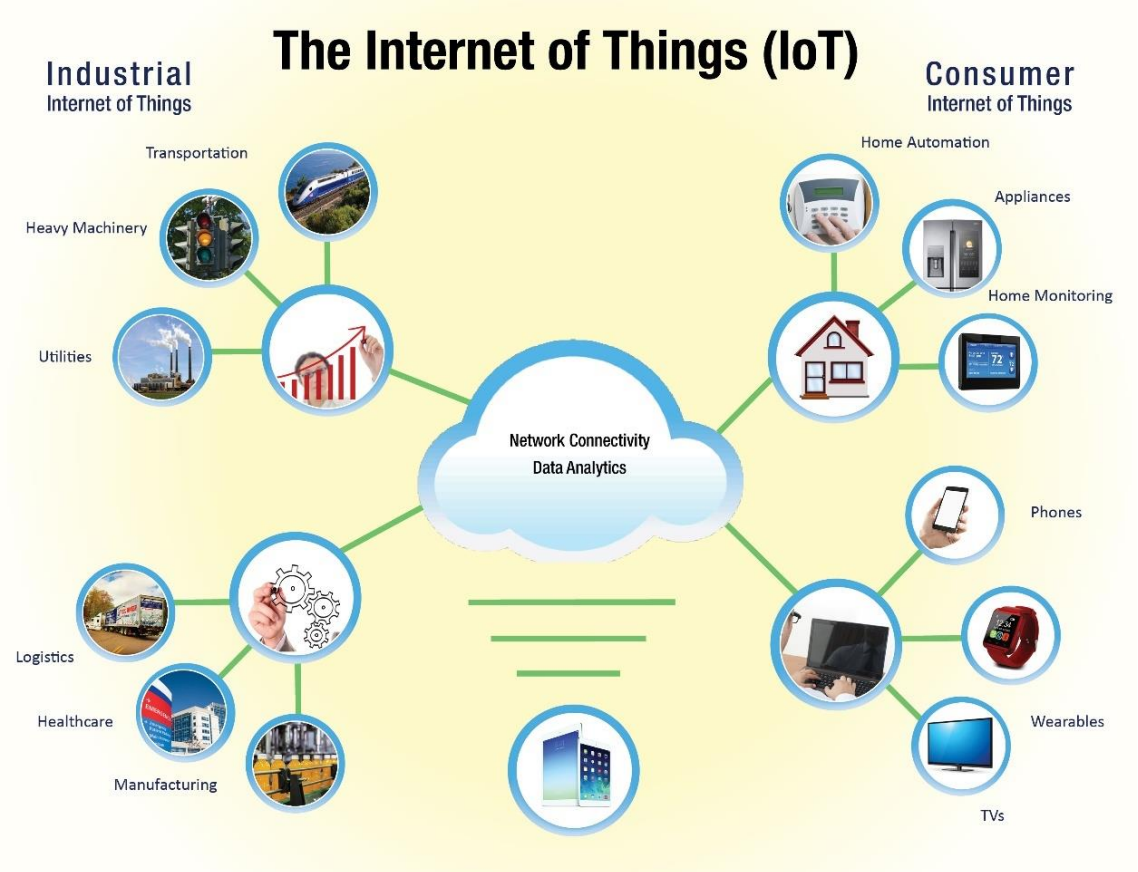
IoT	Internet of Things	Internet věcí
DDoS	Distributed Denial of Service	Distribuované odepření služby
LAN	Local Area Network	Lokální síť
WAN	Wide Area Network	Rozlehlá síť
SPOF	Single Point of Failure	Kritické místo poruchy
BLE	Bluetooth Low Energy	Nízkoenergetický Bluetooth
NAT	Network Address Translation	Překlad síťových adres
MQTT	MQ Telemetry Transport	MQ Telemetrický transport
TCP	Transmission Control Protocol	Protokol kontroly přenosu
IP	Internet Protocol	Internetový protokol
RSA	Rivest Shamir Adleman	Rivest Shamir Adleman
ECC	Elliptic curve cryptography	Kryptografie eliptických křivek
ECDH	Elliptic curve Diffie-Hellman	Diffie-Hellman s využitím eliptických křivek
MITM	Man In The Middle	Člověk uprostřed
AES	Advanced Encryption Standard	Standard pokročilého šifrování
IV	Initialization Vector	Inicializační vektor
DSA	Digital Signature Algorithm	Algoritmus digitálního podpisu
ECDSA	Elliptic Curve DSA	DSA s využitím eliptických křivek
MEMS	MicroElectroMechanical Systems	MikroElektroMechanické systémy
XML	Extensible Markup Language	Rozšiřitelný značkovací jazyk
JSON	JavaScript Object Notation	JavaScriptový objektový zápis
QoS	Quality of Service	Kvalita služby
UPS	Uninterruptible Power Supply	Zdroj nepřerušovaného napájení
ID	Identifier	Identifikátor
OS	Operating System	Operační systém
MIS	Modular Interface Socket	Modulární rozhraní
AP	Access Point	Přístupový bod
HTTPS	Hypertext Transfer Protocol Secure	Zabezpečený protokol přenosu hypertextu
DCS	Device Control Software	Software pro kontrolu zařízení
API	Application Programming Interface	Rozhraní pro programování aplikací

UID	User ID	ID uživatele
AI	Artificial Intelligence	Umělá inteligence
RGB	Red Green Blue	Červená Zelená Modrá
HMAC	Keyed-Hash Message Authentication Code	Autentizační kód zprávy s hashovací funkcí
SHA	Secure Hash Algorithm	Bezpečný hashovací algoritmus
PBKDF2	Password-Based Key Derivation Function 2	Funkce pro generaci klíče z hesla
UI	User Interface	Uživatelské prostředí
REST	Representational state transfer	Reprezentativní převod stavu
ISP	Internet Service Provider	Poskytovatel internetového připojení
LED	Light Emitting Diode	Elektroluminiscenční dioda
ToF	Time of Flight	Doba letu
WLAN	Wireless Local Area Network	Bezdrátová lokální síť
SSID	Service Set Identifier	Identifikátor bezdrátové sítě
mDNS	multicast Domain Name System	Multicast systém doménových jmen
PIR	Passivce Infrared Sensor	Pasivní infračervený senzor
DIY	Do It Yourself	Udělej si sám – kutilství
PSK	Pre-Shared Key	Předsdílený klíč
IPS	Indoor Positioning System	Systém lokalizace lidí v domácnosti
SPIFFS	SPI Flash File System	SPI flash souborový systém
AC	Alternating Current	Střídavý proud
DC	Direct Current	Stejnosměrný proud

1 Úvod

1.1 Internet věcí

Systémy založené na myšlence Internetu věcí (IoT) jsou v posledních letech na vzestupu, a tento trend se bude projevovat i v budoucnu. Primární idea je, že všechna zařízení budou mezi sebou moci komunikovat, např. sdílet svoje data ze senzorů, využívat data ze senzorů jiných zařízení, či být na dálku ovládáno určitým centrálním řídícím systémem. Předpokládá se, že rokem 2025 bude do sítě připojeno až 75 miliard těchto zařízení [1]. Využití těchto zařízení je nesmírně široké a protíná jednotlivé sféry společnosti. Ve spotřebitelské sféře se jedná především o chytrou domácnost (smart home), což jsou systémy umožňující domácí automatizaci různých spotřebičů a jiných koncových zařízení. V komerčním a průmyslovém sektoru je množina aplikací velmi široká. Od řízení a optimalizace výrobních procesů nebo sběru a analýzy logistických dat, až po zemědělství, kde je možné měřit různé veličiny, na jejichž základě lze poté automatizovaně řídit zemědělské procesy. Konkrétním přínosem těchto systémů, například právě v zemědělství, je minimalizace nákladů na práci a zvýšení efektivity činnosti (automatizované zalévání), či minimalizace škod na plodinách. Další možnost IoT aplikace je na metropolitní úrovni, tzv. inteligentní města (smart city). V současnosti existuje velmi málo těchto inteligentních měst, v budoucnu je ale plánováno několik projektů, jak tato inteligentní města implementovat. Opět se nabízí nespočet aplikací, jakožto například monitorování životního prostředí, řízení osvětlení, optimalizace svozu odpadu nebo například systém inteligentního parkování. Zde by řidič vozidla hledající parkovací místo byl například mobilní aplikací, či zabudovanou navigací, naváděn na nejbližší vyhovující místo k zaparkování.



Obr. 1.1 *Ilustrace myšlenky Internetu věcí* [2]

1.2 Chytrá domácnost

Idea ulehčování lidské práce originálními nápady je stará jako lidstvo samo. Když na přelomu 19. a 20. století, v roce 1898, vynalezl Nikola Tesla první dálkový ovladač pro svojí dálkově ovládanou lodičku, otevřel dveře mnoha dalším vynálezcům. Od této doby uplynulo 122 let a celý technický svět se za tuto dobu změnil k nepoznání. Lidé mezitím vynalezli spoustu dalších spotřebičů usnadňující lidskou práci jako například pračky, myčky nádobí, či mikrovlnné trouby. Teslův nápad vzdáleného ovládání se za tu dobu rozšířil do téměř každé domácnosti. Přestože tato zařízení ve velké míře usnadňují život, nejedná se stále o chytrou domácnost. Až na přelomu tisíciletí se začaly objevovat první náznaky chytré domácnosti [3].

S rozvojem síťových technologií se možnosti chytrých domácností posunuly na novou úroveň. Ovládání jakéhokoliv spotřebiče z libovolného místa na planetě díky Internetu není pouze sci-fi. V dnešní době je technologie chytré domácnosti na vzestupu a stále více lidí si do svých domácností pořizuje tato chytrá zařízení. Koncept chytré domácnosti se ale posunul od pouhého vzdáleného ovládání pomocí síťových technologií k více automatizovanému přístupu. Uživatel již nemusí pouze vzdáleně ovládat spotřebiče. Může pro tyto spotřebiče nastavit celou řadu automatizačních schémat, založených na myšlence podmíněného vykonávání akcí. O ovládání takto nastavených spotřebičů se pak v ideálním případě není nutno vůbec starat.

Věřím, že v blízké budoucnosti se chytrá domácnost s rozvojem IoT a umělé inteligence stane skutečně chytrou. V dnešní době se z mého pohledu o skutečné inteligenci nedá hovořit. Inteligencí je zde stále sám člověk. Skutečně inteligentní domácnost založená na umělé inteligenci by pro nás mohla vykonávat spoustu úkonů, které lidem zabírají cenný čas. A mezi ně nepatří jen úkony v domácnosti, ale i ty mimo ni, jako například objednávka nákupu. V tom totiž dle mého názoru spočívá skutečný přínos chytré domácnosti pro mnoho lidí na světě. Hlavní úkol těchto technologií by měl zůstat stále stejný jako u Teslova dálkového ovladače – usnadňovat lidem život.

1.3 Definice problému

V dnešní době je na trhu poměrně velké množství systémů domácí automatizace. Pro běžného uživatele je ale poměrně složité vybrat, který z těchto systémů je pro něj ideální volbou. Běžné, komerčně dostupné systémy domácí automatizace mají většinou určité restrikce pro použití s různými zařízeními. Jinými slovy, ne všechny systémy podporují všechna zařízení. Pro koncového uživatele toto znamená omezení ve výběru integrovatelných zařízení. Tato skutečnost je do jisté míry omezující i pro samotné vývojáře koncových zařízení. Aby jejich produkt podporoval co nejširší spektrum systémů, musí implementovat integraci do jednotlivých systémů zvlášť. Tím se prodlužuje čas vývoje a potenciálně i výsledná prodejní cena vyvíjeného zařízení.

K dispozici je ale několik „open source“ systémů, které se snaží tyto nedostatky odstranit. Jejich problémem je ale jistá nepřívětivost vůči netechnickým uživatelům, pro které by nastavení chytré domácnosti založené na takových systémech mohlo být složitým úkolem.

2 Architektura

2.1 Úvod do architektury systému

Jako řešení problému definovaného v sekci 1.3 se nabízí vývoj nové, jednotné architektury pro systémy domácí automatizace. Tato architektura má za cíl nejen stanovit principy pro vývoj nových zařízení, ale i stanovit metody, kterými je možné do systému zahrnout i koncová zařízení, která nebyla vyvíjena podle principů definovaných touto architekturou. Tím je možné dosáhnout určité univerzálnosti a přívětivosti systému z pohledu koncového uživatele. Průměrný netechnicky orientovaný uživatel tak nebude muset komplikovaně zkoumat, zda je dané zařízení kompatibilní se zbytkem jeho systému domácí automatizace. Další výhodu je možné spatřit i ze strany samotných vývojářů a firem, jelikož se potenciálně usnadní vývoj koncových zařízení.

Nejprve bylo nutné stanovit obecné požadavky na navrhovaný systém. Tyto požadavky se dají shrnout do několika následujících bodů [4]:

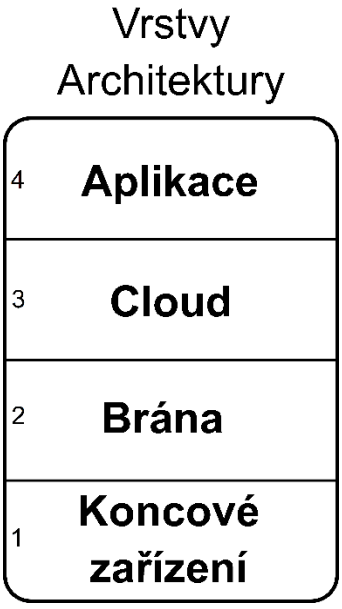
- Bezpečnost
- Cena
- Flexibilita
- Škálovatelnost
- Spolehlivost

Bezpečnost v IoT je dnes obecně velmi důležitým, avšak často opomíjeným tématem. S postupem času, kdy se stává IoT více rozšířenou technologií, budou stoupat i útoky na systémy založené na této myšlence. Příkladem problému v bezpečnosti IoT může být převzetí kontroly nad velkým množstvím koncových zařízení, tedy vytvoření tzv. „IoT botnetu“, který je schopen provádět DDoS útoky v obrovských rozměrech. [5]

Vzhledem k tomu, že v uvažované architektuře se jedná o systém určený primárně pro domácí užití, lze očekávat že bude obsahovat a pracovat s velmi citlivými daty uživatelů. Proto je nutné nasadit odpovídající mechanismy, aby nedocházelo k jejich úniku. Nicméně v tomto případě nejde jen o data, ale i o funkčnost samotného systému. Případný útočník totiž, v případě nedostatečného zabezpečení, může vyřadit z funkčnosti celý systém domácí automatizace, což by vedlo (alespoň částečně, záleží na míře automatizace) k nefunkční domácnosti. Vzhledem k propojení IoT systémů s reálným světem, může nedostatečné zabezpečení určitých typů zařízení, znamenat i ohrožení osob v objektu. Příkladem je chytrý zámek od vstupních dveří. V případě napadení takového zařízení může útočník dveře odemknout, což může vést k újmě na zdraví, či majetku

Flexibilita, škálovatelnost a spolehlivost by se daly zahrnout pod pojem „pohodlí pro uživatele“. Je velmi důležité, aby navržená architektura byla jednoduše škálovatelná. Lze očekávat, že koncový uživatel si bude chtít časem pořídit více chytrých zařízení, která musejí být snadno integrovatelná do již provozovaného systému. Stejně tak spolehlivost systému je jednou z předních priorit. Často je bohužel tato priorita v kompromisu s dalším kritériem, kterým je cena.

Kvůli rozsáhlosti problému řešení takové architektury, bylo vhodné rozdělit si celou architekturu do vrstev. Toto rozdělení je blokově znázorněno na *obr. 2.1*.

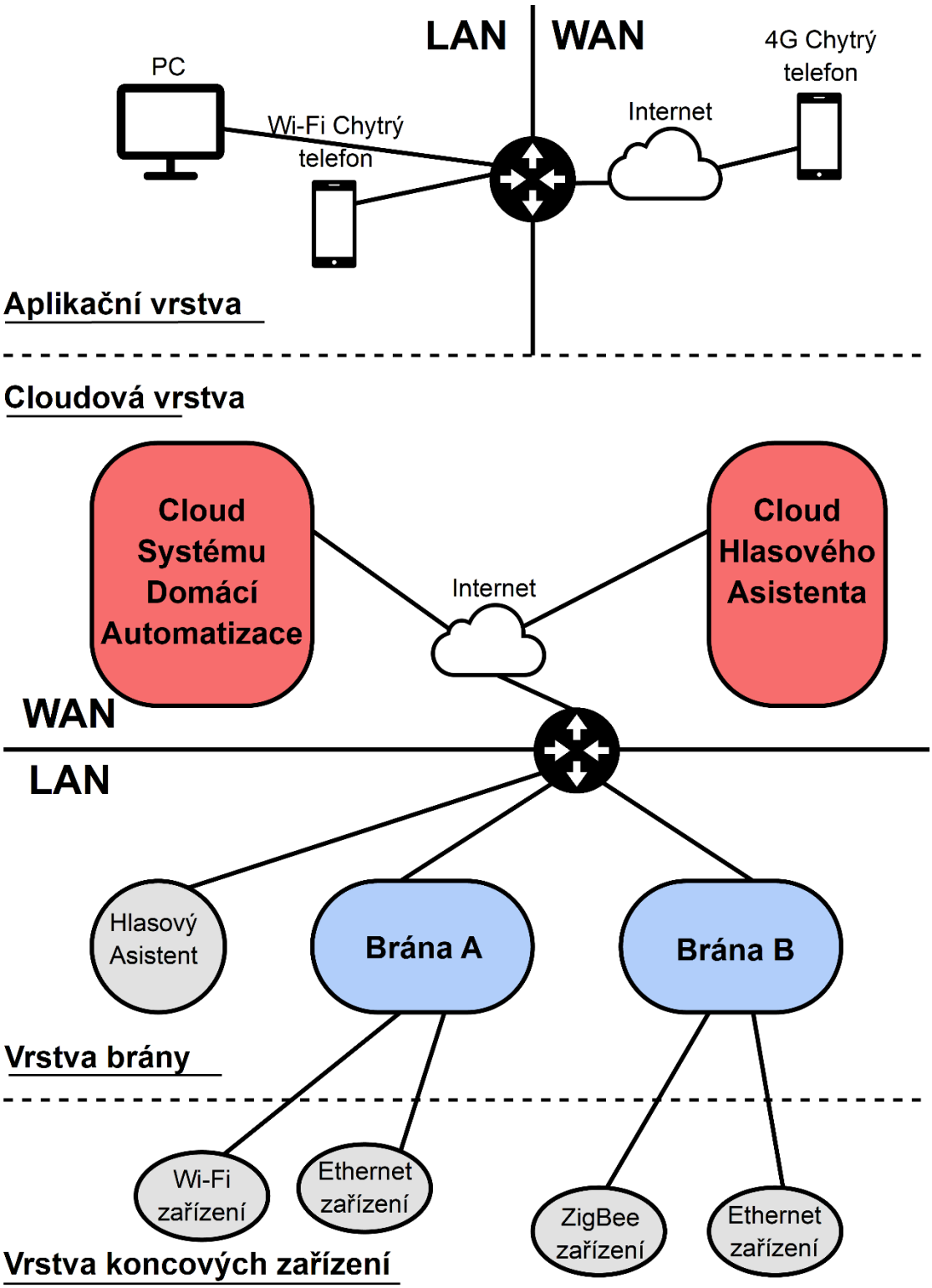


Obr. 2.1 *Rozdělení vrstev architektury systému*

Jednotlivé vrstvy architektury reprezentují konkrétní druhy zařízení, či služeb v rámci celého systému. Každá vrstva má své dílčí úkoly, které musí řešit. V každé vrstvě jsou definovány principy pro návrh příslušných druhů zařízení. Jednotlivé vrstvy byly navrženy tak, aby co nejvíce splňovaly výše definované požadavky na systém. Úkoly dílčích vrstev budou vysvětleny v příslušných kapitolách 3 až 6.

2.2 Dílčí vrstvy architektury

Jak již bylo zmíněno v sekci 2.1, architektura je rozdělena do čtyř vrstev. Na *obr. 2.2* je znázorněná jedna z možných realizací této architektury. V následujících odstavcích budou vysvětleny jednotlivé vrstvy, znázorněny právě na *obr. 2.2.*



Obr. 2.2 *Příklad realizace navrhované architektury*

2.2.1 Vrstva koncových zařízení

Tato vrstva architektury představuje koncová zařízení systému domácí automatizace, které lze rozdělit na senzory a akční členy, či kombinaci obou.

Zařízení obsahující senzory převádí měřenou fyzikální veličinu, na snadno měřitelnou elektrickou veličinu (např. napětí). Ta je případně dále zpracovávána A/D převodníkem. Vzhledem k povaze řešeného systému budeme předpokládat, že zařízení obsahující senzory bude obsahovat i mikrokontrolér, který je vybavený síťovým rozhraním, pomocí kterého bude možné sdílet naměřené hodnoty s vyššími vrstvami architektury. V kontextu domácí automatizace se nejčastěji bude jednat o zařízení jako teploměr, vlhkoměr, senzor pro intenzitu osvětlení, či PIR senzor pohybu.

Akční členy, či aktuátory, jsou naopak zařízení vykonávající určitou činnost ve fyzickém světě. Převádí tak informaci na akci. Bez takových zařízení by systém domácí automatizace neměl velký význam. Cílem těchto zařízení je nahradit lidskou činnost a usnadnit lidem život. I zde je samozřejmě nutné, aby zařízení bylo schopné komunikovat v rámci datové sítě. Příkladem akčních členů mohou být například žárovky, chytré zásuvky s relé, systém pro ovládání žaluzií, ale i robotický vysavač. Na rozdíl od senzorů, které mohou fungovat samostatně bez nutnosti řízení a pouze posílají naměřená data, u akčních členů je nezbytné řídit jejich funkci. Vytvořit síť s „mesh“ topologií, kde by všechna koncová zařízení sdílela navzájem data, by bylo v určitých případech velmi komplikované a nákladné. Proto je vhodné použít centrální řídící jednotku.

2.2.2 Vrstva brány

Abstrakcí centrální řídící jednotky je tato vrstva. Jedná se o jedno, či více zařízení, do kterých je soustředěn síťový provoz. Každý systém domácí automatizace založený na této architektuře musí mít alespoň jednu bránu. Primárním úkolem vrstvy brány je agregovat síťový provoz, sbírat data o systému a na jejich základě řídit ostatní koncová zařízení. Další povinností této vrstvy je umožnit komunikaci s vyšší vrstvou nacházející se mimo LAN, tedy s cloudem.

Výhoda, do určité míry (architektura podporuje vícenásobné brány), centralizovaného řízení je v jednoduchosti systému a také v ceně. Tento přístup, ale oproti distribuovanému systému přináší i několik nevýhod. Jednou z nich je tzv. „single point of failure“ (SPOF), čili problém, kde v případě výpadku jednoho z uzlů sítě, dojde k výpadku celého systému. Dalším potenciálním problémem souvisejícím s centralizací síťového provozu je i otázka bezpečnosti. S těmito nevýhodami je nutné při návrhu této vrstvy počítat a vhodným způsobem je řešit.

2.2.3 Cloudová vrstva

Cloud může poskytovat uživateli služby, které by bylo problematické řešit na úrovni brány, např. samotná registrace a spravování uživatelů, či dlouhodobý záznam dat o systému a jejich zpracování. Úkolem této vrstvy je řešení problémů, které by nižší vrstva z různých důvodů nebyla schopná řešit. Jedná se třeba o problém vzdáleného ovládání koncových zařízení například chytrým telefonem připojeným k internetu pomocí LTE (či jiné mobilní datové sítě). Díky použití NAT na straně domácího routeru není možné se připojit přímo na bránu a ovládat koncová zařízení tímto způsobem. Jeden způsob řešení tohoto problému je na domácím routeru nakonfigurovat „port forwarding“ na bránu, čímž bude toto zařízení přístupné z veřejného internetu. Toto ale vnáší do systému jistá bezpečnostní rizika. Další problém spočívá v tom, že pro netechnicky orientovaného koncového uživatele je konfigurace „port forwardingu“ velmi komplikovaným úkonem. Lepším řešením je tedy využití cloudové vrstvy, která bude v případě vzdáleného ovládání mezičlánkem mezi ovládacím a ovládaným zařízením.

Dalším problémem, který brána pravděpodobně nebude muset být schopná efektivně řešit, jsou určité výpočetně náročné operace. Může se jednat například o zpracování videa, či hlasu. Lze totiž očekávat, že brány budou ve většině případů, z důvodu finanční úspory, stále méně výkonná. Než zvyšovat výkon a tím i cenu brány, je efektivnějším řešením předat tento úkol cloudové vrstvě, která již bude mít dostatečné výpočetní prostředky.

2.2.4 Aplikační vrstva

V dnešní době jsou aplikace pro chytré telefony vnímány jako již běžná věc. Tyto aplikace fungující na zařízeních, které má uživatel běžně k dispozici (PC, mobil), symbolizuje právě aplikační vrstva. Přestože systém domácí automatizace by měl být co nejvíce autonomní a neměl by vyžadovat nadbytečné úkony ze strany uživatele, je stále nutné, aby byl uživatel schopen se systémem komunikovat. Uživatel by měl mít možnost přistupovat k informacím o aktuálním stavu domácnosti, či vzdáleně i lokálně ovládat koncová zařízení z aplikace. Uživatel si také při prvním spuštění musí vytvořit účet v této službě, který bude spojený s konkrétní domácností, či domácnostmi. Dále je nutné řešit integraci nových koncových zařízení do stávajícího systému, kde je nutná akce ze strany uživatele. Tento druh problémů řeší právě tato vrstva architektury.

3 Vrstva koncových zařízení

3.1 Technologie a protokoly

3.1.1 Technologie

Navrhovaná architektura podporuje na úrovni brány více komunikačních technologií, (viz sekce 4.1). Koncová zařízení tedy není striktně nutné omezovat na určitou komunikační technologii. Nicméně v této oblasti je několik nejpoužívanějších bezdrátových komunikačních technologií. Ty jsou pro srovnání uvedeny v *tab. 3.1*.

Technologie	**Wi-Fi**	**ZigBee**	**Z-Wave**	**Bluetooth LE**
Standard	802.11 a/b/g	802.15.4	NA	802.15.1
Kmitočet	2.4 GHz	2.4 GHz (868 MHz EU, 915 MHz US)	800-900 MHz	2.4 GHz
Šířka pásma	20 MHz	2 MHz	NA	2 MHz
Max. přenos. rychlost	54 Mbit/s	250 kbit/s	100 kbit/s	1 Mbit/s
Topologie	Hvězda	Mesh	Mesh	Ad-Hoc
Zabezpečení	WPA2	AES-128	AES-128	AES-128

Tab. 3.1 *Srovnání technologií používaných v domácí automatizaci*

Z důvodu přenosů malých velikostí zpráv, není přenosová rychlost uvedených technologií limitujícím faktorem pro většinu aplikací. Ovšem lze uvést příklady jako přenos videa ve vyšším rozlišení, kde není vhodné použít např. Z-Wave jako komunikační technologii.

Kmitočet a šířka pásma je důležitým parametrem, se kterým je třeba počítat při provozu v zarušeném prostředí daným kmitočtem. Více ohledně tohoto problému je uvedeno v sekci 3.5.

V tabulce je záměrně vynecháno srovnání parametrů jako je například dosah. Měření tohoto parametru je velmi komplikované z důvodu značné závislosti na konkrétní implementaci výrobce. Dále také záleží na scénáři použití – jiný dosah bude uvnitř budovy přes několik zdí, oproti otevřenému prostoru. ZigBee a Z-Wave navíc využívají topologii „mesh“, čímž lze dosah zvětšit. Obecně lze ale tvrdit, že ze zmíněných technologií má Wi-Fi největší dosah. To souvisí i s dalším parametrem, kterým je spotřeba. I zde velmi záleží na typu zařízení i na implementaci výrobce. I v tomto případě lze obecně tvrdit, že Wi-Fi má oproti technologiím, které byly vyvíjeny jako „low-power“, násobně vyšší spotřebu. Z toho důvodu je pro baterií napájená zařízení ideální nasazení některé z „low-power“ technologií jako ZigBee, či Z-Wave.

Ethernet

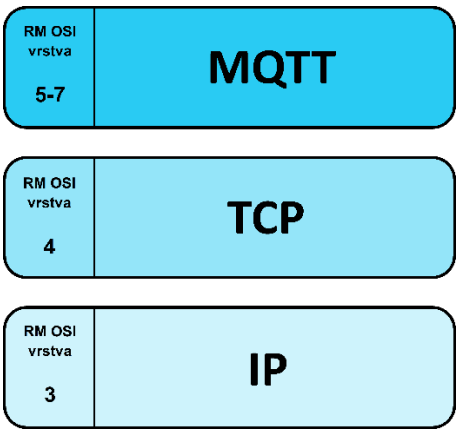
V neposlední řadě je důležité zmínit i technologie využívající metalická vedení, jako např. Ethernet. Hlavním problémem je zde nutnost položení kabeláže. Metalická vedení ale mají oproti bezdrátovým technologiím spoustu výhod. Primární výhodou je velká spolehlivost datového přenosu. U bezdrátových technologií je problémem interference vlnění a v hustě zarušeném pásmu může docházet k datovým ztrátám. U metalických vedení se tento problém nevyskytuje. Další výhodou je potenciálně vyšší bezpečnost oproti bezdrátovým technologiím. Celá síť totiž může být útočníkovi fyzicky nedostupná, což u bezdrátových technologií není z podstaty možné. Kvalita bezdrátové komunikace také může být úmyslně degradována rušičkou signálů. O tomto problému je více informací v sekci 3.2.3.

Díky mnoha výhodám oproti bezdrátové komunikaci je tam, kde je to možné, ideální používat technologii Ethernet. Taková možnost se nabízí při výstavbě nového domu, či při rozsáhlé rekonstrukci bytu.

3.1.2 Protokoly

Protože organizace a posílání zpráv je na úrovni brány řešena pomocí protokolu MQTT, je nezbytné, aby koncová zařízení podporovala tento typ komunikace. Protokol MQTT byl oproti jiným zvolen díky jeho jednoduchosti, což je důležité z důvodu, že většina koncových zařízení jsou málo výkonné mikrokontrolery. Další důvod je velmi malý „overhead“ protokolu, což šetří jak místo v paměti mikrokontrolerů, tak síťový provoz. V případě MQTT je také velmi dobrá podpora ze strany komunity a existuje mnoho „open source“ řešení. [6]

Protokol MQTT je vystaven na protokolové rodině TCP/IP a jedná se v podstatě o poslední 3 vrstvy RM OSI modelu, jak je znázorněno na *obr. 3.1*. Aby zařízení mohlo používat MQTT, musí mít tedy implementovanou podporu protokolů TCP/IP. Z toho důvodu není možné přímo používat MQTT u zařízení na technologiích jako ZigBee, Z-Wave a BLE. Existuje varianta MQTT zvaná MQTT-SN (SN je zkratka pro Sensor Network), která nevyžaduje implementaci TCP/IP. V době psaní této práce ale není mnoho aplikací tohoto protokolu, ani velká nabídka jeho řešení. Protokol MQTT v této architektuře budou tedy implementovat pouze zařízení vybavená protokolovou rodinou TCP/IP. O převod jiných komunikačních protokolů na protokol MQTT se stará vrstva brány.



Obr. 3.1 *Zasazení protokolu MQTT do kontextu RM OSI*

3.2 Bezpečnost

3.2.1 Šifrování

Pro šifrování komunikace je třeba zvolit vhodný šifrovací algoritmus, a to s přihlédnutím na výpočetní možnosti koncových zařízení a požadovanou robustnost zabezpečení. Běžný způsob šifrování komunikace je nejdříve použít asymetrickou šifru pro generaci privátních klíčů, a poté tyto klíče použít pro zašifrování dat symetrickou šifrou. Důvod pro tento způsob šifrování je ten, že asymetrické šifry jsou oproti symetrickým algoritmům náročnější na výpočet a také snáze napadnutelné.

Asymetrické šifry

Na *obr. 3.2* je znázorněné srovnání délky klíčů u RSA (podle autorů, Rivest Shamir Adleman) a ECC (Elliptic-curve cryptography). Je zřejmé, že potřebná délka veřejných klíčů pro stejnou úroveň bezpečnosti je pro ECC daleko menší než u RSA. Díky menší délce veřejného klíče jsou kladeny menší nároky na paměť a výkon zařízení. S délkou klíče potom souvisí i výpočetní doba šifrovacího algoritmu, která je s menším klíčem kratší. ECC je tedy daleko efektivnější než RSA. Z těchto důvodů lze usoudit, že algoritmy na bázi eliptických křivek jsou pro IoT aplikace vhodnější [7].

Dle autorů článku [8] je ideální způsob generace soukromých klíčů pro symetrickou šifru v IoT je použití ECC v kombinaci s Diffie-Hellman protokolem. Diffie-Hellman protokol je způsob, jak vytvořit na nezabezpečeném kanálu symetrické šifrovací klíče. Varianta tohoto protokolu užívající kryptografii na bázi eliptických křivek je pak ECDH (Elliptic-curve Diffie-Hellman), což je touto architekturou preferovaná metoda. Nevýhoda ECDH oproti RSA tkví v tom, že ECDH neřeší autentizaci účastníků. Bez aplikovaného algoritmu pro digitální podpis, je pak tento protokol náchylný na útok typu MITM (Man In The Middle).

	public key length	equivalent symmetric key length	Subjective Security
RSA	1024	80	Not Recommended
	2048	112	Good
	3072	128	Suite B TOP SECRET
	15360	256	Future
ECC	163	80	Not Recommended
	224	112	Good
	256	128	Great
	384	192	Suite B TOP SECRET
	521	256	Future

Obr. 3.2 *Srovnání asymetrických šifrovacích algoritmů RSA a ECC* [7]

Symetrické šifry

Nejpoužívanější symetrickou šifrou je dnes AES (Advanced Encryption Standard). Tuto šifru je vhodné aplikovat i v této architektuře. Hlavním důvodem, proč je AES vhodnou volbou pro šifrování dat jsou zejména malé nároky na paměť zařízení stejně jako na jeho výpočetní výkon. AES-128 je pro šifrování dostatečně bezpečné a není třeba používat delší klíče. V *tab. 3.2* je uvedena délka klíče a doba, kterou by trvalo prolomení této šifry „brute force" metodou. Je uvažován superpočítač s výpočetním výkonem 10.51 PFLOPS při potřebném výkonu 1000 FLOPS pro vyzkoušení jedné možné kombinace [9].

Key size	Time to Crack
56-bit	399 seconds
128-bit	1.02×10^{18} years
192-bit	1.872×10^{37} years
256-bit	3.31×10^{56} years

Tab. 3.2 *Porovnání délky klíče a nutné doby pro „brute force" útok* [9]

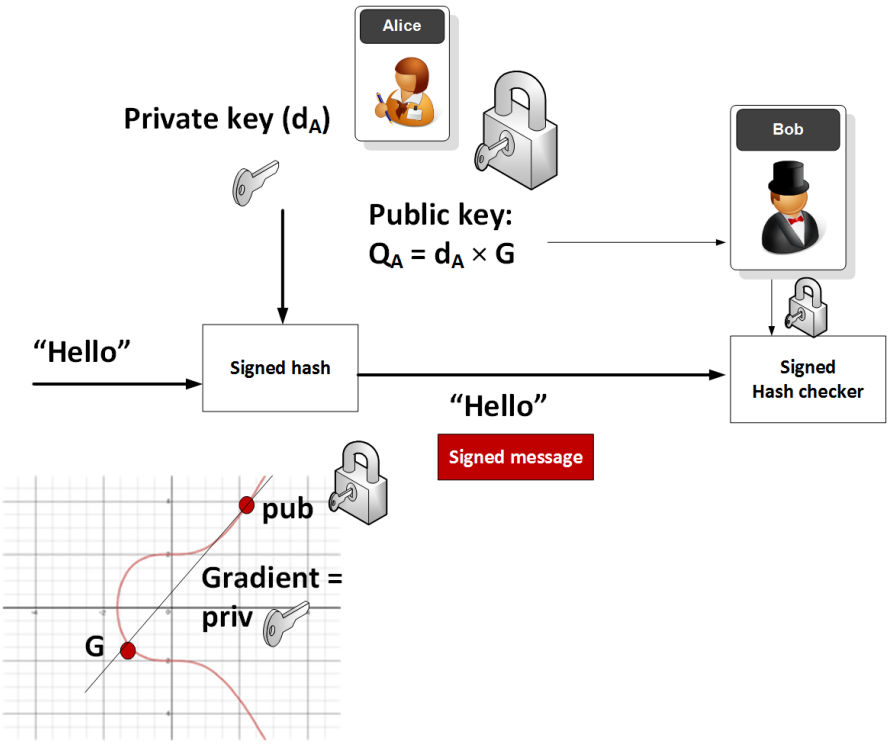
Problém nastává při opakovaném posílání stejné zprávy, která bude mít za důsledek stejnou zprávu v šifrované podobě. Útočníkovi, který odposlouchává zprávy, se pak naskytuje příležitost najít v komunikaci určité vzorce a šifru prolomit. Kvůli tomuto problému je nutné použít tzv. inicializační vektor (IV). Jedná se o náhodné doplnění vstupu, které je známé oběma stranám šifrované komunikace, zajišťující rozdílný výstup šifry. Pro každou zprávu je tedy nutné použít jiný IV.

3.2.2 Autentizace

Autentizace v bezpečnosti znamená, že zařízení vždy ví, že protější strana je opravdu ta, za kterou se vydává. Jedním z algoritmů, které mohou být použité k digitálnímu podpisu je i již zmíněné RSA. Další možností je DSA (Digital Signature Algorithm). I v případě DSA existuje varianta, která pracuje za použití eliptických křivek. Protokol ECDSA (Elliptic Curve Digital Signature Algorithm) má oproti RSA i DSA výhody, že je díky použití algoritmů na bázi eliptických křivek výpočetně méně náročný, při zachování stejné úrovně bezpečnosti. Opět je to díky kratším veřejným i soukromým klíčům. Výsledný digitální podpis má taktéž menší velikost a zmenšuje se tak objem přenesených dat. Je tedy vhodný pro nasazení v IoT zařízeních. Je potřebné provést vzájemnou autentizaci koncových zařízení a brány. Potenciální problém ECDSA i DSA oproti RSA je ten, že při použití nedostatečně kvalitního generátoru náhodných čísel může dojít k prolomení šifry. Na *obr. 3.3* je znázorněn proces digitálního podpisu pomocí ECDSA. [10]

Další částí autentizace koncových zařízení je schválení připojení těchto zařízení do sítě samotným uživatelem. To provede uživatel ze zařízení v aplikační vrstvě, které je přihlášené na jeho účet.

Proces autentizace v systému domácí automatizace, ale obecně i v IoT je velmi důležitý. Bez něj by mohl útočník připojit do sítě libovolné zařízení, které by se mohlo vydávat za kterýkoliv z článků systému. V případě MITM útoků by útočník potenciálně mohl převzít kontrolu nad celým systémem.

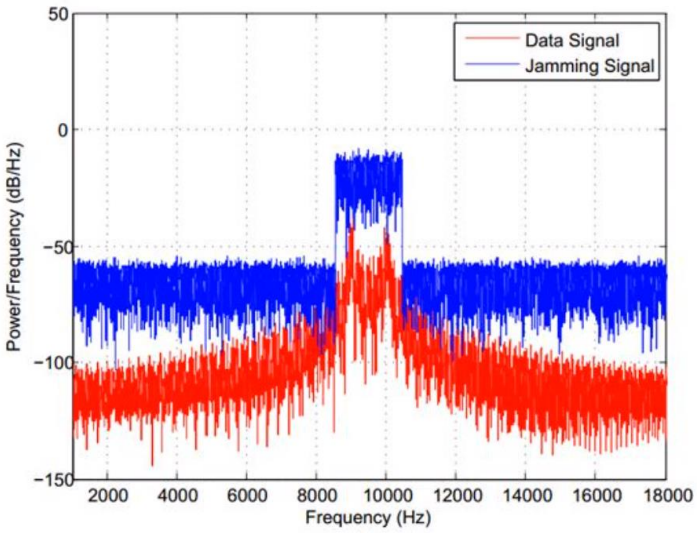


Obr. 3.3 *Proces digitálního podpisu pomocí ECDSA* [11]

3.2.3 Fyzické zabezpečení

Jak již bylo zmíněno v sekci 3.1.1, je nutné systém zabezpečit i proti fyzickým hrozbám, jako je například rušení komunikačního média. V případě použití bezdrátové komunikace v kritických aplikacích, jako je například bezpečnostní systém domácnosti, se uživatel potenciálně vystavuje odstavení funkce tohoto systému rušičkou signálu. Z tohoto důvodu je nutné zajistit, aby komunikace kritických aplikací probíhala, pokud možno, po metalickém vedení. Na *obr. 3.4* je vyobrazeno zachycení takového rušení signálu na spektrálním analyzátoru.

V článku [12] je uveden způsob kontroly hlasových asistentů pomocí namíření laseru na MEMS mikrofon daného zařízení. Útočník tak může skrze okno vysílat tento laserový signál, který si MEMS mikrofon interpretuje jako řeč k ovládání domácnosti. Tento specifický problém je dalším příkladem bezpečnostních trhlin v moderních zařízeních pro chytrou domácnost.



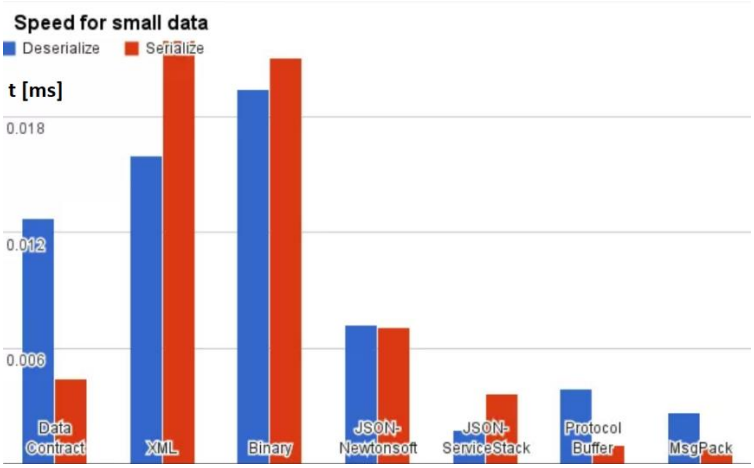
Obr 3.4 *Rušení signálu – červeně datový signál, modře rušící signál* [13]

3.3 Datový formát

Tato sekce se zabývá pouze formátem zpráv určených pro komunikaci mezi zařízeními a bránou. Ve speciálních případech, kdy zařízení potřebuje odeslat např. zvukovou nahrávku, či video se použije odpovídající typ souboru (.mp3, .mp4, atd.).

Nejjednodušší možností je použít tzv. prostý text. V tomto případě by senzor měřící teplotu pouze poslal naměřenou hodnotu jako např. „22.5“. Brána již ví, že zařízení, ze kterého zpráva přišla, měří teplotu. Brána tak dokáže data interpretovat správným způsobem. Tato možnost je z hlediska velikosti dat nejefektivnější.

Problém u prostého textu nastává při přenosu komplikovanějších zpráv obsahujících více dat. Zde je vhodné řešení použít specifický datový formát. Tímto formátem může být XML, či JSON. V případě XML je ale velikost výsledné zprávy větší než v případě daleko úspornějšího JSON. To je v případě mikrokontrolerů nežádoucí z důvodu úspory paměti i času potřebného ke zpracování tohoto formátu. Na *obr. 3.5* je znázorněn čas potřebný pro zpracování různých datových formátů. V tomto grafu je zmíněn i relativně nový formát MessagePack. Jedná se o velmi úsporný binární formát. Na *obr. 3.6* je uveden příklad konfigurační zprávy a její velikost v bytech při použití jak formátu JSON, tak formátu MessagePack. Lze vidět, že za použití formátu MessagePack bylo v tomto konkrétním případě ušetřeno více než 30% datové velikosti zprávy [14].



Obr. 3.5 *Srovnání času nutného ke zpracování různých datových formátů* [14]

JSON 70 bytes

```
{"msg": "config", "ID": "ABC", "service": "actuator", "type": "light"}
```

MessagePack (hex) 47 bytes 67 %

```
84 a3 6d 73 67 a6 63 6f 6e 66 69 67 a2 49 44 a3 41 42 43 a7 73 65 72 76 69
63 65 a8 61 63 74 75 61 74 6f 72 a4 74 79 70 65 a5 6c 69 67 68 74
```

Obr. 3.6 *Srovnání velikosti zprávy v JSON oproti formátu MessagePack*

Z těchto dat vyplývá, že oba formáty JSON i MessagePack jsou vhodné pro aplikaci v této architektuře. MessagePack má potenciál ušetřit jak místo v paměti, tak při velkém množství zařízení i přenášený objem dat v síti. Oproti JSONu ovšem není tak moc často používaným formátem, proto je z tohoto pohledu JSON vhodnější volbou.

3.4 Komunikace s vrstvou brány

3.4.1 Konfigurace

Po připojení zařízení k bráně je nutné, aby připojené koncové zařízení poslalo zprávu, která obsahuje všechny důležité údaje o právě připojeném koncovém zařízení. Příklad této zprávy je uveden v sekci 3.3 na *obr 3.6*. Není nutné posílat veškeré informace, brána má určité defaultní vlastnosti, pro dané třídy zařízení. Nicméně je nutné specifikovat klíčové informace o připojeném zařízení. Mezi tyto konfigurační údaje patří například tyto informace:

- ID
- Služba
- Typ

ID

Značí jméno zařízení. ID nemusí být v síti unikátní, tento problém řeší brána (např. přidáním číslovky za ID v případě více zařízení se stejným ID – Žárovka 1, Žárovka 2, atd.).

Služba / Service

Značí úlohu zařízení v systému domácí automatizace. Tato položka specifikuje celkem 4 možnosti:

- Actuator (akční člen)
- Sensor (senzor)
- Combined (kombinace akčního členu a senzoru)
- Other (zařízení nespadající do této kategorie)

Typ / Type

Značí konkrétní druh zařízení. Jedná se o podmnožinu položky „Služba“. Například v případě „service“: „actuator“ může být „type“: „light (switch, blinds…)“.

Mezi volitelné konfigurační údaje pak patří:

- ON/OFF zpráva
- Jednotky měření
- …

ON/OFF zpráva / message

Platí pouze pro akční členy. Specifikuje zprávu při jejímž přijetí se změní funkční stav zařízení.

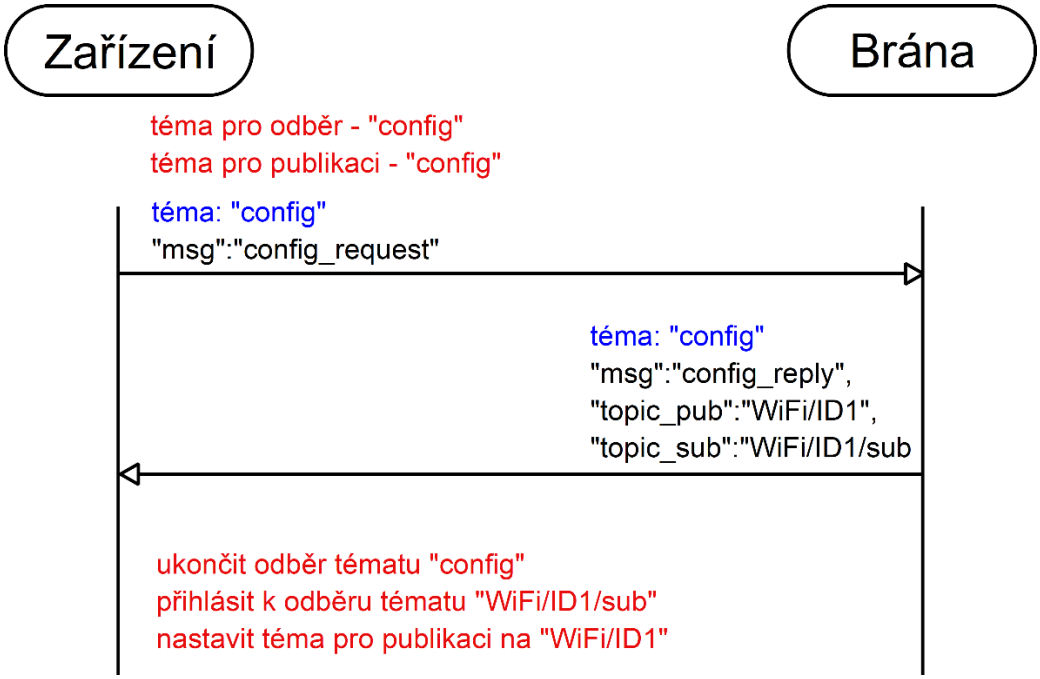
Jednotky měření / Units

Platí pouze pro senzory. Specifikuje jednotku měřené veličiny.

3.4.2 Témata pro publikace a odebírání zpráv

Tato sekce se zabývá problémem MQTT témat u zařízení s implementovanou rodinou protokolů TCP/IP. U zařízení bez implementace TCP/IP, komunikujících na jiných protokolech, se o tento problém stará middleware na úrovni brány.

Proces získání informace, která témata používat, je znázorněn na *obr. 3.7.* Zařízení ihned po připojení k bráně odešle konfigurační zprávu na „config“ téma, ve které je obsaženo i ID zařízení. Brána po obdržení zprávy prohlédne svůj konfigurační soubor, ve kterém jsou uložené informace o připojených zařízení a zjistí, zda nedochází ke konfliktu ID. Pokud ano, modifikuje ID tak, aby nedocházelo ke konfliktu. Pokud již ke konfliktu ID nedochází, odešle na „config“ téma názvy témat, které má dané zařízení používat. Po obdržení této zprávy, se zařízení odhlásí z „config“ tématu a přihlásí se k odběru přiděleného tématu. Dále nastaví téma, na které se posílají zprávy („publish topic“). Obdobný proces probíhá i u zařízení bez implementované protokolové rodiny TCP/IP, akorát pouze na úrovni middleware.



Obr. 3.7 *Schéma zprovoznění komunikace mezi zařízením a bránou*

3.4.3 Struktura zpráv

V případě složitějších zařízení a systémů není možné, aby komunikace probíhala pouze pomocí nestrukturovaných zpráv bez jasných pravidel. Stanovení pravidel pro obsah zpráv zjednoduší a zefektivní vývoj koncových zařízení. Uvedená struktura je navržena za použití objektové notace ve formátu JSON.

Message/msg

Tato položka specifikuje typ zprávy, kterou zařízení posílá. Druh zprávy není vždy nutné specifikovat. Například v případech, kdy zasílaná data pochází ze senzoru měřící pouze teplotu, brána může předpokládat druh dat, které od daného zařízení přijímá. Nicméně v situacích jako je např. zasílání konfigurační zprávy, jejíž obsah byl rozebrán v předchozí sekci, je nutné specifikovat její druh.

Zde je uveden výpis několika možných druhů zpráv.

- config
- data
- control
- other

Config

Význam tohoto druhu zprávy i jejího obsahu byl již vysvětlen.

Data

Specifikuje, že zpráva obsahuje obecná data (hodnoty ze senzorů, data o stavu zařízení, atd.).

Příklad:

```
{
  "msg": "data",
  "temperature": 22.5,
  "humidity": 43,
}
```

Control

Zpráva obsahuje příkaz pro zařízení s instrukcemi, co má koncové zařízení vykonat.

Příklad:

```
{
  "msg": "control",
  "light": "ON",
  "brightness": "50",
  "effect": "flash"
}
```

Other

Používá se pro přenos jiných zpráv s blíže neurčeným obsahem.

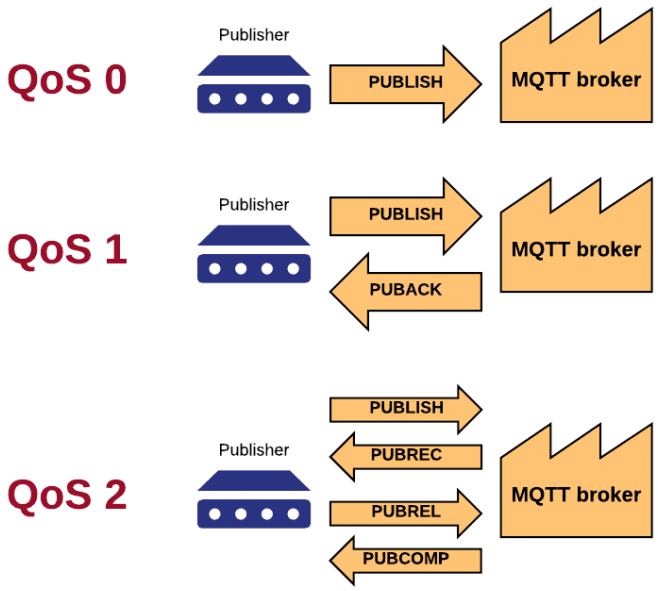
3.4.4 Zajištění funkčnosti komunikace

Protokol MQTT specifikuje 3 možnosti QoS. Ty jsou následující:

- QoS – 0: Nejvýše jednou
- QoS – 1: Alespoň jednou
- QoS – 2: Právě jednou

Obzvláště v bezdrátových komunikacích lze očekávat neobdržení zprávy. Proto je nutné nastavit mechanismy garance obdržení zprávy. V případě použití MQTT by tedy měla být nastavena úroveň QoS na min. 1. Vyšší úroveň QoS je možné použít pouze v nutných případech. V mnoha aplikacích totiž představuje zbytečně mnoho přenesených zpráv.

V případě akčních členů není dostačující jen garance přijetí zprávy, ale i garance provedení úkonu. Po vykonání požadované činnosti musí akční člen poslat zprávu se svým aktuálním stavem jako potvrzení, že se požadovaná činnost skutečně vykonala.



Obr. 3.8 *Znázornění jednotlivých úrovní QoS v MQTT* [15]

3.5 Řešení problémů

3.5.1 Interference

Z *tab. 3.1* v sekci 3.1.1 je zřejmý problém možné destruktivní interference signálů za použití určitých komunikačních technologií. Wi-Fi, ZigBee i BLE používají pro komunikaci stejnou frekvenci 2.4 GHz. Z tohoto důvodu je nutné vybrat ty komunikační kanály, na kterých dochází k minimální interferenci. Problémem výběru těchto kanálů se zabývá vrstva brány. Více viz sekce 4.2.2.

3.5.2 Výpadek brány

Centralizovaným řízením sítě se celý systém vystavuje problému SPOF. V případě výpadku centrální brány tak hrozí pád celého systému. Tento problém je řešitelný duplikací bran. Koncová zařízení ovšem musí s touto možností počítat. Více v sekci 4.1.3.

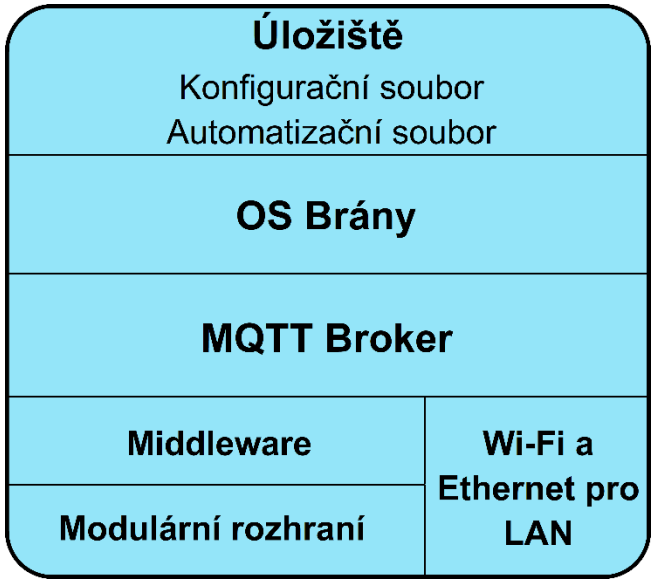
V případě, že se jedná o zařízení s bezdrátovou komunikační technologií, začne zařízení ve chvíli detekce ztráty spojení ihned hledat jinou bránu. K té se následně, pokud je to možné, připojí. Pro zajištění maximální spolehlivosti musí být všechny brány rozmístěné tak, aby koncové zařízení mělo vždy možnost připojení k jiné bráně.

3.5.3 Výpadek elektrické energie

Pro určitá zařízení je nutná možnost manuálního ovládání v případě výpadku elektrického proudu. Mezi zařízení vyžadující tuto možnost patří například chytrý zámek dveří, či elektrické otevírání okna. Jedná se o zařízení běžně fungující bez potřeby elektrické energie (tedy ne například žárovky, zásuvky, atd).

Další možností při nutnosti uchování plné funkčnosti systému je zajištění náhradního generátoru elektrické energie, či systému UPS.

4 Vrstva brány



Obr. 4.1 *Blokové znázornění funkčnosti brány*

4.1 Konektivita s vrstvou koncových zařízení

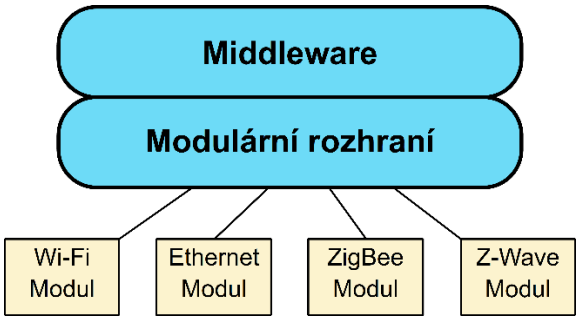
4.1.1 Modulární brána

Díky velké rozmanitosti používaných technologií, a snaze o podporování většiny z nich bylo nutné přijít s efektivním řešením tohoto problému. V článku *„Design and Implementation of Multi-Protocol Gateway for Internet of Things“* [16] byla představena multiprotokolární brána podporující různé druhy protokolů a komunikačních technologií. Navrhované řešení v této architektuře na této myšlence staví. Tímto řešením je modulární přístup k hardwarové i softwarové části konektivity. Představa modulární brány je taková, že koncový uživatel bude moct v případě požadavku podpory určité technologie pouze dokoupit a nainstalovat modul podporující onu technologii, a to jako „plug and play“ modul. Výrobce brány může nabídnout základní rozsah podporovaných technologií, kde by moduly byly potřebné pouze v případě nutnosti rozšíření podpory technologií.

Každý tento modul zastává pozici centrální jednotky sítě. V případě Wi-Fi by tento modul sloužil jako přístupový bod pro koncová zařízení. V technologii ZigBee by pak tento modul zastával pozici koordinátora celé ZigBee sítě. Jiná situace nastává v případě modulu pro technologii Ethernet. V tomto případě by Ethernet modul byl klasický switch.

Na *obr. 4.1* je podpora pro tuto modulární technologii znázorněna v bloku „Modulární rozhraní“ (Modular Interface Socket - MIS). Tato fyzická rozhraní mohou být například konektory RJ-45, či USB.

Navržená brána je modulární i co se týče SW komponent. V případě nutnosti podpory určitých zařízení, či procesů je možné doinstalovat potřebné programy v aplikaci systému.



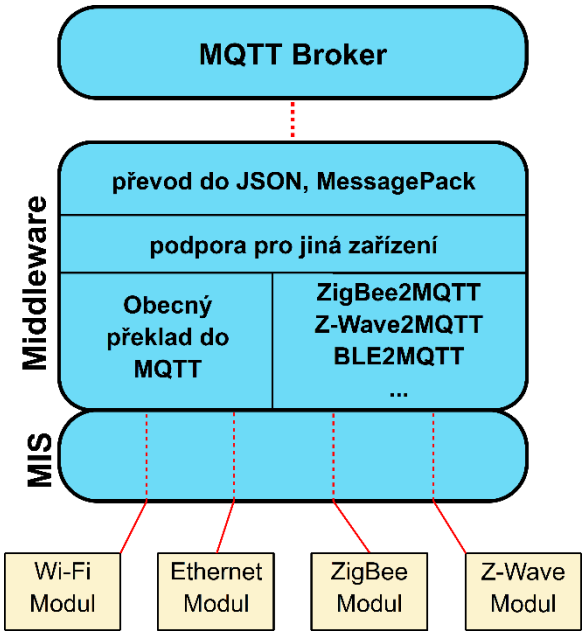
Obr. 4.2 *Znázornění připojených modulů k MIS*

4.1.2 Middleware

Modularita hardwarových prvků ale není sama o sobě dostačující. Je nutné zajistit i softwarovou podporu pro jednotlivé moduly. Touto podporou by byl middleware implementovaný na úrovni brány. U technologií bez rodiny protokolů TCP/IP je nutné zařídit překlad na protokol MQTT. Toto softwarové řešení je již k dispozici v open-source podobě pro nejpoužívanější bezdrátové technologie jako ZigBee, Z-Wave nebo BLE [17] [18] [19].

Aby byla brána skutečně univerzální, je nutné implementovat podporu pro zařízení, která nesplňují požadavky této architektury stanovené v kapitole 3. Pro řešení tohoto problému je nutné vědět, jaké protokoly a jakou strukturu zpráv dané zařízení používá. Implementovaná podpora pro tyto zařízení spočívá v tom, že je nutné provést překlad těchto protokolů a zpráv do formy definované na vrstvě koncových zařízení, se kterou mohou vyšší vrstvy softwaru brány pracovat.

Brána také musí mít schopnost rozeznat zařízení, která tuto podporu nepotřebují. To jsou ta zařízení, jejichž zprávy jsou v protokolu MQTT a mají potřebnou strukturu. Tyto zařízení splňující podmínky stanovené v kapitole 3, mohou fungovat bez nutnosti výrazné modifikace v middleware vrstvě.



Obr. 4.3 *Softwarová struktura middleware vrstvy*

4.1.3 MQTT Broker

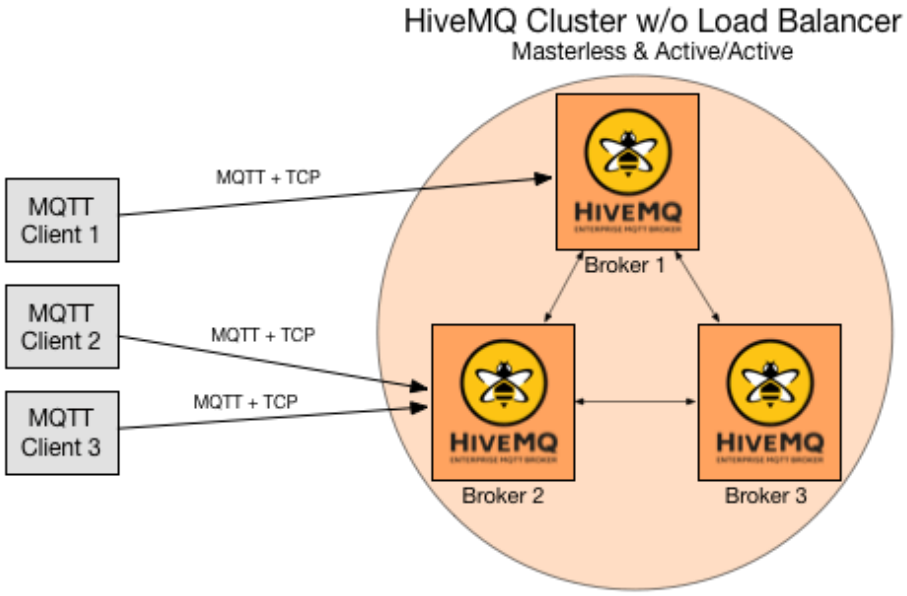
MQTT broker je centrální částí sítě používající MQTT protokol. Stará se o přeposílání zpráv v závislosti na jednotlivých tématech, kterým jsou dané zprávy určeny. Na výběr je několik již hotových open-source řešení. Jejich srovnání je uvedeno v [20]. Obecně je MQTT broker v síti komunikující pomocí MQTT protokolu považován za SPOF.

Z pohledu architektury vrstvy brány je důležité zajistit komunikaci mezi dvěma, či více bránami. Je také nutné zajistit dobrou škálovatelnost systému a spolehlivost systému v případě výpadku jedné z bran. To je zajištěno pomocí implementace tzv. „MQTT Broker Clusteru“, jehož realizace je znázorněna na *obr. 4.4*. „MQTT Broker Cluster“ umožňuje propojení více MQTT brokerů navzájem. Vytvoří se tak distribuovaný systém brokerů, který je reprezentován jako jeden logický celek. Z pohledu koncových zařízení v systému se cluster jeví jako jediný broker [21].

Problém SPOF

Implementací „MQTT Broker Clusteru“ se do určité míry eliminoval SPOF. Jak již bylo uvedeno v sekci 3.5.2, v případě bezdrátových koncových zařízení musí být vždy minimálně dvě brány, ke které se může zařízení v danou chvíli připojit.

Zařízení komunikující po technologii Ethernet jsou připojena do Ethernetového switche (Ethernet modul). Pro řešení SPOF a zachování nejlepší spolehlivosti v tomto případě je nutné, aby do switche byla zapojena víc než jedna brána.



Obr. 4.4 *„MQTT Broker Cluster“ s použitím brokeru HiveMQ* [21]

4.2 Brána v LAN

4.2.1 Připojení brány do LAN

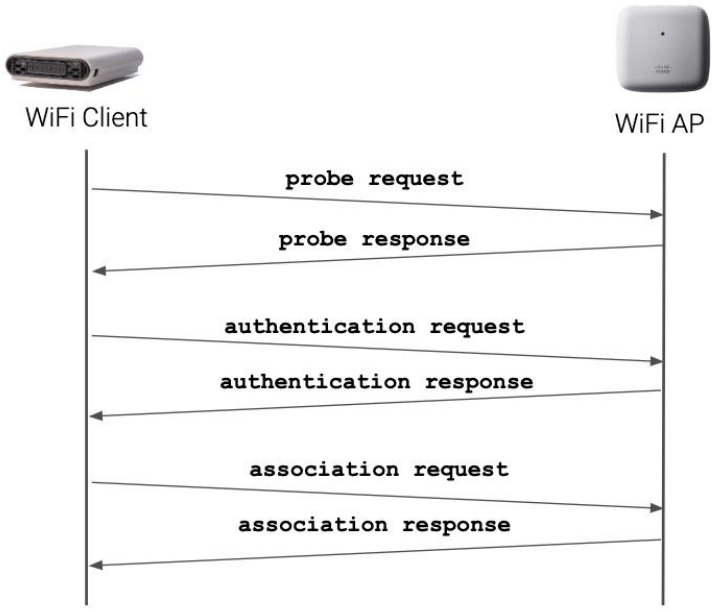
Připojení na lokální síť je potřeba pro zajištění komunikace s internetem, zejména cloudovou vrstvou. V případě použití technologie Ethernet je připojení k bráně triviální a pro zajištění maximální kvality služby je toto preferovaná metoda. Je možné také připojit bránu do domácí sítě pomocí Wi-Fi, kde je proces připojení mírně komplikovanější.

Při připojení brány do WLAN se brána přepne do režimu „Access Point“, ke které se uživatel může připojit svým mobilním zařízením. V aplikaci uživatel zadá potřebné údaje pro připojení k WLAN.

4.2.2 Brána jako přístupový bod

Každá brána bude plnit úlohu přístupových bodů („Access Point - AP“) pro koncová zařízení. Koncová zařízení se tedy nijak nepřipojují do LAN a figurují pouze v síti vytvořené bránou. Jako přístupový bod má brána na starosti i navázání spojení s koncovými zařízeními. Tento proces navázání spojení je pro technologii Wi-Fi zobrazen na *obr. 4.4.*

Pro zabránění neoprávněného připojení do sítě musí být AP nakonfigurovaný s přístupovým heslem, které uživatel sám nastaví. Pro připojení Wi-Fi zařízení se samo koncové zařízení přepne do módu přístupového bodu, ke kterému se uživatel připojí a zadá potřebné údaje pro připojení k bráně.



Obr. 4.5 *Proces navázání spojení u technologie Wi-Fi* [22]

Dalším důležitým úkolem brány v kontextu přístupového bodu je zajistit minimální možnou míru interference. Kritickou frekvencí u bezdrátových komunikací v domácí automatizaci je 2.4 GHz. Jak bylo uvedeno v sekci 3.1.1, na této frekvenci funguje několik komunikačních technologií [23].

Každá brána má za úkol zmapovat míru zarušení na jednotlivých komunikačních kanálech a vybrat pro svoji činnost kanál s nejmenším naměřeným výkonem.

Obr. 4.6 *Rozdělení komunikačních kanálů pro různé technologie na 2.4 GHz* [23]

4.2.3 Externí zařízení

O externích zařízení v kontextu vrstvy brány mluvíme v případě zařízení, která není z různých důvodů vhodné integrovat do systému domácí automatizace na této vrstvě. Typickým příkladem těchto externích zařízení jsou různí hlasoví asistenti jako Google Home, Amazon Echo, Apple HomePod, atd. Ovšem lze najít i mnoho jiných příkladů. Jedná se o zařízení, která ke svojí funkci potřebují komunikaci s cloudovou službou výrobce. Externí zařízení je znázorněno v kapitole 2 na *obr 2.1* právě v podobě hlasového asistenta.

Tyto zařízení se stejně jako brána připojí do LAN a jako takové komunikují s cloudem výrobce. Integrace těchto zařízení je možné realizovat na cloudové úrovni. Tato integrace je dále popsána v kapitole 5.

4.3 Automatizace

4.3.1 Konfigurační soubor

Konfigurační soubor je permanentně uložen v bráně. Tento soubor obsahuje veškeré informace o připojených koncových zařízeních. Ve chvíli, kdy na bránu přijde konfigurační zpráva od koncového zařízení, zkontroluje, zda již toto zařízení nezná.

V případě, že koncové zařízení již v minulosti připojilo a brána má v konfiguračním souboru údaje o tomto zařízení, tuto zprávu ignoruje. Když sice brána má určité informace o tomto koncovém zařízení, ale zasílaná konfigurační zpráva je v nějakém smyslu aktualizovaná, brána aktualizuje záznam o tomto zařízení v konfiguračním souboru.

V případě, že na bránu dorazí konfigurační zpráva, kde záznam o ní není obsažen v konfiguračním souboru, brána tento záznam do souboru uloží. Při vkládání záznamu do souboru musí brána zkontrolovat, zda nedochází ke konfliktu ID koncových zařízení. Pokud ke konfliktu ID dochází, brána ID upraví tak, aby ke konfliktu nedocházelo, a to například přidáním číslovky za ID zařízení. Příklad obsahu konfiguračního souboru je uveden v *tab. 4.1*.

Record #	ID	Service	Type	ON/OFF	Units
0	RGB Light	Actuator	Light	Default	
1	RGB Light1	Actuator	Light	"LightON"	
2	Outlet	Actuator	Switch	"1"	
3	Temp Sensor	Sensor	Temperature		"°C"

Tab. 4.1 *Příklad obsahu konfiguračního souboru*

V případě, že se v systému nachází větší počet bran, je úkolem každé dílčí brány poslat ostatním svůj konfigurační soubor. Každá brána poté svůj konfigurační soubor doplní o chybějící části. Takto bude mít každá dílčí brána sítě kompletní seznam všech připojených zařízení do systému.

Konfigurační soubor se při jakékoliv změně zálohuje na cloud.

4.3.2 Automatizační soubor

Automatizační soubor je také uložen v permanentní paměti brány. Jeho obsahem je seznam automatizovaných úkonů. Tedy jaké konkrétní akce se mají při splnění určitých podmínek vykonat. Tyto podmínky mohou být libovolného charakteru. Může se jednat o časovou podmínku, aktuální stav jiných zařízení, reakce na hodnoty senzorů nebo například stav z různých internetových zdrojů (západ slunce, počasí, atd…).

Automatizační soubor je na bránu zasílán z cloudové vrstvy. Lokálně dostupný automatizační a konfigurační soubor zajišťují poměrně plnohodnotnou funkci systému domácí automatizace i při výpadku připojení k internetu. Lokálně nastavená automatizace je tak velkou výhodou oproti systémům založených na cloudových službách.

Jelikož aktualizace automatizačního souboru přichází z cloudové vrstvy, není nutné, aby se na úrovni brány řešilo předávání automatizačního souboru v případě více bran. Každá brána si tento soubor stáhne z cloudu.

Automation #	Action Device ID	Action	Condition Device ID	Condition
0	RGB Light	set_colour:"red"	security_sensor1	value: "1"
1	Outlet	set_state:"ON"	TIME	20:00
2	RGB Light RGB Light1	set_state:"OFF"	Outlet	state: "ON"

Tab. 4.2 *Příklad obsahu automatizačního souboru*

4.3.3 Operační Systém

Samotné soubory, či MQTT broker nejsou schopny dávat jednotlivým zprávám zasílaným z koncových zařízení smysl. Je nutné v OS brány implementovat část softwaru, která bude řešit právě tento problém.

Samotná brána je jedním z klientů v MQTT síti a je na ní směřována většina zpráv. V případě, že zpráva přišla z jednoho z koncových zařízení se záznamem v konfiguračním souboru, brána zkontroluje obsah této zprávy s obsahem automatizačního souboru. V případě shody s určitým záznamem v automatizačním souboru se pošle zpráva na jedno, či více zařízení (podle seznamu „device ID“ v souboru) s konkrétním obsahem zprávy (opět podle záznamu v souboru).

4.4 Bezpečnost

Vrstva brány používá stejné principy a protokoly pro zabezpečení komunikace, jako ty definované na vrstvě koncových zařízení v sekci 3.2.

4.4.1 Autentizace vůči vrstvě koncových zařízení

Jak bylo definováno v sekci 3.2.2, brána se při komunikaci s koncovým zařízením musí autentizovat. Určitou autentizaci brány zařízení opět provádí sám uživatel při párování dané brány se svým účtem. Poté, co uživatel ve své mobilní aplikaci (ze svého účtu) připojí bránu do sítě, se brána vůči cloudu jeví jako autentizovaná. Takovým branám pak může cloud, jakožto důvěryhodná lokální certifikační autorita pro tento systém, vydat certifikát, kterým se pak brána bude autentizovat vůči připojeným zařízením.

4.4.2 Zabezpečení komunikace s cloudem

Zabezpečení komunikace s cloudem probíhá za použití stejných metod jako zabezpečení komunikace mezi koncovými zařízeními a branou. I zde je při inicializaci komunikaci nutné provést vzájemnou autentizaci, a to z důvodu, že brána v tomto systému není ve vztahu ke cloudu pouhým klientem. Cloud musí mít svůj vlastní certifikát od příslušné certifikační autority.

4.5 Komunikace s cloudovou vrstvou

Stav koncových zařízení

Cloud a brána si často vyměňují informace o stavu koncových zařízení. Může jít například o přímou kontrolu zařízení uživatelem z aplikace ve směru z cloudu do brány, či aktualizace stavu akčního členu nebo senzoru v opačném směru. Tato komunikace probíhá za použití protokolu MQTT. Cloud je v pozici MQTT klienta a komunikuje s MQTT brokerem (či broker clusterem), který je implementován v bráně.

Důvodem pro MQTT je velice snadná a přehledná komunikace ve struktuře Koncové zařízení-Brána-Cloud. Další důvod je ten, že tento typ komunikace bude ve vztahu Brána-Cloud ten nejčastější a jelikož je MQTT velice úsporným protokolem, bude ušetřen jak síťový provoz, tak výkon brány.

Jiná komunikace

Jedná se o komunikaci mezi bránou a cloudem, která se přímo netýká stavu koncových zařízení. Příkladem může být přeposílání konfiguračních a automatizačních souborů, či využívání služeb cloudu (více v sekci 4.5 a kapitole 5). Tato komunikace je realizována protokolem HTTPS nebo MQTT. Tento druh komunikace bude výrazně méně častý, není zde tedy přímo nutné omezovat se na MQTT.

4.6 Další procesy

„Gateway computing“ a „Cloud offloading“

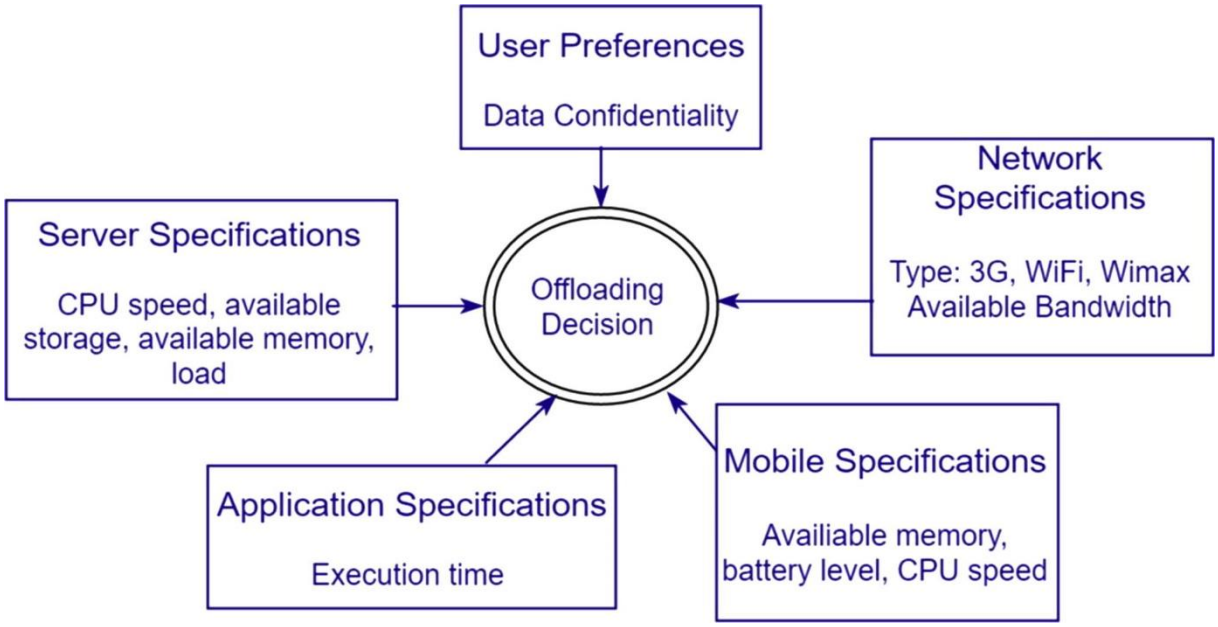
Určitá zařízení mohou využít bránu jako výpočetní sílu pro své výpočetně náročné aplikace. Příkladem náročné aplikace může být například koncové zařízení zpracovávající lidskou řeč. Po doinstalování příslušné softwarové podpory do brány může toto zařízení poslat například audio nahrávku na bránu. Brána poté zpracuje tuto nahrávku a výsledek pošle zpět na koncové zařízení. V tomto případě se jedná o tzv. „gateway computing“.

Existují ale aplikace, kde i výkon brány může být nedostatečný. Těmito aplikacemi může být pokročilé zpracování videa, jako například rozpoznání obličeje na chytrém zámku s kamerou. S řešením tohoto problému přišli autoři článku *„An Extensible Home Automation Architecture Based on Cloud Offloading“* [24]. Zde byla navržena architektura domácí automatizace s využitím tzv. „cloud offloadingu“, tedy předání výpočetně náročné operace výkonnějšímu serveru. Ten opět provede výpočetně náročnou část a bráně vrátí výsledek procesu.

Jelikož se ale jedná o cloudovou službu, je nutné počítat s tím, že může být cloud z nějakého důvodu nedostupný. Je tedy vhodné implementovat různé režimy této služby.

- „cloud offloading“
- lokální zpracování
- omezený režim

V prvním případě je celý zpracovávaný soubor (po případné kompresi) zpracováván na cloudu. Případ lokálního zpracování je ten, kdy je brána ještě schopna v rozumném čase tento výpočetní problém vyřešit, ale již se nejedná o nejrychlejší možný způsob a je používán pouze v případech, kdy není k dispozici cloud. Posledním případem je omezený režim, použitý pouze v krajních případech, kdy by požadovaný výpočet omezil celkovou funkčnost systému domácí automatizace velkým zatížením brány. Omezení je v oblasti funkčnosti služby, kdy daná aplikace nemusí správně fungovat. Je nutné zajistit, aby k tomuto případu nedocházelo, a to vhodným výběrem brány, která má výkonnostní možnosti pro lokální zpracování požadované aplikace.

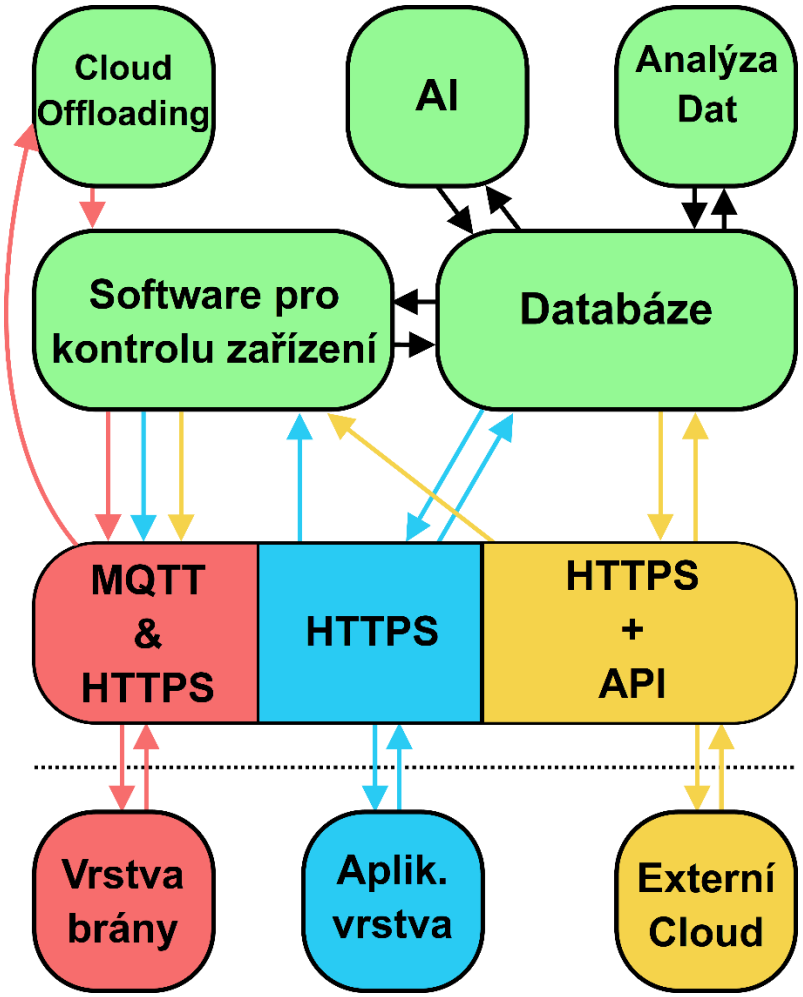


Obr. 4.7 *Parametry algoritmu rozhodující o použití „cloud offloadingu“* [25]

Žádné logování dat

Je důležité zmínit, že na úrovni brány neprobíhá žádné logování dat. Logování dat, jako naměřené hodnoty ze senzorů nebo stavy zařízení, probíhá až na cloudové vrstvě. Toto opatření je zavedeno z důvodu, že je možné očekávat, že brány budou mít jako úložnou paměť pouze nějakou formu flash paměti, kde v nejmenších případech bude stačit kapacita řádově nižší desítky GB (16-32 GB). V případě častého logování velkého množství dat může potenciální dojít k opotřebení flash paměti a k výraznému snížení životnosti tohoto zařízení a následně k jeho poruše.

5 Cloudová vrstva



Obr. 5.1 *Blokové schéma architektury cloudu*

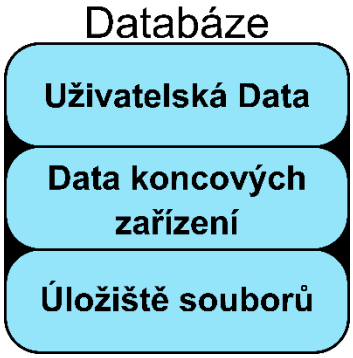
5.1 Software pro kontrolu zařízení

Software pro kontrolu zařízení (Device Control Software - DCS) je blok architektury cloudu určen ke kontrole koncových zařízení, v případech jiných, než je lokální kontrola na úrovni brány. Lze uvést tři případy, kdy je tuto službu nutné využít. Na *obr. 5.1* lze vidět, na které podněty DCS reaguje.

- Vzdálené ovládání z aplikace
- Kontrola pomocí externích zařízení
- Kontrola při využití „cloud offloading“

V případě přijetí relevantních dat (např. požadavky na změnu stavu) z kterékoliv z těchto tří oblastí, pošle DCS kontrolní zprávu na bránu pomocí protokolu MQTT.

5.2 Databáze



Obr. 5.2 *Struktura databáze*

Uživatelská data

Databáze v cloudu je využitá pro ukládání uživatelských dat. Jsou zde uloženy např. „hash“ záznamy přihlašovacích údajů, osobní údaje o uživatelích, či UID účtu. Více v sekci 5.7.

Data koncových zařízení

V této části se ukládají data z koncových zařízení. Ukládají se funkční stavy koncových zařízení v čase, či hodnoty ze senzorů. Tato data jsou poté použita pro další analýzu, viz sekce 5.4 a 5.5.

Úložiště souborů

Zde jsou uloženy soubory nutné pro běh automatizačního systému. V kontextu celkové navrhované architektury se pak jedná zejména o automatizační soubor a zálohu konfiguračního souboru (sekce 4.3).

5.3 Cloud offloading

„Cloud offloading“ je služba poskytovaná pro vrstvu brány cloudovou vrstvou. Pokud koncové zařízení a brána vyhodnotí proces jako příliš výpočetně náročný (tzn. je výhodnější provést „cloud offloading“), přepošlou se potřebné soubory a instrukce na výpočetní část cloudu. Tímto způsobem se dramaticky zvýší rozsah úkonů, které zvládnou koncová zařízení [24].

Velký problém tohoto přístupu je latence. V případě že se jedná o časově citlivou, „real-time“ aplikaci, minimální latence je často klíčovým parametrem. Bohužel i při nasazení nejlepších síťových technologií je zde klíčovým parametrem fyzická vzdálenost mezi koncovým zařízením a cloudem. Pokud bude latence příliš vysoká nemusí již „offloading“ být výhodným řešením.

Řešením tohoto problému může být fyzické oddělení části cloudu, která bude využitá přímo pro „cloud offloading“. Tento proces se tak bude vykonávat na jiném serveru, který je fyzicky blíž, a tak není

zatížen vysokou latencí. Tuto myšlenku aplikoval prof. Satyanarayanan a kol. v [26]. Zde je definován pojem „cloudlet“, tedy jakýsi mezi-stupeň mezi koncovým zařízením a cloudem, využívaný především pro „real-time cloud computing“. „Cloudlet“ je znázorněn na *obr.5.3.* Více informací, ohledně problematiky „cloud offloadingu“, je možné dohledat na [25] [27].

Obr. 5.3 Koncept „cloudletu“ [26]

5.4 Umělá inteligence

V navrhované architektuře je možné využít konceptu umělé inteligence. AI v této architektuře má za úkol rozpoznávání vzorů v používání domácnosti uživatelem. AI čerpá data z databáze, kde jsou uloženy stavy zařízení a data ze senzorů. Z těchto dat je potenciálně možné rozlišit určité vzory, jako například příchod uživatele do domácnosti v pravidelnou hodinu a vykonání určité rutiny. Tato myšlenka byla publikována v práci s názvem *„Artificial intelligence and Internet of Things in a „smart home“ context: A Distributed System Architecture“* [28].

Cílem AI v této navržené architektuře je přivést uživateli vyšší komfort při používaní chytré domácnosti. AI je totiž z rozlišených vzorů schopná vytvořit, či dynamicky modifikovat automatizační soubor. Celý proces automatizace, by tedy byl pro koncového uživatele snadnější. Klasický způsob vytváření automatizačního souboru je takový, že uživatel musí ze zařízení v aplikační vrstvě manuálně nastavit jednotlivé automatizace sám. S velkým počtem koncových zařízení je ale takové vytváření velmi zdlouhavé a problematické. Pokud uživatel nevynaloží potřebné úsilí pro tvorbu těchto automatizací a nastaví pouze pár základních, není plně využit potenciál chytré domácnosti. AI má tedy za úkol tento zdlouhavý proces, alespoň částečně, eliminovat.

Dalším možným využitím je i částečné přímé řízení koncových zařízení. AI by byla schopna detekovat určité podněty, na které je nutné reagovat určitým způsobem. Tyto podněty nespadají do rutinních, a není tak možné využít rozpoznávání vzorů a dynamickou modifikaci automatizačního souboru. Tyto podněty ale mohou být velmi různorodé a vývoj takové AI by byl značně náročný. V této oblasti je nutný další výzkum.

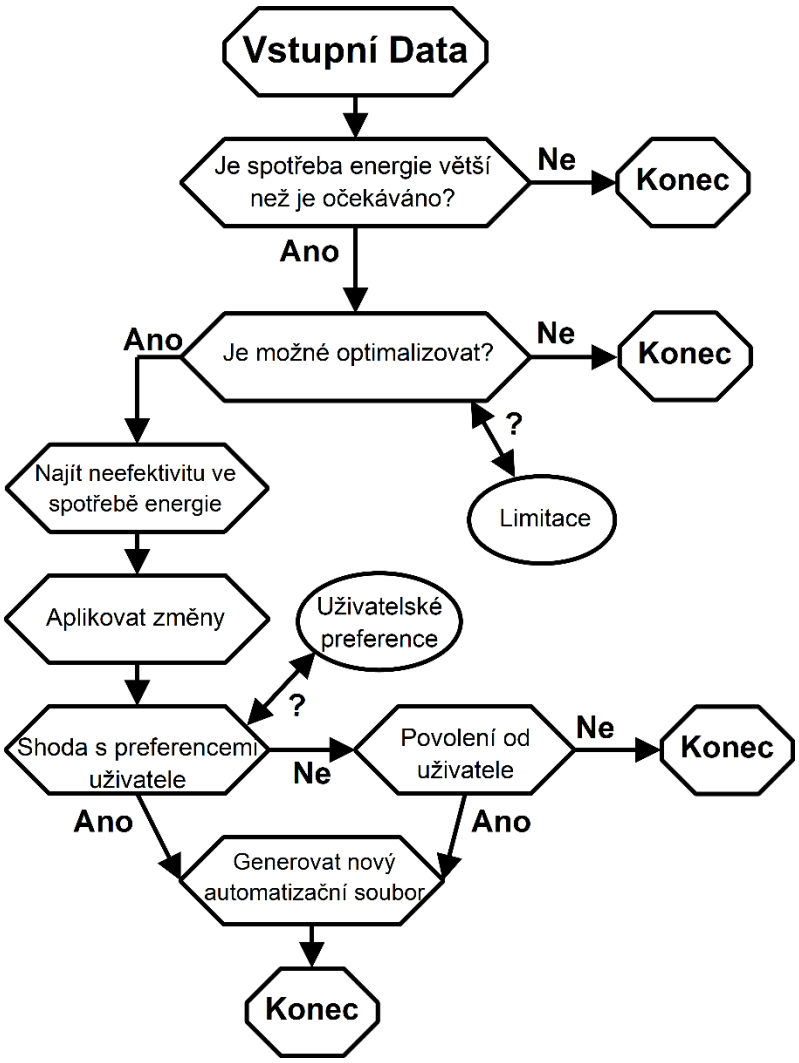
5.5 Analýza dat

Tato část cloudu je druhou částí využívající dat koncových zařízení z databáze. Úkolem je analyzovat a interpretovat zjištěná data a určitou formou zpětně poskytnout zpětnou vazbu uživateli.

Prvním druhem zpětné vazby mohou být například grafy, či tabulky znázorňující průběhy měřených veličin nebo zobrazující statistická data. Uživatel má poté tedy možnost zpětně posoudit vývoj měřené veličiny, na základě kterých, může například změnit některé své činnosti.

Druhým výstupem analýzy dat jsou určitá doporučení pro uživatele. Systém má implementované algoritmy, které jsou schopné zjistit, zda uživatel například nějakým způsobem neplýtvá energií. Tato doporučení si uživatel může zobrazit v aplikaci a vzít je v potaz a následně si vybrat, zda tento problém bude řešit, či nikoliv.

Příklad takového algoritmu pro analýzu spotřeby energie a následnou úpravu a generaci automatizačního souboru na základě této analýzy je uveden na *obr. 5.4.*



Obr. 5.4 *Vývojový diagram příkladu algoritmu pro optimalizaci spotřeby elektrické energie*

Algoritmus pravidelně kontroluje, zda je celková spotřeba energie vyšší než referenční hodnota, která může být buď nastavena uživatelem, či brána systémem jako průměr za určité období. Pokud je možné optimalizovat, tzn. pokud není v činnosti ideální případ, algoritmus prohledává databázi pro neefektivity. Adeptem na neefektivní využívání energie může být například svícení při dostatečné úrovni světla nebo bez přítomnosti osob, topení s otevřeným oknem, či přetápění místností, atd. Po aplikaci těchto změn v automatizačním souboru se zkontroluje, zda je některá ze změn v konfliktu s preferencí uživatele na běh domácnosti (např. RGB osvětlení pro estetické účely, které by jinak systémem bylo vyhodnoceno jako neefektivní). Pokud ke konfliktu nedochází vygeneruje se nový automatizační soubor, který si brána stáhne. V případě konfliktu s uživatelskou preferencí je nejprve uživatel dotázán, zda se změnami souhlasí.

5.6 Externí cloud a server

Externí cloud je cloud, se kterým nemá brána přímou komunikaci. V kontextu architektury se bude nejčastěji jednat o dvě možnosti.

První možností jsou cloudy externích zařízení, která byla definována v sekci 4.2.3. Způsob komunikace těchto externích zařízení je ten, že mají pouze přímé spojení s cloudem své vlastní služby. Nejsou tak integrována do vrstvy brány jako ostatní koncová zařízení v architektuře. Možnost, jak tyto zařízení integrovat do systému se tedy nabízí až na úrovni cloudu. Externí cloud musí mít k dispozici API, které bude možné pro komunikaci mezi cloudy využít. Samozřejmostí je implementace podpory pro toto API externího cloudu i na cloudu tohoto systému. Příkladem tohoto případu může být Amazon Alexa. Hlasoví asistenti jako Echo Dot komunikují s vlastními servery Amazonu, kde se provádí veškeré zpracování hlasu uživatele. V této architektuře by tedy uživatel nainstaloval určitý modul (v případě Alexa – „skills“) umožňující využití API. Tak by se umožnila komunikace mezi cloudem Amazonu a cloudem této architektury.

Druhou možností v případě komunikace s externími servery na této úrovni je komunikace se servery poskytující různé služby. Může se jednat například o servery poskytující předpovědi počasí, či dopravní situace. I v tomto případě je nutné, aby daná služba měla k dispozici API pro komunikaci.

5.7 Registrace uživatelů

Cloud má ve spolupráci s aplikační vrstvou na starosti registraci a přihlašování uživatelů. Pro vytvoření účtu je nutné poskytnout e-mail a zvolit dostatečně silné heslo. Na e-mail následně přijde kód pro verifikaci účtu. Pro každý uživatelský účet je generováno unikátní UID.

Pro ukládání přihlašovacích údajů musí být dodržené přísné bezpečnostní standardy, zejména pro ukládání hesel. Řešením může být použití kódu HMAC v kombinaci s použitím bezpečné hashovací funkce. Definice HMAC je:

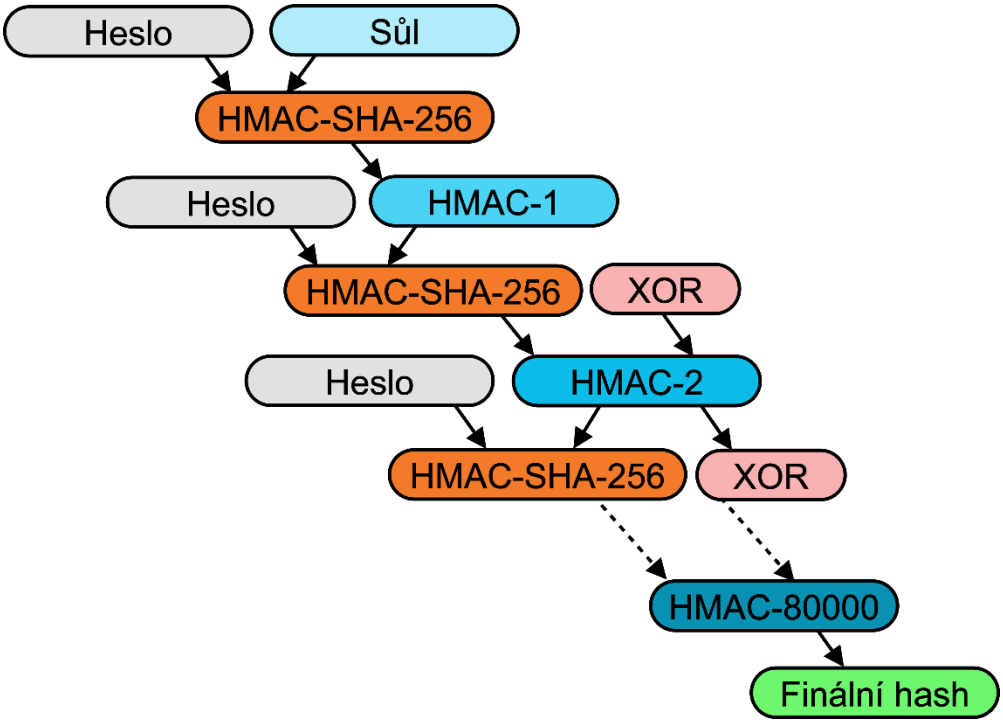
$$HMAC_K(m) = h\Big((K\ XOR\ opad)||h\big((K\ XOR\ ipad)||m\big)\Big) \tag{5.1}$$

kde:

- *h* je zvolená hashovací funkce
- *K* je tajný klíč, salt
- *m* je zpráva, pro kterou se HMAC počítá
- || je zřetězení
- *XOR* je logická funkce exkluzivního součtu
- *opad, ipad* jsou konstanty.

Zvolenou hashovací funkcí může být například populární SHA-256, či SHA-512. V této kombinaci se hovoří o HMAC-SHA-256, resp. HMAC-SHA-512 [29].

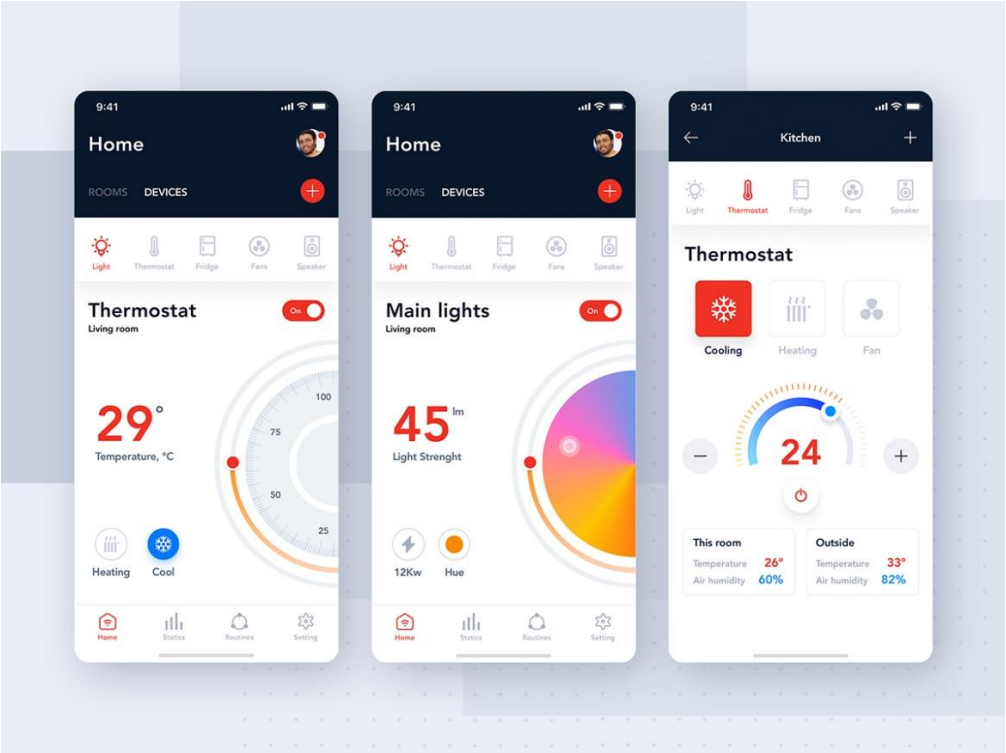
S výpočetním výkonem dnešních počítačů není výpočet hashovací funkce příliš komplikovaný. V případě úniku databáze přihlašovacích údajů se útočníkovi nabízí „brute-force“ útok. Způsob, kterým je možné útočníky zpomalit je „hashovat“ heslo vícekrát. PBKDF2 je kryptografickou funkcí, která provádí právě tento proces. Na *obr. 5.5* je znázorněn průběh funkce PBKDF2 [29].



Obr. 5.5 *Použití funkce PBKDF2 s 80 000 iteracemi k vytvoření „hashe“ hesla*

6 Aplikační vrstva

6.1 Uživatelské prostředí



Obr. 6.1 *Grafické uživatelské prostředí aplikace pro chytrou domácnost* [30]

Uživatelské prostředí (UI) aplikační vrstvy musí obsahovat informace o domácnosti a možnosti na její ovládání. UI ve velmi přehledné formě poskytuje uživateli naměřená data ze senzorů. Je nutné také zajistit ovládací prvky pro jednotlivé akční členy, obsažené v chytré domácnosti a zobrazit informaci o aktuálním stavu jednotlivých akčních členů.

UI je primárním způsobem, jak je uživateli umožněno ovládat jeho chytrou domácnost a zjišťovat informace o jeho stavu. Proto musí být uživatelské prostředí navrženo pro plynulou funkčnost a orientaci v něm. Uživatel musí být schopný intuitivně a bez obtíží přidávat a ovládat akční členy. Způsobem, jakým lze dosáhnout intuitivního ovládání je rozdělit domácnost do jednotlivých segmentů, a to buď podle místností nebo podle jednotlivých zařízení. Příklad vhodného UI je k dispozici na *obr. 6.1*.

6.2 Komunikace s Cloudovou vrstvou

Jak již bylo deklarováno v kapitole 5 na *obr. 5.1,* pro komunikaci aplikační vrstvy s cloudovou byl zvolen protokol HTTPS. Standardní metodou pro komunikaci mobilních aplikací je právě využití protokolu HTTPS v kombinaci s architekturou REST a využitím datového formátu JSON.

Primárním důvodem pro použití této kombinace je snadná implementace, jedná se o nejpoužívanější protokoly a způsoby programování v této oblasti. Programátoři jsou tedy s těmito postupy velmi dobře seznámeni a vývoj takové aplikace bude snadnější. Kromě snadné implementace poskytuje tento přístup další výhody. Jednou z nich je i relativně malý „overhead“ protokolu HTTPS, který v kombinaci s úspornou architekturou REST dokáže komunikovat velmi efektivně a rychle. Výhody formátu JSON, popsané v sekci 3.3, jsou aplikovatelné i zde.

6.3 Lokální a vzdálený režim

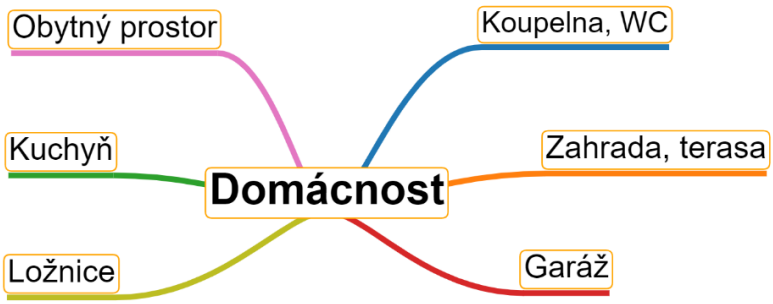
V případě, že je uživatel s mobilním telefonem, či jiným zařízením v aplikační vrstvě připojen v lokální síti, ve které se nachází i brána, je vhodné, aby se eliminoval mezistupeň v podobě komunikace s cloudem. Aplikace pozná, zda se nachází v domácí Wi-Fi síti, ve které je přítomna i brána a podle toho přepne režim na lokální. Implementací této funkce se zlepší responzivita celého systému. Také se tím řeší řada problémů z kategorie „SPOF“. Příkladem může být situace, kdy cloudová vrstva nebude z nějakého důvodu dosažitelná. Další problémová situace může být ta, kde uživatel má mobilní telefon připojen na lokální Wi-Fi síť, ale ze strany ISP dojde k výpadku sítě. Aplikace tak již nebude moc komunikovat s cloudovou vrstvou, přičemž řešením by právě byla implementace lokálního režimu.

Pokud uživatel využívá mobilní internet, nebo je připojen do jiné sítě nežli té, ve které se nachází brána, aplikace se přepne do vzdáleného režimu a celá komunikace a řízení systému probíhá skrze cloud.

7 Identifikace možných aplikací

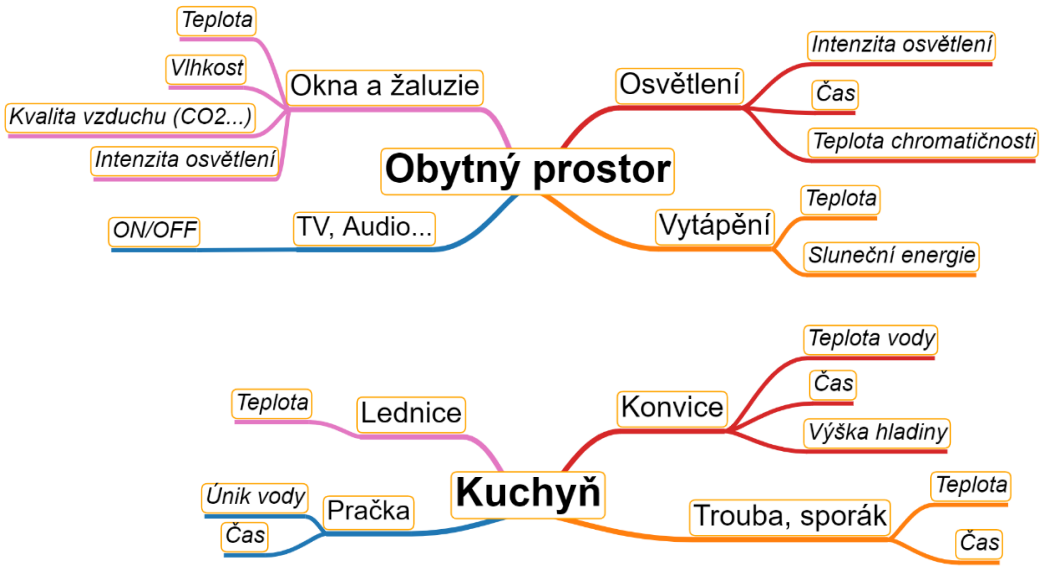
Navrhnout zajímavé a originální aplikace využitelné běžným uživatelem je poměrně složitým úkolem, a to zejména v dnešní době, kdy je na trhu k dispozici velké množství produktů v různých oborech, včetně domácí automatizace. Pro systematičtější přístup k tomuto úkolu byla prvním krokem tvorba myšlenkové mapy. Byla tvořena s cílem rozložit komplexní problém návrhu aplikace pro chytrou domácnost, na jednodušší podproblémy. Pro přehlednost byla celá myšlenková mapa rozdělena na více jednotlivých map.

Tvorba této mapy probíhala analýzou objektu „domácnost“. Domácnost byla rozdělena na jednotlivé místnosti, či přilehlé pozemky a její jiné části. Tyto sub-objekty byly poté podrobeny další analýze.



Obr. 7.1 *Fundamentální rozdělení domácnosti na její jednotlivé části*

U jednotlivých částí byla identifikována aktuální a budoucí technická zařízení náležící do dané části domácnosti. Ve snaze o úsporu místa a zvýšení přehlednosti mapy jsou u jednotlivých sub-objektů vypsána pouze ta zařízení, která jsou typicky použitá v dané části domácnosti. Jejich uvedení v každé části by bylo velmi zdlouhavé. Tím se ale nevylučuje přítomnost těchto zařízení v jiných částech domácnosti. Příkladem by mohlo být okno, vyskytující se téměř v každé části domácnosti. Dále byly identifikovány měřitelné veličiny a stavy přímo související s daným zařízením. Tyto stavy a veličiny byly vybírány tak, aby na základě jejich hodnot mohla být jednotlivá zařízení automaticky ovládána. Výsledek této analýzy je k dispozici na myšlenkové mapě na *obr. 7.2*.



Obr. 7.2 *Myšlenková mapa k jednotlivým částem domácnosti*

Díky tomuto rozdělení bylo možné organizovaně hledat souvislosti mezi jednotlivými objekty. Příklady možných aplikací, které jsem navrhnul jsou uvedeny v následujících sekcích. Navrhované aplikace byly vytvářeny s myšlenkou uživatelské přívětivosti, jakožto hlavního kritéria. Přívětivost spočívala v minimální interakci uživatele se systémem, kromě prvotního nastavení. Zásadní byl tedy velký podíl automatizace v aplikaci.

7.1 Systém pasivní regulace osvětlení a teploty

Primární myšlenka tohoto systému vznikla z větve „Obytný prostor“ z myšlenkové mapy na obr. 7.2. Byla nalezena souvislost mezi automatizovaným ovládáním žaluzií a oken a regulací teploty a osvětlení. Dovolím si nejprve nastínit jednu ze situací, kterou tento systém může inteligentně řešit pomocí technologií domácí automatizace. V horkých letních dnech se velice často stává, že se brzy stane pobývání v domácnosti, či jiných uzavřených objektech velice nepříjemné díky vysokým teplotám. Reakce lidí je často taková, že začnou otevírat okna a větrat. Tato reakce ale často není ideální. Venkovní teploty touto dobou již pravděpodobně převyšují ty vnitřní a větrání za účelem snížení teploty tak postrádá smysl. Systém pasivní regulace osvětlení a teploty by mohl být řešením obdobných problémů.

Princip systému by byl takový, že na základě uživatelova nastavení ideální teploty a intenzity osvětlení by se pouze pomocí pasivních možností, jako otevíráním okna a vhodným nastavením žaluzií, snažil systém udržovat toto nastavené prostředí. Je vysoce pravděpodobné, že například v horkých letních dnech by pouze tento systém nebyl dostačujícím řešením pro dostatečný komfort. Systém ale má velký potenciál dosáhnout nemalých finančních úspor za spotřebu elektrické energie, a to zejména za klimatizaci, v určité míře i za osvětlení. Při zatažených žaluziích s odrazivým povrchem je možné zablokovat až zhruba 45 % prostupující solární energie oknem [31].

Regulace okenní ventilace je realizovatelná i pro běžná okna, není tedy nutno investovat do drahých, plně elektrických oken. Podobné zařízení vyvíjí a nabízí například česká firma Vektiva [32].

Obr. 7.3 *Přístroj na řízené otevírání oken [32]*

Zde je uveden výčet senzorů a akčních členů potřebných pro realizace této aplikace.

Senzory

- Pyranometr, či referenční fotovoltaický článek, pro měření solárního záření [33]
- Senzor teploty a vlhkosti
- Venkovní senzor teploty, alternativně data o venkovním počasí z internetu
- Gyroskop umístěný na žaluziích
- Fotorezistor, senzor intenzity osvětlení

Akční členy

- Regulace otevření okna
- Ovládání žaluzií

Systémy automatického ovládání žaluzií jsou již běžně k dostání na trhu. Podstatné je, aby bylo možné žaluzie regulovat náklonem jednotlivých listů. Z gyroskopu umístěném na jednom z listů žaluzií

je pak možné získat zpětnou vazbu o náklonu. Nastavení náklonu je podstatné pro optimalizaci osvětlení a propuštěného tepla. Z pyranometru získá systém informaci o intenzitě solárního záření, na základě kterého je možné odpovídajícím způsobem regulovat náklon žaluzií. Pro tuto aplikaci je možné použít i méně přesný referenční fotovoltaický článek, jak je uvedeno v článku [33]. Žaluzie nemusí být vždy plně zavřené, lze je nastavit pod takovým úhlem, aby docházelo k odrazu určité části světla na strop. Nebylo by v tom případě nutné používat umělé osvětlení, které by snížilo finanční úspory dosažené tímto systémem. Mohly by ale nastat i situace, kdy je výhodnější plně zatáhnout žaluzie a při příliš velkém poklesu intenzity osvětlení používat úsporné LED žárovky, než více klimatizovat prostor a nechat prostupovat světlo.

Tato aplikace jde naproti trendu ve stavebnictví. Od roku 2020 je povinné dle evropské legislativy stavět nové budovy s nízkou energetickou náročností. U této aplikace je potenciál, že by mohla pomoci snížit energetickou náročnost budovy.

Myšlenky pasivní úspory energie pomocí žaluzií se ujala i firma Smarterhome. Ta vyvinula chytré žaluzie umožňující automatizaci na základě měřených parametrů prostředí [34].

7.2 Systém pro zkvalitnění spánku a zdraví

Kvalitní spánek je nesmírně důležitou součástí našeho života. Nedostatečný, či nekvalitní spánek má prokázaný negativní vliv na zdraví. Mezi tyto negativní vlivy, kterým se jedinec takovým spánkem vystavuje, patří například zvýšené riziko diabetu, kardiovaskulárních chorob, imunodeficience, či obezity. Kromě somatických potíží je nedostatečný spánek příčinou i mnoha psychických poruch jako jsou deprese, úzkostná porucha nebo bipolární porucha [35] [36].

Jako řešení těchto problémů se uvádí zejména vytvoření striktního spánkového režimu a dodržování tzv. „spánkové hygieny“. Ta zahrnuje několik člověkem poměrně snadno ovlivnitelných doporučení jako omezení alkoholu a kofeinových nápojů ve večerních hodinách, omezit konzumaci těžkých jídel na noc nebo například pravidelná fyzická aktivita.

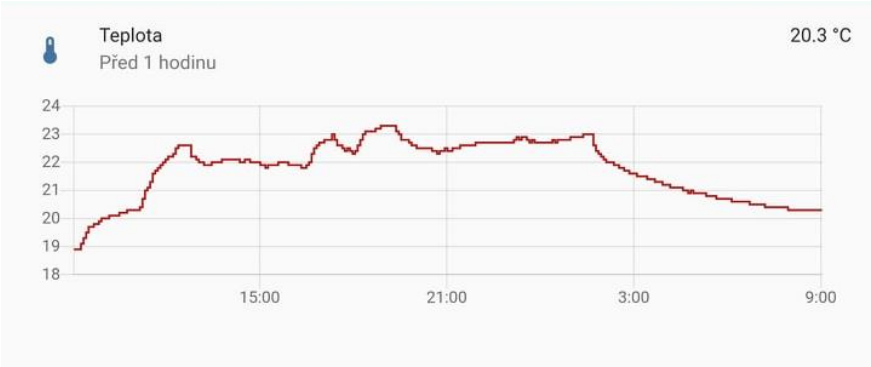
Na kvalitu spánku má ale velký vliv i samotné prostředí, ve kterém spíme. Udržovat ideální parametry prostředí během spánku je poměrně velký problém, což z této činnosti činí ideální úkol pro systém domácí automatizace. Mezi tyto důležité parametry teplota prostředí, vlhkost, či světlo. V této konkrétní aplikaci je tedy cílem dosáhnout ideálního prostředí pro spánek pomocí sensorů a akčních členů.

Teplota

Jako ideální rozsah teploty místnosti pro spánek se uvádějí teploty okolo 16-18 °C. Příliš vysoká teplota má negativní vliv na kvalitu spánku, jak je uvedeno v [37]. V této studii je ale popsáno i riziko plynoucí z příliš nízké teploty při spaní. Studie ukazuje, že při nízkých teplotách se v určitých částech spánku zvyšuje aktivita parasympatiku, mající vliv na srdeční aktivitu. Ve studii [38] je pak popsána možná korelace mezi velmi nízkou teplotou při spaní a zvýšenou mortalitou na ischemickou chorobu srdeční při těchto podmínkách.

Částečným řešením tohoto problému by byl adekvátní systém klimatizace a vytápění, který by byl schopen automaticky regulovat teplotu. Mnoho domácností tímto bohužel vybaveno není, a tak je nutné přijít s alternativním řešením.

V této aplikaci se počítá s použitím obdobného systému okenní ventilace jako v sekci 7.1. Zařízení regulující okenní ventilaci by bylo řízeno zpětnou vazbou z teplotních senzorů rozmístěných v pokoji. Systém by byl schopný zjistit (buď dalším senzorem, či údaji o teplotě z internetu) venkovní teplotu, aby rozpoznal, zda je v aktuální situaci výhodné pro dosažení ideální teploty okno otevřít, či nikoliv. Pro demonstraci schopnosti regulovat teplotu v domácnosti pomocí okenní ventilace jsem provedl 24 hodin dlouhé měření teploty pomocí senzoru DHT-22. Na obr. 7.4 jsou všechny poklesy teploty způsobené otevřením okna v místnosti. Toto měření ukazuje schopnost regulovat teplotu v místnosti. Dále by musel systém zhodnotit, zda otevřením okna nedojde ke změně dalších parametrů mimo přípustnou mez, více dále.



Obr. 7.4 *Měření teploty*

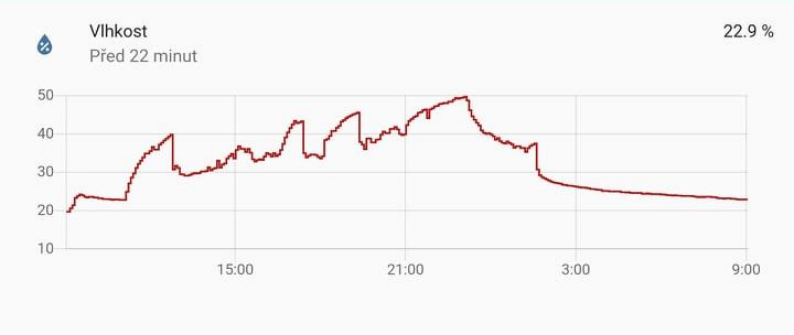
Vlhkost

Vhodná hodnota relativní vlhkosti se pohybuje v rozmezí 40-50 %.

Snížená vlhkost má opět vliv na zdraví jedince. Může se projevit například suchou pokožkou, škrábáním v krku, či méně hydratovanou sliznicí nosohltanu, čímž se snižuje její obranyschopná funkce a jedinec se stává náchylnějším k různým infekcím. I to je jedním z důvodů, proč jsou v zimních obdobích, kdy je vlhkost nejnižší, nejvíce rozšířená různá respirační onemocnění [39].

Příliš vysoká vlhkost souvisí s předchozím parametrem – teplotou. Kromě zvýšeného rizika plísní v místnosti, klesá termoregulační schopnost těla a člověk může pociťovat diskomfort v tomto ohledu [40].

Příliš nízkou vlhkost je pak možné regulovat omezením větrání, ideálně pak integrací zvlhčovače do automatizovaného systému. Vysokou vlhkost je naopak možné snížit větší mírou větrání. Pro demonstraci tohoto jevu jsem opět provedl 24 hodin dlouhé měření vlhkosti pomocí senzoru DHT-22. Měření na obr. 7.5 probíhalo se zapnutým zvlhčovačem v pokoji po celou dobu. Stejně jako u teploty, i zde jsou patrné poklesy vlhkosti způsobené otevřením okna.



Obr. 7.5 *Měření vlhkosti*

Osvětlení

Neméně důležitým faktorem je pak kvalitní osvětlení domácnosti. Zejména pak teplota chromatičnosti použitého světelného zdroje. Ve večerních hodinách jsou problémem příliš studené barvy s teplotou chromatičnosti přes zhruba 6000 K. Ve spektru studenějších světel je výrazná modrá složka, která má neblahé účinky na cirkadiánní rytmus. Vystavení této modré složce světla má za důsledek opožděné vyplavování melatoninu [41]. Naopak přes den je prospěšnější používat právě studenější zdroje světla, blízké tomu dennímu.

Tato aplikace počítá s implementací světel s variabilní teplotou chromatičnosti. Teplota světla by se řídila pomocí denní doby a času, který uživatel nastavil jako preferovaný čas pro spánek. V ranních hodinách by naopak bylo možné implementovat systém probouzení pomocí umělého východu slunce. Ten spočívá v tom, že místo klasického budíku, který může být pro tělo určitým šokem se používá postupné zesilování intenzity světla v místnosti, což má potenciální vliv na „energičtější“ probuzení [42].

Další parametry

Vyjmenované parametry jsou jen jedny z nejdůležitějších s přímým vlivem na kvalitu spánku a s tím souvisejícího zdraví. Pro kvalitnější implementaci aplikace, je vhodné monitorovat i další parametry na kterých by záviselo automatizované ovládání akčních členů. Mezi patří například koncentrace CO_2, hladina hluku nebo déšť.

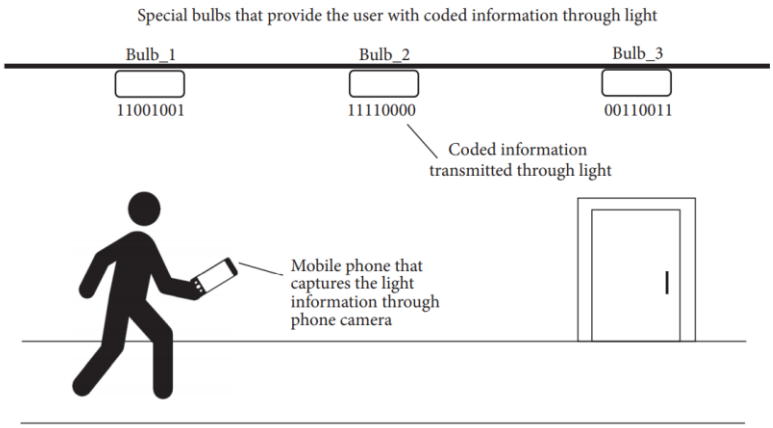
7.3 Systém lokalizace osob v domácnosti

Tento systém vychází z veličiny, která není zmíněná v myšlenkové mapě, protože se neváže pouze k jednomu konkrétnímu spotřebiči, či zařízení. Je jí počet lidí v jednotlivých částech domácnosti. Je velice časté, že různé spotřebiče zbytečně běží a spotřebovávají energii, pokud například v místnosti nikdo není. Obdobně by některé automatizace mohli být spuštěné až tehdy, přesáhne-li počet lidí v místnosti nějakou konstantu.

Použití tohoto systému v již zařízené „chytré domácnosti“ se systémem automatizace má potenciál poskytnout uživateli určitý komfort. Ovládacím prvkem celé domácnosti je de facto sám uživatel a jeho pozice v domácnosti. Odpadá tak nutnost ovládat chytrá zařízení například z aplikace v mobilním telefonu, či hlasovým asistentem. Díky ovládání a možnosti nastavit automatizace na základě pozice uživatele je možné dosáhnout finančních úspor. Jednoduchým příkladem může být zhasnutí světel poté, co počet osob v místnosti dosáhne nulového počtu. Je ovšem možné nastavit komplikovanější automatizace, záleží pouze na kreativitě uživatele.

Přístupů k řešení tohoto problému se již v minulosti objevilo několik. V článku *„Evolution of Indoor Positioning Technologies: A Survey“* [43] je předloženo několik takových přístupů. Dále bude pro srovnání uvedeno několik příkladů.

V uvedeném článku je zmíněno několik technologií lokalizace osob, která vyžaduje přítomnost zařízení vysílající/přijímající různé signály přímo u monitorovaných osob. Tímto zařízením může být i běžný mobilní telefon. Na obr. 7.6 je uvedeno řešení využívající zakódovanou informaci vysílanou viditelným světlem pomocí LED žárovek. Tento systém má ale velmi mnoho nevýhod. Mobilní telefon uživatele musí mít přímou viditelnost na konkrétní žárovku, což je pro koncového uživatele velice omezující.



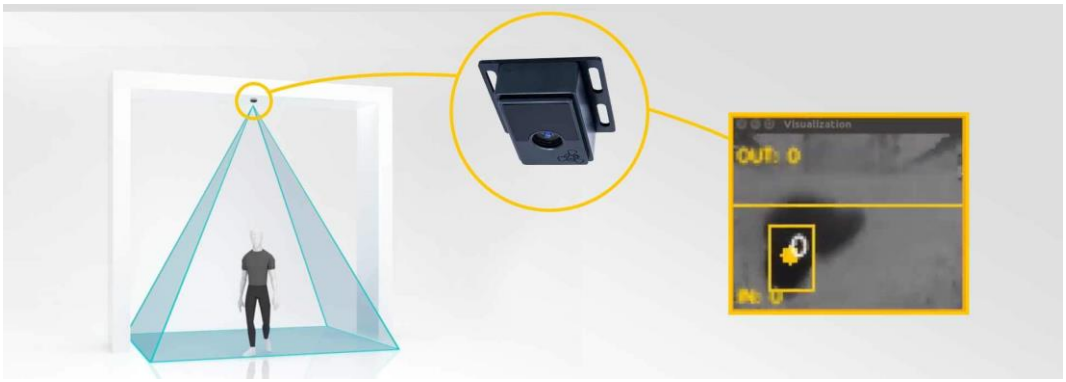
Obr. 7.6 *Lokalizace pomocí zakódovaného signálu ve viditelném světlu [43]*

Další metodou, kde je použitý mobilní telefon pro lokalizaci osob je zjištění polohy pomocí bezdrátových komunikačních technologií jako jsou Wi-Fi, či Bluetooth. Zpracování těchto dat je většinou velice podobné. Jedna z nabízených možností je využít údajů o síle signálu z několika AP k výpočtu polohy uživatele, resp. jeho mobilního telefonu. I zde je opět problém, že uživatel musí u sebe neustále nosit mobilní telefon, či jiné zařízení. I tento přístup je tedy do určité míry, zejména v domácím prostředí, omezující.

Jako zajímavější se jeví technologie zpracovávající zvukové vstupy. Je možné využít mikrofonů z čím dál tím rozšířenějších hlasových asistentů, kteří mohou být umístěni ve všech částech domácnosti,

k získání takových dat. Principiálně pak jde o podobný přístup jako v případě bezdrátových technologií. Zdrojem zvukového signálu je zde ale sám uživatel. Bohužel i zde lze identifikovat několik problémů tohoto přístupu. Například v případech, kdy uživatel nevydává žádný zvuk, či je uživatelem vydávaný zvukový signál slabší nežli hluk pozadí, ztrácí systém možnost uživatele spolehlivě lokalizovat.

Dalším přístupem k systému lokalizace osob je systém znázorněn na obr. 7.7. Jedná se o systém využívající 3D kameru založené na principu „Time-of-Flight (ToF)“. Tato kamera, či obecně ToF senzor, umístěný na vrchní straně zárubně nebo na stropu, je schopen zaregistrovat jednotlivé průchody osob z jedné místnosti do druhé a provádět lokalizaci osob na úrovni zón nebo přímo místností. Přestože se jedná pravděpodobně o nejlepší z dosud zmíněných systémů lokalizace, i zde je poměrně velký problém. Kvalitní ToF senzory jsou poměrně drahé. Cena této 3D kamery na obr 7.5 je (v době psaní této práce) 250 € bez DPH. Levnější varianta v podobě ToF senzoru se prodává za cenu okolo 100 € bez DPH [44]. Po uvážení faktu, že by pro plnou funkci systému bylo třeba mít tato zařízení na každé hranici dvou místností, lze shledat, že tato částka je v prostředí mnoha domácností příliš vysoká. Toto zařízení ale jistě najde uplatnění v profesionální sféře, jako například ve firmách, obchodech, továrnách, atd.



Obr. 7.7 *Použití ToF kamery v systému lokalizace lidí [44]*

Vzhledem k tomu, že výše jmenované příklady systémů lokalizace osob v domácnostech se ukázaly jako z různých důvodů nevyhovující, rozhodl jsem se věnovat se vývoji této aplikace. Tento vývoj je popsán v dalších kapitolách.

8 Idea funkce navrženého systému lokalizace osob

V sekci 7.3 byly prezentovány různé technologie použité k lokalizaci osob, kde společným jmenovatelem byly různé limitující faktory. Mým hlavním cílem bylo navrhnout takový systém, kde jsou ve velké míře tyto limitující faktory omezeny. Požadavky na systém jsem si stanovil tyto:

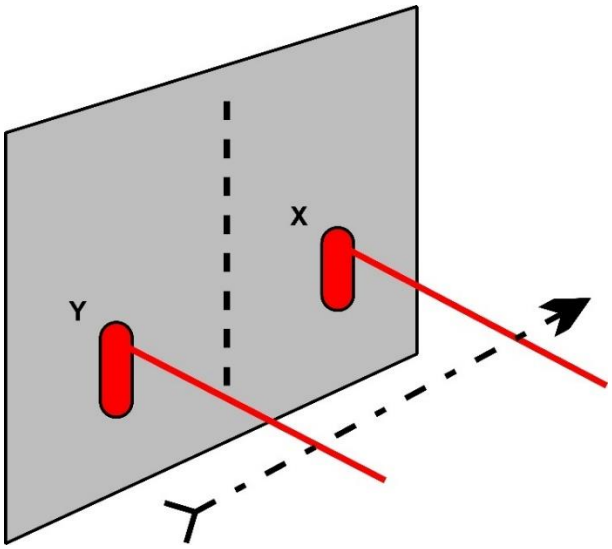
- Uživatel u sebe nemusí nosit jakékoliv zařízení
- Detekce probíhá spolehlivě při všech možných světelných, či hlukových podmínkách
- Cenově dostupné
- Uživatel kvůli použití navrhovaného systému nemusí provádět zásadní změny v domácnosti
- Uživatelsky přívětivé nastavení.

8.1 Navržený princip lokalizace osob v domácnosti

Navrhovaný systém počítá s rozdělením celé domácnosti do několika zón. Zóna je definována jako jedna nebo více místností. Rozdělení domácnosti na jednotlivé zóny závisí plně na uživateli. Jádrem tohoto systému je zařízení umožňující detekovat průchod osoby mezi těmito zónami. Na obr. 8.1 je půdorys domu s dispozicí 2+kk a užitnou plochou 49 m^2. Do tohoto půdorysu byly zakresleny názvy jednotlivých zón a načrtnuto umístění jednotlivých detektorů průchodu na hranicích místností. Z obrázku lze vidět, že pro pokrytí celé domácnosti stačí pouze tři detektory (při zanedbání technické místnosti s plochou 2 m^2). Záleží ovšem na dispozici domácnosti. Na obr 8.1 je patrné, že zóna „Chodba“ je společná pro celkem tři zařízení. Každý detektor je připojen k centrální bráně celého systému domácí automatizace, kam se posílají data o počtech uživatelů nacházejících se v jednotlivých zónách, více v sekci 8.2.

Obr. 8.1 *Půdorys domu se se zakreslenými detektory průchodu [45]*

Princip detektoru průchodu je vyobrazen na obr. 8.2. Spočívá v rozdělení roviny daným zařízením na dvě poloroviny „X“ a „Y“ pomocí senzorů. Ilustrovaná situace zobrazuje objekt pohybující se z poloroviny „Y“ do poloroviny „X“. Zařízení dokáže rozpoznat, ze které strany se objekt pohybuje podle informace, který ze dvou senzorů zaznamená jako první pohybující se objekt. Na ilustrovaném příkladu z obr. 8.2 zařízení nejprve zaznamená objekt na straně „Y“ a až poté na straně „X“. Z toho je detektor schopen rozeznat, že objekt se pohybuje do poloroviny „X“. V tomto případě detektor přičte tento objekt na polorovinu „X“ a naopak z poloroviny „Y“ jeden objekt odečte.



Obr. 8.2 *Navrhovaný princip zařízení pro lokalizaci osob v domácnosti*

8.2 Navržený princip síťové funkcionality zařízení

S ohledem k navržené architektuře v předchozích kapitolách, se jedná o zařízení na nejnižší vrstvě koncových zařízení. Z důvodu, že některé z definovaných vlastností jsou podmíněné funkcionalitou na vyšších vrstvách, není v této práci možné implementovat všechny vlastnosti zmíněné v kapitole 3. Dále je popsán navržený princip síťové funkcionality.

Vzhledem k důvodům uvedeným v kapitole 3 sekci 3.1, jsem zvolil protokol MQTT jako použitý komunikační protokol v této aplikaci. Použitý mikrokontroler musí disponovat implementací protokolové rodiny TCP/IP. Pro pohodlí uživatele a vyšší mobilitu zařízení jsem se rozhodl použít bezdrátovou komunikační technologii Wi-Fi.

Nejprve je nutné implementovat připojení detektoru průchodu do sítě, ve které se nachází i centrální brána systému domácí automatizace. Aby nebylo nutné provádět připojení do sítě WLAN při každém spuštění, je potřeba uložit SSID a heslo k dané síti do flash paměti detektoru. Díky tomuto uložení bude moct zařízení při dalších spuštěních načíst tyto údaje z paměti a provést připojení autonomně. V případě, že v paměti tyto údaje uložené nejsou, bude se muset zařízení přepnout do AP režimu. Myšlenka je taková, že se v tomto režimu k zařízení uživatel připojí pomocí jiného zařízení s Wi-Fi. V této fázi bude probíhat prvotní nastavení celého detektoru průchodu. Zde uživatel zadá jak přihlašovací údaje k síti, tak i názvy zón, které náleží příslušným polorovinám „X“ a „Y“. Je nutné, aby byl uživatel při pojmenovávání těchto zón konzistentní. Tyto údaje si zařízení uloží do flash paměti pro příští spuštění.

Jelikož je MQTT „publish/subscribe“ protokol, je nutné znát jméno tématu na které bude navrhovaný detektor posílat údaje o počtu osob v monitorovaných zónách. Navržený systém počítá s tím, že pro každou ze dvou zón, které zařízení monitoruje, bude vytvořeno vlastní téma. Jméno takového tématu je možné vytvořit pomocí jmen zón, které uživatel zadal během prvotního nastavení. Aby byla možná synchronizace počtu osob v zónách mezi detektory co tyto zóny sdíli, je nutné, aby se zařízení přihlásilo k odběru zpráv na těchto tématech.

Dále je třeba implementovat identifikaci IP adresy brány v síti. Jakožto nejefektivnější způsob jsem zvolil použití protokolu mDNS. Je ale nutné zvolit vhodnou bránu, která podporuje protokol mDNS.

9 Použitý hardware a software

9.1 Senzory

Na trhu je velké množství senzorů na bázi různých technologií. Pro použití ve své práci jsem musel identifikovat ty z nich, které jsou vyhovující pro uvedený princip detekce průchodů, a to i v souladu s požadavky definovanými v kapitole 8.

V první řadě jsem vyloučil jakékoliv systémy senzorů pracující v režimu vysílač-přijímač, kde jsou vysílač a přijímač oddělené součástky. Důvodem je skutečnost, že uživatel by musel umístit do domácnosti dvě různá zařízení na protilehlé pozice. To způsobuje komplikace s napájením a vzájemnou komunikací obou zařízení takovým způsobem, aby nedocházelo k degradaci estetiky interiéru. Dále by musela každá ze dvou částí zařízení obsahovat svůj mikrokontroler, čímž dále roste cena.

V úvahu tedy přišly následující technologie:

- Ultrazvukový senzor
- PIR senzor
- Radar
- ToF senzor

ToF senzor

ToF senzory používají měření doby letu (Time-of-Flight) odraženého infračerveného signálu. V dnešní době se jedná o již široce používanou technologii. Bohužel levnější senzory postavené na této technologii jsou poměrně citlivé na okolní intenzitu světla. S větší intenzitou světla klesá použitelný dosah a přesnost senzoru. Při přímém osvětlení senzoru slunečním zářením senzory nemusí dobře fungovat a může být ohrožen bezproblémový chod zařízení. Dražší ToF senzory jsou v tomto ohledu lepší, ale zde je nevyhovujícím parametrem právě cena.

Radar

Detekce pohybu osob na principu Dopplerova radaru je běžně používaná technika. Své uplatnění nachází například v detekci osob v osvětlení společných prostor v bytových domech, ale i jinde. Určitým problémem je ale příliš široké zorné pole a schopnost detekce pohybu i skrze zdi, která může být v této aplikaci rušivým elementem. Zde poté může docházet k falešně pozitivním detekcím. Malé radary jsou poměrně cenově dostupné s cenou okolo 100 Kč v době psaní práce.

PIR senzor

Velká výhoda PIR senzoru je ta, že reaguje pouze na předměty vydávající tepelné záření. Tímto se do určité míry řeší problém detekce neživých předmětů. Je žádoucí zaznamenávat pouze průchody lidí, nikoliv i ostatních objektů. Z tohoto důvodu může být PIR senzor kandidátem pro použití v aplikaci. Potenciální problém PIR senzorů může být detekce osob s velkou vrstvou oblečení izolující tepelné vyzařování.

Ultrazvukový senzor

Ultrazvukový senzor vysílá ultrazvukový signál a měří dobu, kterou trvá vrácení odraženého signálu zpět. V tom tkví i jeden z hlavních problémů ultrazvukového senzoru. Některé povrchy mají špatnou zvukovou odrazivost a tento signál pohltí. Senzor v tomto případě naměří zcela chybný výsledek. Tento problém se ale dá do určité míry vyřešit vhodným softwarem. Dalším problémem je samotná frekvence, která je často okolo 40 kHz. Ta je pro lidské ucho mimo slyšitelné spektrum, ale domácí mazlíčci jako

kočky, či psi tuto frekvenci mohou bez problému slyšet. Výhodou těchto senzorů je jejich velmi nízká cena. Ultrazvukový senzor lze sehnat za zhruba 25 Kč.

Vybrané senzory

Výsledkem úvah bylo použití senzorické fúze PIR senzorů s ultrazvukovými senzory vzdálenosti. Primárním důvodem pro použití těchto senzorů byla jejich velmi nízká cena. Díky tomuto parametru jsem měl oba tyto senzory k dispozici, čímž jsem mohl otestovat jejich použití v této aplikaci. Testováním byly tyto senzory shledány jako vhodné pro použití. Nasazením obou senzorů najednou lze potenciálně zvýšit přesnost detekce. Více v kapitole 10.

Pro ultrazvukový senzor byl zvolen populární HC-SR04. Jedná se o senzor s maximálním dosahem 4 m a minimální detekovatelnou vzdáleností 2 cm. Pracovní proudový odběr je 15 mA s napájecím napětím 5 V. Cenově se tento senzor pohybuje od 20-25 Kč, v závislosti na prodejci. Zmíněné parametry jsou pro navrhovanou aplikaci dostačující [46].

Obr. 9.1 *HC-SR04 [47]*

Jako použitý PIR senzor byl zvolen taktéž populární HC-SR501. Dokáže zaznamenávat pohyb až do vzdálenosti 7 m. Má nastavitelnou dobu, po kterou je výstup senzoru ve stavu „1“ poté, co byl detekován pohyb. Tato doba je nicméně i při nejkratším nastavení poměrně dlouhá, asi 5 s. Řešení tohoto problému je uvedeno v kapitole 10. Cenově se tento senzor taktéž pohybuje od 25 Kč. [48]

Obr. 9.2 *HC-SR501 [49]*

9.2 Mikrokontroler

Jako vhodného kandidáta pro použití v této aplikaci jsem zvolil čip ESP8266 od Espressif Systems. Jedná se o velmi populární mikrokontroler v DIY komunitě, své využití ale poslední dobou nachází i v komerčních produktech. Důvodem jeho popularity je možnost Wi-Fi připojení, 80 MHz frekvence čipu (možných až 160 MHz) a velmi nízká cena, kde právě cenou dokáže velmi dobře konkurovat podobným čipům. Také je programovatelný pomocí stejného jazyka jako velmi známé Arduino.

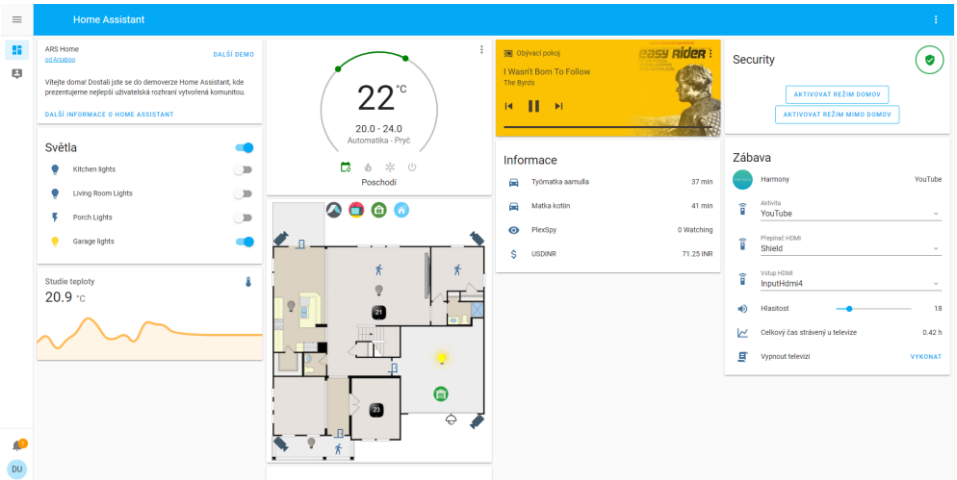
Jelikož existuje několik variací tohoto čipu, bylo nutné zvolit konkrétní vývojovou desku. Zvolil jsem vývojovou desku Wemos D1 Mini. Díky menším rozměrům oproti ostatním deskám bylo D1 Mini jednoznačným rozhodnutím. Deska má 9 digitálních I/O pinů a jeden analogový vstup. Dále disponuje 4 MB flash paměti a 64 kB SRAM.

Obr. 9.3 *Wemos D1 Mini [50]*

9.3 Systém pro domácí automatizaci – centrální brána

Na internetu je k dispozici velké množství open source platform pro domácí automatizaci. Jedná se o software pro zařízení na druhé úrovni architektury z kapitoly 4. Dvě nejvýraznější platformy jsou Open HAB a Home Assistant. Vzhledem k tomu, že mezi těmito platformami nejsou zásadnější rozdíly, rozhodl jsem se v této práci použít Home Assistant a to z důvodu, že jsem s tímto systémem již měl v minulosti zkušenost.

Pro zařízení, na kterém systém Home Assistant běží, byl zvolen jednodeskový počítač Raspberry Pi 3 B+.



Obr. 9.4 *Příklad uživatelského rozhraní Home Assistant [51]*

10 Vývoj zařízení a ověření funkčnosti

Pro vývoj detektoru průchodu jsem použil programovací jazyk C++ s nadstavbou jazyka Wiring, používaného zejména pro programování platformy Arduino. Program jsem psal v editoru Visual Studio Code s využitím modulu PlatformIO.

10.1 Vývoj a implementace systému detekce průchodů

10.1.1 Ověření základní myšlenky

Jelikož bylo podstatné snímat pouze lidský pohyb, nejdříve jsem se rozhodl otestovat přístup s dvěma PIR senzory. V počáteční fázi bylo nejprve nutné ověřit funkčnost idey uvedené v sekci 8.1.

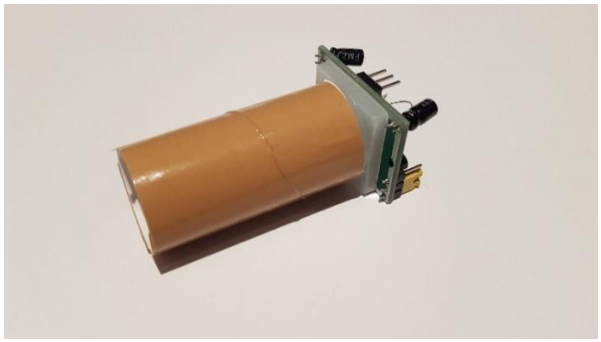
Hlavní princip detekce průchodů za pomocí dvou PIR senzorů je znázorněn na obr. 10.1. Použil jsem dva PIR senzory *„PIR_X“* a *„PIR_Y“*, které jsou v názorném kódu reprezentovány příslušnými proměnnými. Tyto proměnné nabývají pouze hodnot „1“ nebo „0“ v závislosti na tom, zda pohyb byl detekován, či nikoliv. PIR senzory jsem v rovině rozmístil tak, aby mezi nimi byla mezera zhruba 10 cm. Tyto dva senzory tvoří dvě poloroviny, jak bylo znázorněno v sekci 8.1 na obr. 8.2. Ve chvíli, kdy pouze jeden ze senzorů zaznamenal pohyb, uloží se, z které strany se uživatel pohybuje. Další fáze detekce probíhá tehdy, pokud oba senzory naráz zaznamenají pohyb. Podle údaje, ze které poloroviny uživatel přišel, systém dokáže rozhodnout jakým směrem uživatel prošel a upraví počty osob na příslušných stranách.

```
if(PIR_X xor PIR_Y)
    if(PIR_X)
        checkX = true
    else if(PIR_Y)
        checkY = true


if(checkY or checkX)
    if(PIR_X and PIR_Y)
        if(checkX)
            sideY = sideY + 1
            sideX = sideX - 1
        else if(checkY)
            sideX = sideX + 1
            sideY = sideY - 1
```

Obr. 10.1 *Princip detekce průchodu pomocí PIR senzorů*

S tímto přístupem se objevily problémy s přesnou detekcí průchodů. Tuto fundamentální implementaci jsem orientačně testoval pouze za použití dlaně, jakožto pohybujícího se objektu vydávající teplo. Hlavními identifikovanými problémy byly falešně pozitivní „průchody“, či zaregistrované „průchody“ ve špatném směru. Již po 20 testovacích „průchodech“ bylo 5 vyhodnoceno špatně, přesnost se tedy pohybovala okolo 75 %, což jsem vyhodnotil jako nevyhovující. Z tohoto důvodu nemělo smysl pokračovat v dalším testování této základní konfigurace. Nicméně se díky částečné funkčnosti ukázalo, že tento systém má potenciál pro další rozvoj.

Široké zorné pole senzoru HC-SR501, které činí zhruba 120°, jsem vyhodnotil jako nevhodné pro tuto aplikaci a identifikoval jsem tento parametr jako potenciální zdroj chybovosti. Není totiž žádoucí PIR senzorem monitorovat příliš velký úhel. Vhodnější je detekovat pouze pohyb přímo před senzorem, a to z důvodu, že pohyby v širším zorném poli mohou způsobit falešně pozitivní detekci průchodu. Pro zúžení zorného pole jsem na Fresnelovu čočku PIR senzoru nasadil papírový tubus.



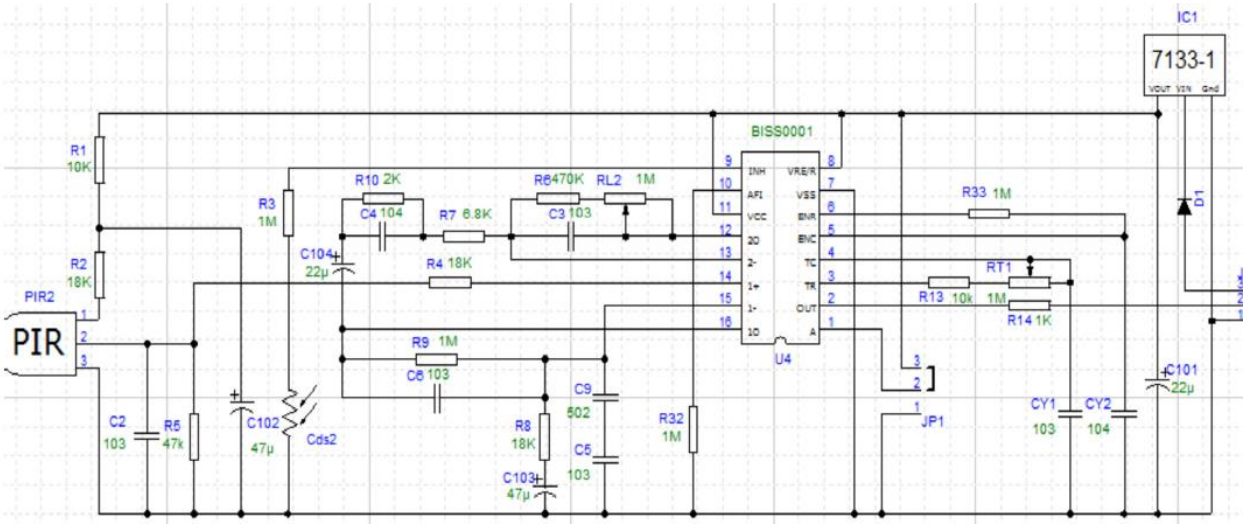
Obr. 10.2 *PIR senzor s tubusem pro omezení zorného pole*

Při tomto testu byl odhalen další problém PIR senzorů. HC-SR501 má trimr pro nastavení doby, po kterou je výstup senzoru po detekci pohybu ve stavu log. 1. Tato doba je nastavitelná v intervalu 5 s až 500 s. K tomuto intervalu se přičítá ještě doba cca 2-3 s, po kterou senzor ignoruje veškerý pohyb. Tato časová prodleva je pro reálný provoz špatně použitelná. V HC-SR501 je použitý čip BISS001, který je zodpovědný za vyhodnocení signálu z PIR senzoru a za odpovídající digitální výstup. V dokumentaci k tomuto čipu [52] jsou uvedeny vztahy pro výpočet výše zmíněných dob v závislosti na použitých součástkách. Ke vztahu s obvodem na obr. 10.3 jsou tyto vztahy:

$$Tx \approx 24576 \cdot R13 \cdot CY1 \quad (10.1)$$

$$Ti \approx 24 \cdot R33 \cdot CY2, \quad (10.2)$$

kde *Tx* je šířka pulsu v sekundách a *Ti* je doba, po kterou senzor ignoruje jakýkoliv pohyb.

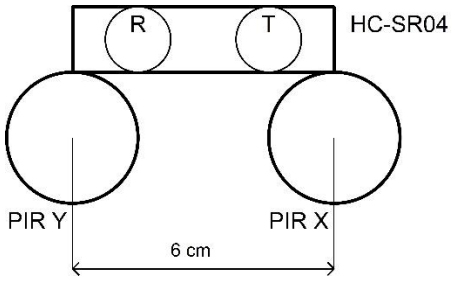


Obr. 10.3 *Schéma obvodu pro senzor HC-SR501 [53]*

Zkratováním příslušných rezistorů a odstraněním příslušných kondenzátorů podle vztahů (10.1) a (10.2), se mi podařilo dosáhnout výrazného zkrácení doby *Ti*, na dobu v řádu desítek až stovek ms. Perioda *Tx* se zkrátila z původních 5 *s* na zhruba 2 *s*. Tento problém tedy vyžadoval další řešení, které je uvedené dále v textu.

10.1.2 Senzorická fúze

Pro eliminaci chybných průchodů jsem k dosavadním PIR senzorům přidal ultrazvukový senzor vzdálenosti HC-SR04. Účelem tohoto senzoru v této verzi byla konfirmace zaznamenávaného pohybu. Tato verze je znázorněna na obr. 10.4. Písmena „R“ a „T“ na obrázku jsou zkratkami pro „Receiver“ (přijímač) a „Transmitter“ (vysílač).



Obr. 10.4 *Náčrt konfigurace detektoru průchodu, v1*

Před plným uvedením detektoru do provozu je nutné zjistit referenční hodnotu doby letu ultrazvukového signálu odpovídající vzdálenosti ultrazvukového senzoru od protilehlé překážky, nejčastěji zdi. Z toho důvodu jsem implementoval do detektoru kalibrační část programu, která má za úkol po spuštění zjistit právě tuto referenční hodnotu. Pro tuto aplikaci není nutné provádět přepočet na vzdálenost. Princip zjištění referenční časové hodnoty je následující. Ultrazvukový senzor změří 100 hodnot časů echa s pauzou 10 ms mezi každým měřením. Každý výsledek měření je uložen do pole. Toto pole je pomocí algoritmu řazení slučováním (merge sort) seřazeno od nejmenších prvků po největší. Vynecháním spodních 10 a horních 10 hodnot vznikne soubor 80 hodnot, ze kterých se počítá aritmetický průměr, který je pak považován za referenční hodnotu doby letu ultrazvukového signálu, která je úměrná vzdálenosti mezi senzorem a zdí. V případě podezření na špatně provedený proces kalibrace (referenční hodnota moc nízká, či naopak vysoká), se tento proces opakuje znovu.

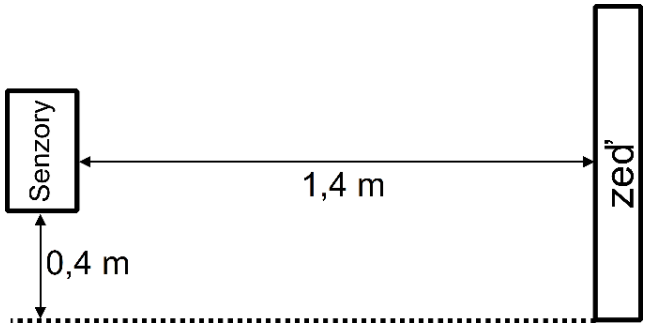
Samotný navržený princip konfirmace pohybu je pak znázorněn na obr. 10.5. Tento proces se spustí ve chvíli, kdy zařízení zaznamená podezření na průchod pomocí PIR senzorů. Celý tento proces běží v cyklu. Celkem tedy proběhne několik měření, při kterých se testuje přítomnost objektu. Funkce *„readHCSR04“* má jako návratovou hodnotu změřenou dobu letu ultrazvukového signálu jedním směrem v µs. Konstanta 9000 je čas v µs odpovídající zhruba 3 m vzdálenosti mezi senzorem a překážkou. Předpokládá se, že většina průchodů bude v daleko menší vzdálenosti od zařízení. Nicméně pokud je změřený čas větší než 9000 µs, pohyb je možné konfirmovat jako správný. Důvodem je špatná zvuková odrazivost některých povrchů, například různých druhů oblečení. V případě, kdy je zvukový signál pohlcen, je hodnota proměnné *echo* na obr. 10.5 pravděpodobně rovna maximální hodnotě bufferu senzoru, která se pohybuje v desítkách tisíc – signál se již nevrátí zpět do přijímače. Konstanta 450 µs odpovídá vzdálenosti zhruba 15 cm. Od protilehlé překážky je tedy 15 cm mrtvé pásmo, kde nedochází ke konfirmaci průchodu při naměření hodnoty spadající do tohoto pásma. Hodnota byla zvolena tak, aby byla dostatečná tolerance chyby, ale zároveň musela být dostatečně malá, aby uživatel procházející blízko protilehlé překážky tvořil větší rozdíl ve vzdálenostech, než uvedených 15 cm. S touto konfigurací jsem provedl testy, jejichž metodika a výsledky jsou uvedeny dále v textu.

```
for(i = 0; i < 50; i++)
    echo = readHCSR04()

    if(echo >= 9000)
        return 2

    else if(reference_echo > echo)
        echo = reference_echo - echo
        if(echo >= 450)
            return 1
```

Obr. 10.5 *Princip konfirmace pohybu*

Metodika testování

Schéma testování je zobrazené na obr. 10.6. Předmětem testů byla přesnost detekce příčných průchodů mezi senzory a zdí. Vždy byl proveden uvedený počet průchodů jedním směrem a poté zpět, viz tabulky níže. Při chybně detekovaném průchodu jsem tuto chybu poznamenal. Chybný průchod znamenal jednu ze tří možností. Průchod nebyl detekován, byl detekován falešně pozitivní průchod nebo byl chybně určen směr průchodu. Celkový počet chyb v daném směru je uveden ve výsledcích testování. Součtem počtu chyb v daných směrem vznikl celkový počet chybných vyhodnocení z celkového počtu průchodů. Následně byla spočítána procentuální hodnota správně vyhodnocených průchodů ze všech provedených průchodů. Tato hodnota je považována za výslednou přesnost detekce. Rychlost pohybu při těchto průchodech odpovídala průměrné hodnotě 1,2 $m \cdot s^{-1}$. Před zahájením každého průchodu bylo nutné vyčkat na ustálení PIR senzorů do stavu log. 0 poté, co detekovali pohyb předchozího průchodu, tedy zhruba 3 s. Tato doba se občas mohla prodloužit až na 5-6 s při falešné „reaktivaci“ PIR senzoru po průchodu.



Obr. 10.6 *Schéma testování*

test #	1
Počet průchodů (25X+25Y)	50
Chybně X	5
Chybně Y	3
Chybně celkem	8
Přesnost X [%]	80
Přesnost Y [%]	88
Přesnost [%]	84

Tab. 10.1 *Test 1 – první konfigurace*

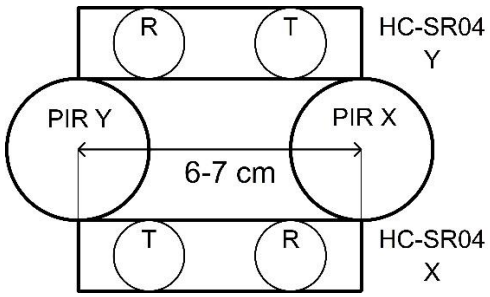
Test, jehož výsledky jsou k dispozici v tab. 10.1, objevil několik nedostatků první verze. Na obr. 10.4 je počet iterací cyklu roven 50. Při tomto testování ale cyklus iteroval pouze 10x, k uvedenému počtu 50 iterací jsem dospěl až se změnami provedenými dalším vývojem. Obecně se tento počet ukázal jako poměrně důležitý faktor pro přesnost. Při nízkých počtech iterací docházelo k případům, kdy průchod nebyl zaznamenán, protože cyklus skončil dříve, než objekt stihl dojít do takové polohy, kde by se spolehlivě odrazil vysílaný ultrazvukový signál zpět do přijímače. Zvýšením počtu iterací cyklu se tak prodloužil potřebný čas pro dokonání průchodu skrze detektor.

Po této úpravě počtu iterací na hodnotu 50 jsem provedl další test, jehož výsledky shrnuje tab. 10.2. Zde je zřejmé, že zvýšením počtu iterací se zvýšila přesnost. Částečně se tak odstranil problém falešně negativních detekcí průchodu.

test #	2
Počet průchodů (25X+25Y)	50
Chybně X	4
Chybně Y	0
Chybně celkem	4
Přesnost X [%]	84
Přesnost Y [%]	100
Přesnost [%]	92

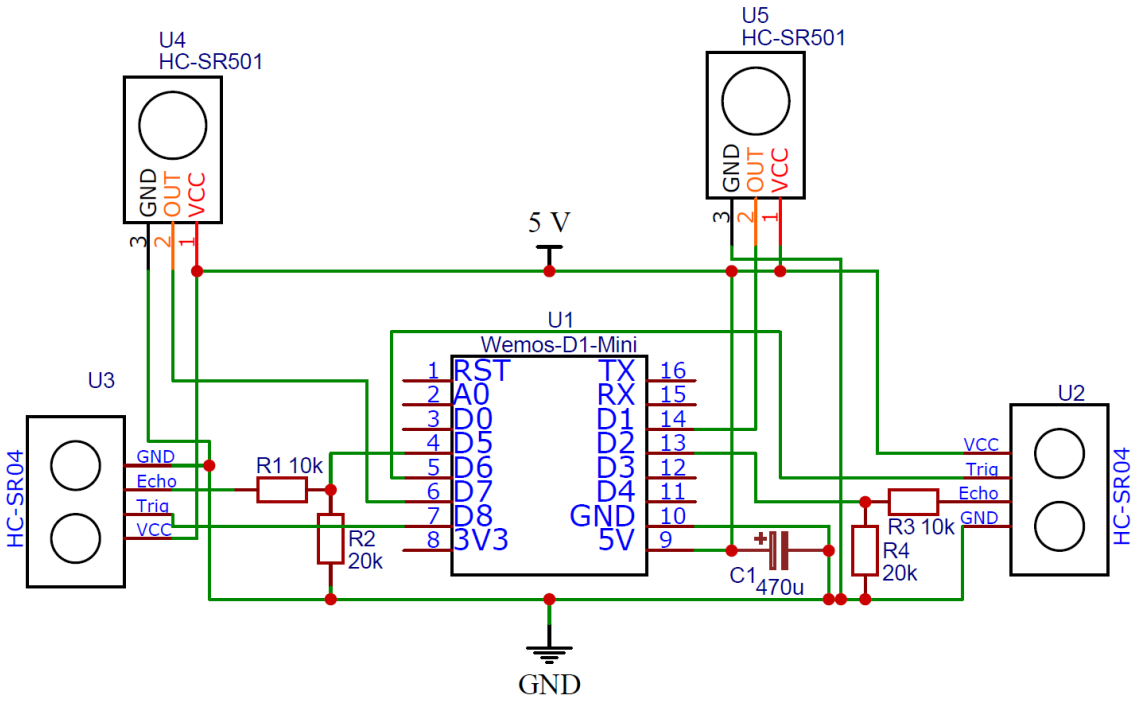
Tab. 10.2 *Test 2 - úprava počtu iterací v cyklu*

Z již provedených testů, ale i z dalšího testování bylo zřejmé, že přesnost detekce průchodu do poloroviny „X“ byla horší, nežli přesnost detekce průchodů do poloroviny „Y“. Jako vysvětlení se nabízí fyzická konfigurace senzorů. Vyšší přesnost detekce průchodů do poloroviny „Y“ mohla být ovlivněna skutečností, že vysílač senzoru HC-SR04 byl umístěn na té straně, ze které směřuje pohyb, tedy blíže senzoru „PIR_X“, viz obr. 10.4. Jako řešení tohoto problému jsem přidal další senzor HC-SR04, který byl otočen o 180° oproti prvnímu HC-SR04. Při podezření na průchod do poloroviny „Y“ byl pro konfirmaci použitý senzor „HC-SR04 Y“, naopak při podezření na průchod do poloroviny „X“ byl použit senzor „HC-SR04 X“. Tato konfigurace je znázorněna na obr. 10.7.



Obr. 10.7 *Náčrt konfigurace detektoru průchodu, v2*

Na obr. 10.8 je uvedeno schéma zapojení navrhované konfigurace. Jelikož ESP8266 nemá 5 V tolerantní vstupy, bylo nutné snížit výstupní napětí senzorů HC-SR04 (ECHO pin) na hodnotu 3,3 V. Toto snížení jsem provedl pomocí napěťového děliče s použitím příslušných hodnot rezistorů. Aby děličem neprotékal zbytečně vysoký proud, byly zvoleny hodnoty rezistorů 10 kΩ a 20 kΩ. Při uvážení hodnoty 5 V jako výstup pinu ECHO je celkový proud protékající děličem roven hodnotě 0,16 mA. Pro zvýšení stability napájecího napětí a pokrytí špiček proudu jsem mezi 5 V pin a GND pin umístil elektrolytický kondenzátor o hodnotě 470 µF.



Obr. 10.8 *Schéma zapojení detektoru průchodu*

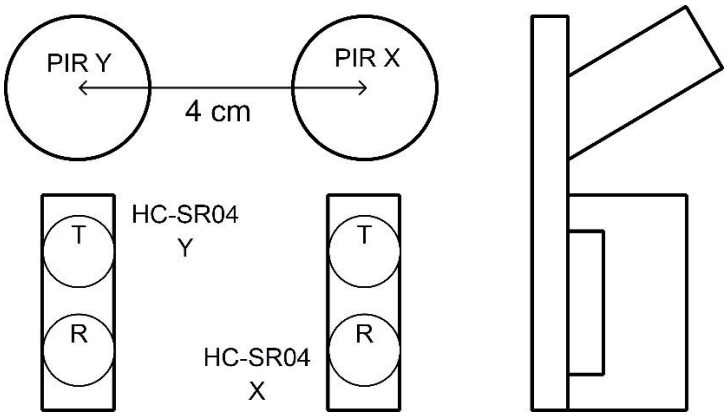
Výše zmíněnou hypotézu o nepřesnosti způsobené pouze jedním použitým ultrazvukovým senzorem jsem ověřil dalším testováním, jehož výsledky jsou v tab. 10.3. Testy hypotézu potvrdily a ukázalo se, že změny konfigurace měly pozitivní výsledky. Z celkového počtu 100 průchodů se dosáhlo 100 % přesnosti detekce. Nicméně je nutno zmínit, že se stále jednalo o testování v jakýchsi ideálních podmínkách. Ty spočívaly ve zmíněném čekání na ustálení PIR senzorů a také ve snaze o „ideální“ průchod skrze detektor (správná rychlost, nohy nezastiňují senzor, atd.).

test #	3
Počet průchodů (50X+50Y)	100
Chybně X	0
Chybně Y	0
Chybně celkem	0
Přesnost X [%]	100
Přesnost Y [%]	100
Přesnost [%]	100

Tab. 10.3 *Test 3 - přidání dalšího HC-SR04*

10.1.3 Finální verze

V návrhu další verze jsem provedl několik nutných změn v rozmístění senzorů. Tyto změny jsem provedl na základě zkušeností z reálného používání zařízení. Tato nová a finální konfigurace je znázorněna na obr. 10.9. Jelikož již nebyly přidány žádné senzory ani součástky, je schéma z obr. 10.8 aktuální i v této verzi.

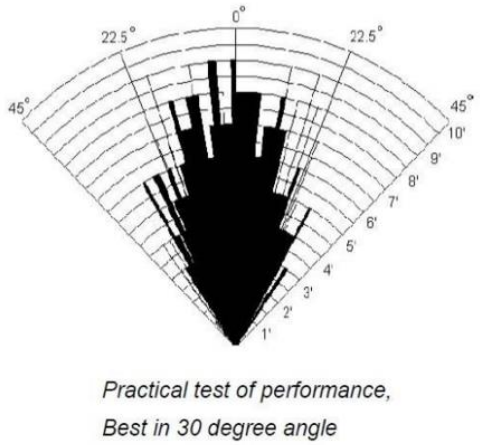


Obr. 10.9 *Náčrt konfigurace detektoru průchodu, v3*

První výraznou změnou bylo umístění PIR senzorů. Senzory s tubusy pro omezení zorného pole v této verzi směřují nahoru pod úhlem 25° vůči rovině. Úhel jsem zvolil jako kompromis mezi výškovými rozdíly v případech, kdy uživatel prochází blízko, a kdy naopak daleko od detektoru. Tato změna byla učiněna vzhledem k pozorování, že při nízko položeném detektoru, zhruba v úrovni kotníků, není detekce pohybu pomocí PIR senzorů přesná. Jako vysvětlení se nabízí celkově menší plocha lidské nohy v této části, či možnost, že uživatel má oděv, který právě v oblasti kotníků odstává a není tedy zahřátý na lidskou teplotu. Ve vzdálenosti d = 50 cm od detektoru, což je považováno za blízkou vzdálenost, činí výškový rozdíl způsobený nakloněním PIR senzorů pod úhlem 25° Δh = 23 cm. Při výšce detektoru nad zemí $h = 20\ cm$ tak míří PIR senzory zhruba na úroveň horní části lýtkového svalu. Při vzdálenějších průchodech ve vzdálenosti $d = 120\ cm$ a zachování výšky detektoru je pak Δh = 56 cm. Tento výškový rozdíl zajistí směřování PIR senzorů na stehenní sval průměrně vysokého člověka. Oba tyto svaly jsou používané při chůzi a jsou tak dostatečným zdrojem tepla pro zaregistrování PIR senzorem.

Další změnou bylo rozmístění ultrazvukových senzorů HC-SR04. Ty jsem v této verzi přesunul vertikálně pod příslušné PIR senzory. Tato konfigurace umožnila provést několik změn a implementovat nové funkce. Jednou z nich byla konfirmace pohybu pro každý PIR senzor zvlášť. Pokud tedy PIR senzor zaznamená pohyb, ihned musí být konfirmován příslušným ultrazvukovým senzorem.

Jedním z dalších identifikovaných problémů bylo zorné pole senzoru HC-SR04, které je až 30°, viz obr. 10.10. Problém zorného pole je podobný jako v případě PIR senzorů. K problémům docházelo zejména v případech rychlých průchodů, které jsou detekovány pouze ultrazvukovými senzory, viz text dále. Pro omezení tohoto problému jsem mezi senzory umístil přepážku omezující jejich zorné pole.



Obr. 10.10 *Zorné pole senzoru HC-SR04 [54]*

Hlavní funkcí, kterou umožnila tato nová konfigurace implementovat, byla možnost detekce průchodů pouze pomocí ultrazvukových senzorů. Tato funkce je možným řešením problému, kdy je výstup PIR senzorů ve stavu log. 1 a čeká se na jejich ustálení do stavu log. 0. Ultrazvukové senzory tak vyplňují tuto několikavteřinovou mezeru. Princip implementace je uveden na obr. 10.11. Názorný kód na obr. 10.11 se volá s každým průchodem hlavní smyčkou programu. Ve výsledku se tak ve velmi rychlém sledu za sebou střídá detekce pohybu pomocí funkce *„motionHCSR04“* v jednotlivých polorovinách „X“ a „Y“. Funkce *„motionHCSR04“* má návratovou hodnotu typu boolean, závisející na tom, zda byl nebo nebyl detekován pohyb. Pokud na jedné z nich dojde ke zjištění pohybu, začne druhý senzor již známou sekvenci konfirmaci pohybu z obr. 10.5.

```
 1 test = motionHCSR04(X)
 2 if(test)
 3     if(confirmMotion(Y))
 4         sideY = sideY + 1
 5         sideX = sideX - 1
 6
 7 test = motionHCSR04(Y)
 8 if(test)
 9     if(confirmMotion(X))
10         sideX = sideX + 1
11         sideY = sideY - 1
```

Obr. 10.11 *Princip detekce pohybu pomocí ultrazvukových senzorů*

Pro další hlubší ladění zařízení bylo nutné zjistit, jaká je průměrná rychlost pohybu uživatelů v domácnosti. Pro tyto účely jsem provedl experiment se čtyřmi uživateli. Každý prošel úsek známé délky a měřil se čas tohoto průchodu. Toto měření se pro každého uživatele opakovalo desetkrát. Následně byly tyto hodnoty zprůměrovány. Celková průměrná rychlost všech uživatelů byla určena na v_{avg} = 1,25 m·s^{-1}. Nejnižší průměrná rychlost uživatele činila v_{min} = 1,14 m·s^{-1} a nejvyšší v_{max} = 1,37 m·s^{-1}. Při vzdálenosti ultrazvukových senzorů 40 mm a při uvažování průměrné rychlosti v_{avg} = 1,25 m·s^{-1} je čas nutný pro uražení této vzdálenosti roven 32 ms. Díky tomuto poznatku bylo možné lépe navrhnout časování zpoždění jednotlivých funkcí v softwaru a tím zlepšit spolehlivost detekce.

Pro finální ověření funkčnosti jsem opět provedl testování. Metodika testování zůstala stejná jako při předchozích testech s tím, že tentokrát byly provedené průchody v co největší míře přirozené. Tímto testem jsem se snažil simulovat reálný provoz navrženého detektoru průchodu. Nejprve byla testována přesnost detekce průchodů s vyčkáním na ustálení PIR senzorů do stavu log. 0, tedy s intervalem max. 5-6 s. Následně byla testována přesnost zařízení, při více průchodech v rychlém sledu za sebou. Druhý test tak simuloval průchod větší skupiny uživatelů s malým odstupem mezi jednotlivými uživateli. Testovala se zde tedy přesnost implementovaného algoritmu pro detekci průchodu pomocí ultrazvukového senzoru.

test #	4
Počet průchodů (75X+75Y)	150
Chybně X	0
Chybně Y	0
Chybně celkem	0
Přesnost X [%]	100
Přesnost Y [%]	100
Přesnost [%]	100

Tab. 10.4 *Test 4 - změna konfigurace senzorů čekání na ustálení PIR senzorů (> 3-5 s)*

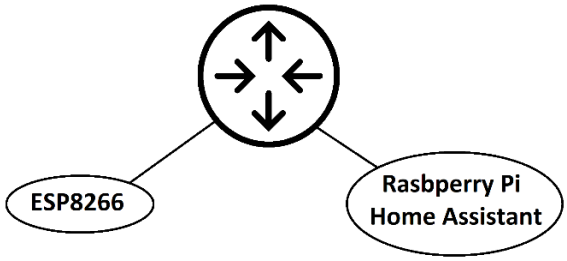
test #	5
Počet průchodů (30X+30Y)	60
Chybně X	3
Chybně Y	0
Chybně celkem	3
Přesnost X [%]	90
Přesnost Y [%]	100
Přesnost [%]	95

Tab. 10.5 *Test 5 – změna konfigurace senzorů rychlé průchody (< 2 s)*

Test č. 4 potvrdil poměrně dobrou funkčnost zařízení v testovacích podmínkách. Test č. 5 naopak ukázal prostor pro zlepšení implementace ultrazvukové detekce průchodů. Vzhledem k faktu, že tento systém detekce je pouze jakýmsi záložním systémem, se ale stále jedná o použitelnou přesnost. Celková bilance z testů č. 4 a 5 je tedy 207 správně vyhodnocených průchodů z celkem 210, což procentuálně odpovídá hodnotě 98,57 %. Tento údaj je ale do určité míry zkreslený právě tím, že méně spolehlivá část v tomto testu tvoří téměř 30 % hodnot. Reálně ale poměr těchto dvou situací bude více ve prospěch jednotlivých průchodů s delším časovým intervalem.

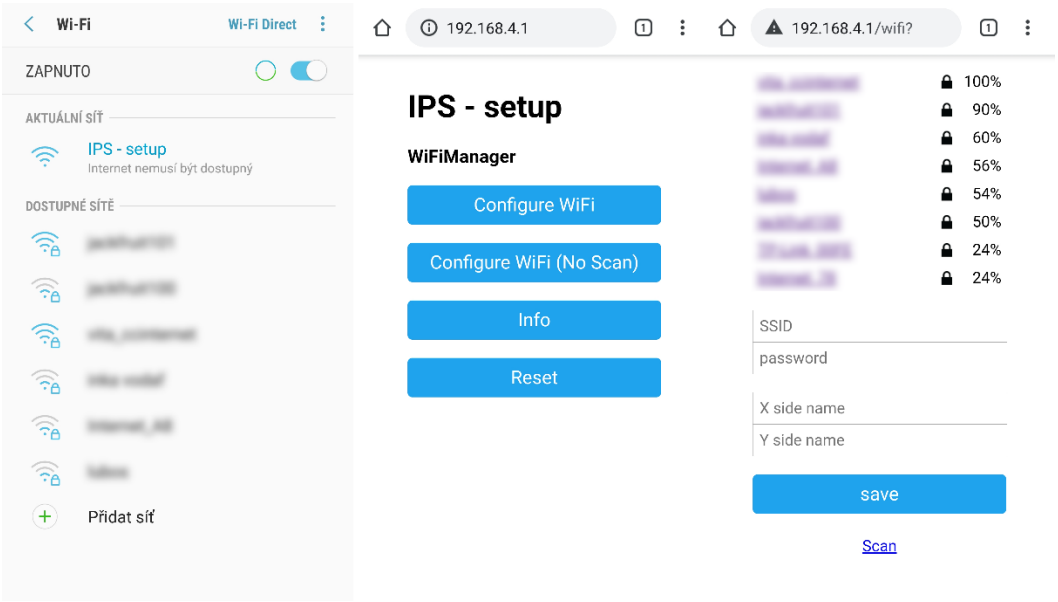
10.2 Vývoj a implementace síťové funkcionality

Jako hlavní úkol v této části jsem si zadal implementovat podporu pro systém Home Assistant. Dále bylo nutné vytvořit systém rozvržení domácnosti, zajistit správné odesílání zpráv a synchronizaci dat mezi jednotlivými detektory. Nezbytné bylo, aby celý proces prvotního nastavení byl co nejvíce uživatelsky přívětivý.



Obr. 10.12 *Schéma síťové komunikace*

Prvním krokem pro komunikaci s Home Assistantem bylo připojit detektor do sítě WLAN. Abych se vyhnul zadávání SSID a PSK do zdrojového kódu, použil jsem pro tyto účely knihovnu WiFiManager [55]. Tato knihovna umožňuje přepnout zařízení do AP módu a po následném připojení mobilním zařízením umožní uživateli zadat potřebné údaje pro připojení do WLAN. WiFiManager také umožňuje nastavit další libovolné údaje, které jsou po uživateli požadovány. V tomto případě se bude jednat o názvy dvou zón, které zařízení tvoří. Celý proces nastavení je vyobrazen na obr. 10.13.



Obr. 10.13 *Proces prvotního nastavení zařízení*

Po připojení uživatele k příslušné síti (zde „IPS – setup“), musí uživatel zadat do prohlížeče IP adresu 192.168.4.1, což je adresa samotného ESP8266. Dále je uživateli prezentováno menu, kde po zvolení „Configure WiFi“ si může uživatel zvolit SSID a zadat příslušné heslo. Poté je zde nutné vyplnit „X side name“ a „Y side name“, tedy jména příslušných zón. Po stisknutí tlačítka „save“ zajistí

WiFiManager stažení zadaných údajů. Pro dlouhodobé uchování těchto údajů jsem implementoval jejich ukládání do souborového systému SPIFFS ve flash paměti ESP8266. Při dalším spuštění již tedy uživatel tímto nastavením procházet nemusí.

Pro odesílání informací o počtu osob v monitorovaných zónách posílá navržený detektor zprávy na dvě rozdílná MQTT témata. Názvy těchto témat jsou tvořeny pomocí zadaných názvů zón a to následovně:

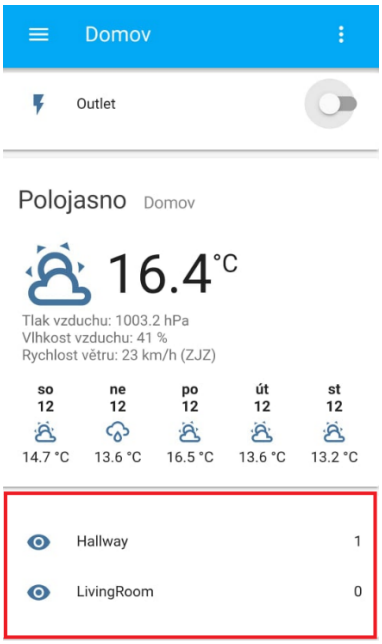
X side name: LivingRoom → **X publish topic**: IPS/LivingRoom

Y side name: Hallway → **Y publish topic**: IPS/Hallway.

Tato témata jsou identická s těmi, k jejichž odběru se zařízení přihlašuje. Odběr těchto témat je nutný z důvodu synchronizace počtu osob v případě, že je v domácnosti více zařízení, která sdílejí stejnou zónu. Tento případ může nastat, pokud je do jedné zóny více vchodů nežli jeden.

Aby detektor mohl komunikovat se systémem Home Assistant, musí znát jeho IP adresu. Home Assistant podporuje protokol mDNS, tudíž se tento způsob ukázal jako nejefektivnější, jak již bylo dříve deklarováno. Pro tyto účely jsem použil knihovnu mDNSResolver [56]. Home Assistant má defaultní lokální doménu *„hassio.local“*. Po spuštění provede detektor mDNS dotaz na tuto doménu, která následně vrátí svou IP adresu.

Dále bylo nutné provést samotnou integraci zařízení do systému Home Assistant. Zde je většinou nutné zařízení manuálně nastavit editací souboru *configuration.yaml*. Pro koncového, netechnicky orientovaného uživatele toto ale může být poměrně komplikovaný úkon. Z tohoto důvodu jsem se rozhodl využít možnost tzv. MQTT Discovery [57]. Tato funkce spočívá v tom, že na téma, jehož přesný název je definovaný v [57], se ve formátu JSON odešlou konfigurační informace ohledně připojovaného zařízení, jako je jeho název, či MQTT téma pro publikaci zpráv. Po odeslání této konfigurační zprávy je zařízení schopné plného provozu v systému Home Assistant. Zde uživatel najde dvě entity, kde každá entita symbolizuje jednu monitorovanou zónu. Z pohledu systému Home Assistant tedy existují pouze tyto entity spjaté se zónami, nikoliv jednotlivé detektory v celé domácnosti. Pro uživatele toto znamená snadné nastavení v prostředí Home Assistant a jednoduchý přístup k potřebným informacím.



Obr. 10.14 Zařízení integrované do prostředí Home Assistant

10.3 Napájení, 3D návrh krabičky, cena

10.3.1 Napájení zařízení

Idea napájení zařízení je, že v mnoha domácnostech je nedaleko dveří umístěna zásuvka. Zařízení by se mohlo do této zásuvky potřebnou orientací zapojit (je nutné zajistit příčný průchod přes senzory, viz obr. 8.2 a obr. 10.17)

Pro napájení zařízení jsem použil modul AC/DC zdroje s výstupním napětím 5 V a max. proudovým odběrem 700 mA. Výkon zdroje tedy odpovídá hodnotě 3,5 W.

Proudové odběry jednotlivých komponent při 5 V jsou

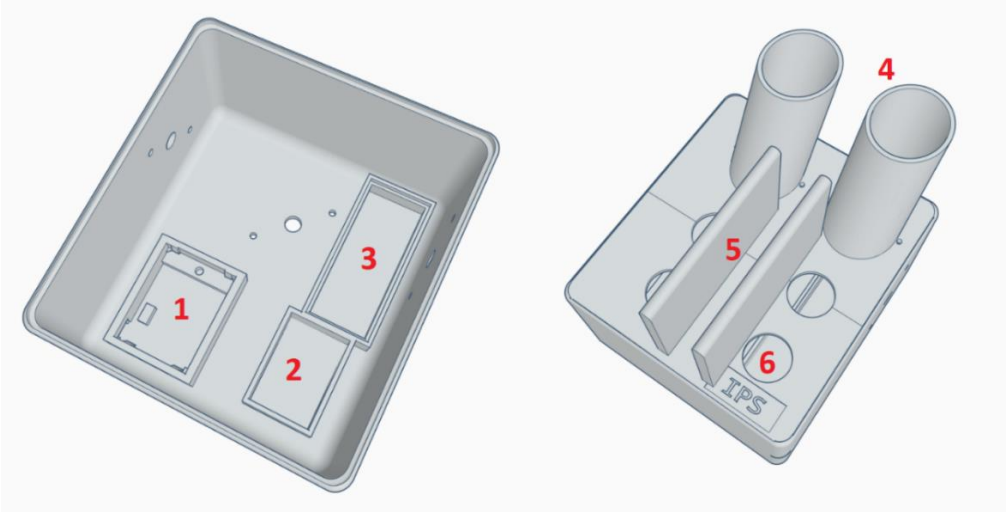
- HC-SR501: I_{max} = 65 mA [48]
- HC-SR04: I_{max} = 15 mA [46]
- Wemos D1 mini: I_{max} = 80 mA (proudové špičky až 250 mA) [58]

Po sečtení maximálních hodnot proudového odběru dosahuje maximální možný proudový odběr hodnoty I_{MAX} = 410 mA, což je pod limitem použitého zdroje.

Obr. 10.15 Použitý AC/DC zdroj [59]

10.3.2 3D návrh krabičky

Návrh krabičky pro 3D tisk jsem tvořil za použití softwaru TinkerCAD. Celkové rozměry zařízení jsou 92 x 113,1 x 90,7 mm. Tento návrh jsem poté na 3D tiskárně vytisknul.

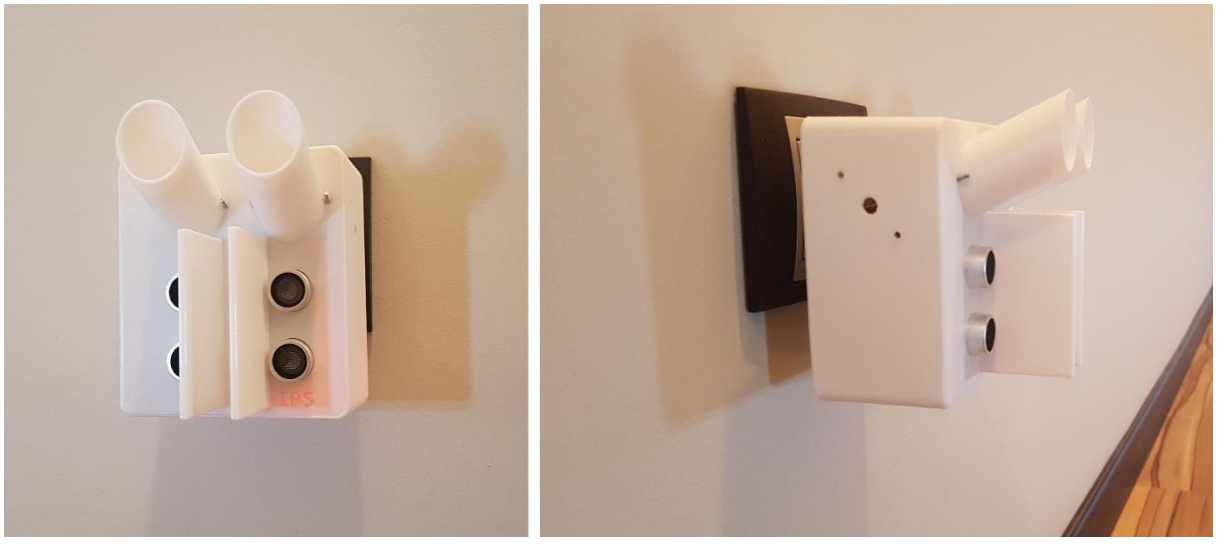


Obr. 10.16 *Výsledný 3D model*

Dále jsou popsány jednotlivé části 3D návrhu:

- 1) Slot pro Wemos D1 mini
- 2) Slot pro zdroj
- 3) Slot pro PCB
- 4) Senzory HC-SR501 + tubusy pro omezení jejich zorného pole
- 5) Přepážky pro omezení zorného pole senzorů HC-SR04
- 6) Slot pro senzory HC-SR04

Na krabičce jsou dále dírky pro upevnění vidlice do 230 V zásuvky. Pro všechny možné orientace zařízení jsou tyto dírky po všech stranách. Na obr. 10.17 je zapojené finální zařízení s vidlicí umístěnou na zadní stěně.



Obr. 10.17 *Finální produkt v provozu*

10.3.3 Cena

Ceny použitých komponent jsou uvedeny v tab. 10.6. Uvedené ceny jsou z e-shopu ebay.com.

Komponenta	Cena [Kč]
2x HC-SR04	40
2x HC-SR501	36
Wemos D1 Mini	70
Zdroj 3,5 W	30
Materiál 3D Tisk	30
Vidlice 230 V	20
Součástky, kabely	5
Cena celkem	**231**

Tab. 10.6 *Cenový rozbor zařízení*

Celková výrobní cena zařízení je odhadnuta na 231 Kč. V porovnání s cenou senzorů použitých v zařízení na bázi ToF, zmíněných v sekci 7.3, je tato cena velmi přívětivá. U těchto senzorů se jejich cena pohybovala od zhruba 100 € (v době psaní práce tato částka odpovídá zhruba 2750 Kč).

Je nutné dodat, že při hromadné výrobě by cena jednotlivých komponent díky většímu množství pravděpodobně klesla. Další úsporou by byla výroba krabičky pomocí jiné technologie nežli použitého 3D tisku.

11 Problémy navrženého zařízení a možnosti dalšího vývoje

I přes poměrně slibné výsledky testů má navržené zařízení některé nedostatky, které je pro bezchybné použití v reálné domácnosti nutné odstranit.

Jeden z největších problémů se projevuje v situaci, kdy se uživatel před zařízením zastaví. V této situaci začne zařízení náhodně přičítat falešně detekované „průchody“ a počet osob v domácnosti již neodpovídá realitě. S tímto problémem souvisí i fakt, že použité ultrazvukové senzory fungují na kmitočtu 40 kHz. Tato frekvence je ve slyšitelném spektru domácích mazlíčků, jako jsou psi nebo kočky. Některá zvířata mohou reagovat zvídavě a před zařízení si stoupnou, což je poznatek z vlastní zkušenosti. Tím opět dochází k náhodné detekci falešných průchodů. Řešením tohoto problému by bylo implementovat systém, který by pomocí ultrazvukových senzorů měřil vzdálenost od statického objektu. Pokud by zařízení nezaznamenalo změnu vzdálenosti, nepřičetlo by průchod. Systém ale musí být schopen po delším intervalu stání před senzorem spolehlivě zaznamenat skutečný průchod – tedy dokonání průchodu v původním směru, či návrat zpět.

Další identifikovaný problém souvisí s implementací detekce pomocí ultrazvukových senzorů v sekci 10.1.3. Díky této funkci se nyní po průchodu osoby detekuje i pohyb „neživých“ objektů, právě pomocí ultrazvukových senzorů. Typická problémová situace nastává v případě, pokud uživatel projde do pokoje a ihned za sebou zavře dveře. Zde se jako průchod detekuje i zavření dveří. Toto předtím díky detekci výhradně pomocí PIR senzorů nebyl problém. Řešení tohoto problému je ale poměrně komplikované. Ideálním řešením by bylo použití PIR senzorů s rychlejším přechodem do stavu log. 0 po detekci pohybu, bez nutnosti použití režimu s ultrazvukovými senzory, čímž by se i zvýšila přesnost vícenásobných průchodů s krátkým intervalem mezi nimi. Senzory s těmito parametry ale nebyly na trhu nalezeny.

Pro reálné použití v domácnosti je nutné se dále zabývat zvyšováním přesnosti. Bylo by vhodné dále experimentovat s různými druhy senzorů, na které by se měl další vývoj zaměřit. Například vhodnější výběr senzorů vzhledem k jejich velikosti, aby bylo možné detektor zmenšit.

Dále je nutné navržený detektor průchodu testovat při vyšších teplotách. Jelikož PIR senzory fungují na principu rozdílu teplot mezi povrchy, je potenciálně možné, že při extrémních venkovních teplotách dosáhne teplota v domácnosti takové úrovně, že PIR senzory nebudou schopny detekovat tento teplotní rozdíl mezi povrchem a uživatelem. V době psaní této práce a navrhování detektoru nedosahovaly teploty takových úrovní, aby bylo tuto hypotézu možné ověřit. Maximální teploty v domácnosti se pohybovaly do 24 °C. V těchto podmínkách fungoval navržený detektor průchodu tak, jak je uvedeno ve výsledcích testování.

Zapojené zařízení zabírá celou zásuvku. To je pro uživatele poměrně velkým omezením. Pro odstranění tohoto problému by bylo potřeba vyvést použitelnou zásuvku se síťovým napětím, například ze spodní strany zařízení. Uživatel by tedy mohl potřebné spotřebiče zapojit přímo do navrženého detektoru průchodu.

12 Závěr

V První části práce jsem se zabýval problematikou návrhu univerzální architektury pro systémy domácí automatizace. Mým cílem bylo navrhnout architekturu, která by sjednocovala dosavadní i budoucí zařízení pro chytrou domácnost. Definované požadavky na tuto architekturu byly zejména bezpečnost, cena, spolehlivost a flexibilita. Nejdříve byla celá komplexní architektura rozdělena do čtyř vrstev. Tyto vrstvy byly tvořeny podle jednotlivých úrovní zařízení v chytré domácnosti. Každá vrstva byla abstrakcí určitých druhů zařízení. V těchto vrstvách jsem navrhl principy, kterými by se měl návrh příslušných druhů zařízení řídit. První vrstva byla „vrstva koncových zařízení“. Zabývala se návrhem koncových akčních členů a senzorů. Jako vhodný komunikační protokol pro tato zařízení byl zvolen protokol MQTT. Pro řešení bezpečnosti u těchto zařízení bylo shledáno jako vhodné použití protokolu ECDH pro generaci soukromých klíčů společně se symetrickým šifrováním pomocí AES. Jako vhodné datové formáty pro komunikaci koncových zařízení byly identifikovány formáty JSON a MessagePack. Druhá vrstva architektury pojednávala o návrhu centrální brány pro systém domácí automatizace. Tato brána byla navržená jako modulární vzhledem k možnostem použití různých komunikačních technologií. Bylo navrženo řešení problému SPOF v podobě implementace MQTT Broker Clusteru. Pro organizaci zařízení a automatizací byly zavedeny konfigurační a automatizační soubory uložené v paměti brány. Předložené koncepty jako „gateway computing“ nebo „cloud offloading“ mají potenciál dále snížit potřebný výkon koncových zařízení, resp. bran a snížit tak jejich cenu. Třetí vrstvou byla vrstva zabývající se technologií cloudu. Kromě použitých komunikačních technologií bylo v cloudové vrstvě navrženo několik nových přístupů. Jedním z nich je například právě „cloud offloading“, či použití AI v systému domácí automatizace. Poslední vrstva se v krátkosti zabývala návrhem mobilních aplikací pro chytrou domácnost. Pro reálné použití této architektury v procesu vývoje nových zařízení pro chytrou domácnost je nutné dále precizovat uvedené principy. Pro budoucnost technologie chytrých domácností je nezbytné vytvoření unifikovaného standardu, který by stavěl na podobné architektuře, jako je tato.

V práci jsem se následně zaměřil na aplikace použitelné v prostředí domácí automatizace. Celkem jsem navrhl tři takové aplikace. U navržených aplikací byla podstatná vysoká míra automatizace, která je považována za nejdůležitější faktor pro uživatelské pohodlí. První z navržených aplikací měla za cíl uživateli poskytnout pohodlí a určité finanční úspory, a to zejména v horkých letních dnech. Hlavní myšlenka byla založena na optimalizaci využívání okenní ventilace a využívání žaluzií pro pasivní regulaci teploty a osvětlení v domácnosti. Kvalitní spánek a lepší zdraví bylo tématem pro druhou navrženou aplikaci. Zde bylo využito technologie domácí automatizace pro dosažení ideálního prostředí pro spánek. Třetí aplikací byla lokalizace osob v domácnosti. Zde byl identifikován velký potenciál pro přínos uživateli a zároveň byl spatřen prostor pro vývoj nového přístupu k tomuto problému.

V poslední části jsem třetí z navržených aplikací prakticky realizoval. Nejprve jsem uvedl myšlenku pro realizaci systému lokalizace osob v domácnosti. Tato myšlenka spočívala v detekci průchodů uživatelů mezi zónami v domácnosti. Detekcí těchto průchodů je možné monitorovat počet osob v jednotlivých zónách. Následně jsem detektor průchodu navrhl a prakticky realizoval. Průchody byly detekovány využitím senzorické fúze PIR senzorů a ultrazvukových senzorů vzdálenosti. S navrženým zařízením jsem provedl testy, kde se finální naměřená přesnost pohybovala mezi 95 % a 100 % v závislosti na tom, zda se jednalo o průchod jedné osoby, či více osob za sebou, kde druhá z možností měla přesnost nižší. V kapitole 11 jsem následně shrnul nedostatky navrženého detektoru a možnosti dalšího vývoje tohoto zařízení. Navržený detektor vyniká oproti ostatním technologiím zejména cenovou dostupností a skutečností, že se jedná o jednoduché zařízení, které uživatel pouze zapojí do zásuvky a provede prvotní nastavení. Po dalším vývoji a odstranění zmíněných nedostatků lze toto zařízení shledat jako vhodné pro reálné nasazení v systémech domácí automatizace.

Literatura

[1] 18 Most Popular IoT Devices In 2020. *Software Testing Help* [online]. 2020 [cit. 2020-05-08]. Dostupné z: https://www.softwaretestinghelp.com/iot-devices/

[2] The Internet of Things and its Benefit to U.S. Water Customers. In: *Kurita America* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://www.kuritaamerica.com/the-splash/the-internet-of-things-and-its-benefit-to-us-water-customers

[3] A QUICK HISTORY OF HOME AUTOMATION. *The Somfy Blog* [online]. 2018 [cit. 2020-05-08]. Dostupné z: https://www.somfy.com.au/Blog/Post/2018-03-07-a-quick-history-of-home-automation

[4] FREMANTLE, Paul. *A Reference Architecture For The Internet of Things* [online]. 2015 [cit. 2020-05-08]. Dostupné z: https://wso2.com/whitepapers/a-reference-architecture-for-the-internet-of-things

[5] DANGE, Simita a Madhumita CHATTERJEE. *IoT Botnet: The Largest Threat to the IoT Network* [online]. 2019 [cit. 2020-05-08]. DOI: 10.1007/978-981-15-0132-6. Dostupné z: https://www.researchgate.net/publication/337427726_IoT_Botnet_The_Largest_Threat_to_the_IoT_Network

[6] YOKOTANI, Tetsuya a Yuya SASAKI. *Comparison with HTTP and MQTT on required network resources for IoT* [online]. 2017 [cit. 2020-05-08]. DOI: 10.1109/ICCEREC.2016.7814989. Dostupné z: https://ieeexplore.ieee.org/document/7814989

[7] *Key Exchange using Elliptical Curve Cryptography* [online]. 2016 [cit. 2020-05-08]. Dostupné z: https://www.silabs.com/community/blog.entry.html/2016/05/27/iot_security_part7-UeMh

[8] KUMAR GOYAL, Tarun a Vineet SAHULA. *Lightweight security algorithm for low power IoT devices* [online]. IEEE, 2016 [cit. 2020-05-08]. DOI: 10.1109/ICACCI.2016.7732296. Dostupné z: https://ieeexplore.ieee.org/abstract/document/7732296

[9] ARORA, Mohit. How secure is AES against brute force attacks?. *EE Times* [online]. 2012 [cit. 2020-05-08]. Dostupné z: https://www.eetimes.com/how-secure-is-aes-against-brute-force-attacks/#

[10] NAZIRIDIS, Nick. *Comparing ECDSA vs RSA* [online]. 2018 [cit. 2020-05-08]. Dostupné z: https://www.ssl.com/article/comparing-ecdsa-vs-rsa/

[11] *Elliptic Curve Digital Signature Algorithm (ECDSA)* [online]. In: . b.r. [cit. 2020-05-08]. Dostupné z: https://asecuritysite.com/encryption/ecdsa

[12] SUGAWARA, Takeshi, Benjamin CYR, Sara RAMPAZZI, Daniel GENKIN a Kevin FU. *Light Commands: Laser-Based Audio Injection Attacks on Voice-Controllable Systems* [online]. 2019 [cit. 2020-05-08]. Dostupné z: https://lightcommands.com/

[13] ČAGALJ, Mario. *Security of Wireless Networks* [online]. In: . b.r. [cit. 2020-05-08]. Dostupné z: https://slideplayer.com/slide/6312328/

[14] NOVAK, Maxim. *Serialization Performance comparison (C#/.NET) – Formats & Frameworks (XML–DataContractSerializer & XmlSerializer, BinaryFormatter, JSON– Newtonsoft & ServiceStack.Text, Protobuf, MsgPack)* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://maxondev.com/serialization-performance-comparison-c-net-formats-frameworks-xmldatacontractserializer-xmlserializer-binaryformatter-json-newtonsoft-servicestack-text/

[15] MALÝ, Martin. Protokol MQTT: komunikační standard pro IoT. In: *Root.cz* [online]. 2016 [cit. 2020-05-08]. Dostupné z: https://www.root.cz/clanky/protokol-mqtt-komunikacni-standard-pro-iot/

[16] HANGGORO, Delphi, Lukman ROSYIDI a Riri FITRI SARI. *Design and Implementation of Multi-Protocol Gateway for Internet of Things* [online]. IEEE, 2019 [cit. 2020-05-08]. DOI: 10.1109/ICIAICT.2019.8784849. Dostupné z: https://ieeexplore.ieee.org/document/8784849

[17] *Zigbee2mqtt* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://github.com/koenkk/zigbee2mqtt

[18] *Zwave2Mqtt* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://github.com/OpenZWave/Zwave2Mqtt

[19] *Ble2mqtt* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://github.com/shmuelzon/ble2mqtt

[20] *Comparison of MQTT Brokers* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://tewarid.github.io/2019/03/21/comparison-of-mqtt-brokers.html

[21] *Creating highly available and ultra-scalable MQTT clusters* [online]. 2016 [cit. 2020-05-08]. Dostupné z: https://www.hivemq.com/blog/clustering-mqtt-introduction-benefits/

[22] GRIDELLI, Stefano. *How WiFi Connection Works* [online]. 2019 [cit. 2020-05-08]. Dostupné z: https://netbeez.net/blog/how-wifi-connection-works/

[23] ZigBee and WiFi Coexistence. In: *Metageek.com* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://www.metageek.com/training/resources/zigbee-wifi-coexistence.html

[24] IGARASHI, Yuichi, Matti HILTUNEN, Kaustubh JOSHI a Richard SCHLICHTING. *An Extensible Home Automation Architecture Based on Cloud Offloading* [online]. 2015 [cit. 2020-05-08]. DOI: 10.1109/NBiS.2015.32. Dostupné z: https://ieeexplore.ieee.org/document/7350618

[25] AKHERFI, Khadija, Micheal GERNDT a Hamid HARROUD. *Mobile cloud computing for computation offloading: Issues and challenges* [online]. 2016 [cit. 2020-05-08]. Dostupné z: https://doi.org/10.1016/j.aci.2016.11.002

[26] SATYANARAYANAN, Mahadev, Paramvir BAHL, Ramon CACERES a Nigel DAVIES. *The Case for VM-based Cloudlets in Mobile Computing* [online]. 2009 [cit. 2020-05-08]. Dostupné z: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/cloudlets09.pdf

[27] GUO, Kai, Mingcong YANG, Yongbing ZHANG a Yusheng JI. *An Efficient Dynamic Offloading Approach Based on Optimization Technique for Mobile Edge Computing* [online]. IEEE, 2018 [cit. 2020-05-08]. DOI: 10.1109/MobileCloud.2018.00013. Dostupné z: https://ieeexplore.ieee.org/document/8350435

[28] LYNGGAARD, Per. *Artificial intelligence and Internet of Things in a "smart home" context: A Distributed System Architecture* [online]. Copenhagen, Denmark, 2015 [cit. 2020-05-08].

Dostupné z: https://vbn.aau.dk/en/publications/artificial-intelligence-and-internet-of-things-in-a-smart-home-co. Ph.D. thesis. Department of Electronic Systems, Aalborg University.

[29] DUCKLIN, Paul. *Serious Security: How to store your users' passwords safely* [online]. 2013 [cit. 2020-05-08]. Dostupné z: https://nakedsecurity.sophos.com/2013/11/20/serious-security-how-to-store-your-users-passwords-safely/

[30] Best Smart Home App UI concept. In: *Uplabs.com* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://www.uplabs.com/posts/best-smart-home-app-ui-concept

[31] *Energy Efficient Window Attachments* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://www.energy.gov/energysaver/energy-efficient-window-attachments

[32] Smarwi - chytré otevírání oken. *Vektiva.com* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://vektiva.com/shop/smarwi

[33] MEYDBRAY, Jenya, Keith EMERY a Sarah KURTZ. Pyranometers and Reference Cells, What's the Difference?. *PV Magazine* [online]. 2012 [cit. 2020-05-08]. Dostupné z: https://www.nrel.gov/docs/fy12osti/54498.pdf

[34] *Tilt Smarterhome* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://www.tiltsmarthome.com/

[35] *Sleep and Health* [online]. 2008 [cit. 2020-05-08]. Dostupné z: http://healthysleep.med.harvard.edu/need-sleep/whats-in-it-for-you/health

[36] *Sleep and mental health* [online]. 2009 [cit. 2020-05-08]. Dostupné z: https://www.health.harvard.edu/newsletter_article/sleep-and-mental-health

[37] OKAMOTO-MIZUNO, Kazue a Koh MIZUNO. *Effects of thermal environment on sleep and circadian rhythm* [online]. 2012 [cit. 2020-05-08]. DOI: 10.1186/1880-6805-31-14. Dostupné z: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3427038/

[38] THE EUROWINTER GROUP. *Cold exposure and winter mortality from ischaemic heart disease, cerebrovascular disease, respiratory disease, and all causes in warm and cold regions of Europe* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://doi.org/10.1016/S0140-6736(96)12338-2

[39] ARUNDEL, A. V., E. M. STERLING, J. H. BIGGIN a T. D. STERLING. *Indirect health effects of relative humidity in indoor environments* [online]. 1986 [cit. 2020-05-08]. DOI: 10.1289/ehp.8665351. Dostupné z: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1474709/

[40] *The Effects of Humidity on the Human Body* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://www.achooallergy.com/blog/learning/the-effects-of-humidity-on-the-human-body/

[41] TOSINI, Gianluca, Ian FERGUSON a Kazuo TSUBOTA. *Effects of blue light on the circadian system and eye physiology* [online]. 2016 [cit. 2020-05-08]. Dostupné z: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4734149/

[42] C. GIMENEZ, Marina, Martijn HESSELS, Maan VAN DE WERKEN, Bonnie DE VRIES, Domien G. M. BEERSMA a Marijke C. M. GORDIJN. Effects of artificial dawn on subjective ratings of sleep inertia and dim light melatonin onset. *Chronobiology International* [online]. 2010, , 1219-1241 [cit. 2020-05-08]. DOI: 10.3109/07420528.2010.496912. Dostupné z: https://www.rug.nl/research/portal/files/6752157/2010ChronobiolIntGimenez.pdf

[43] F. BRENA, Ramon, Juan Pablo GARCÍA-VÁZQUEZ, Carlos E. GALVÁN-TEJADA, David MUñOZ-RODRIGUEZ, Cesar VARGAS-ROSALES a James FANGMEYER. *Evolution of Indoor Positioning Technologies: A Survey* [online]. 2017 [cit. 2020-05-08]. Dostupné z: https://doi.org/10.1155/2017/2630413

[44] *The Latest People Counting Technologies Could Save Businesses Trillions* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://www.terabee.com/the-latest-people-counting-technologies-could-save-businesses-trillions/

[45] Dům na klíč 75. In: *Domy-drevostavby-na-klic.cz* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://www.domy-drevostavby-na-klic.cz/katalog-projekty-rodinnych-domu-zdene-domy-drevostavby-na-klic/dum-na-klic-75-drevostavba-75-drevostavba-na-klic-75/

[46] Ultrasonic Ranging Module HC - SR04. In: *Electro Schematics* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://www.electroschematics.com/wp-content/uploads/2013/07/HCSR04-datasheet-version-1.pdf

[47] HC-SR04. In: *Banggood* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://www.banggood.com/cs/10Pcs-Geekcreit-Ultrasonic-Module-HC-SR04-Distance-Measuring-Ranging-Transducer-Sensor-DC5V-2-450cm-p-942912.html?cur_warehouse=CN

[48] HC-SR501 PIR MOTION DETECTOR. In: *MPJA.com* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://www.mpja.com/download/31227sc.pdf

[49] PIR modul HC-SR501. In: *GM Electronic* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://www.gme.cz/pir-modul-hc-sr501

[50] WeMos D1 Mini ESP8266 WiFi modul. In: *Laskarduino* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://www.laskarduino.cz/wemos-d1-mini-esp8266-wifi-modul/

[51] Home Assistant: Demo. *Home Assistant* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://demo.home-assistant.io/#/lovelace/0

[52] BISS0001: Micro Power PIR Motion Detector IC. In: *Hardware To Software* [online]. b.r. [cit. 2020-05-08]. Dostupné z: http://www.hw2sw.com/wp-content/uploads/2012/09/BISS0001.pdf

[53] PIR sensors: HC-SR501. In: *IoT Experiments* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://www.iot-experiments.com/pir-sensors-hc-sr501/

[54] HC-SR04 Ultrasonic Rangefinder. In: *University of Washington: Digital arts & Experimental Media* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://dxarts.washington.edu/wiki/hc-sr04-ultrasonic-rangefinder

[55] WiFiManager. *Github* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://github.com/tzapu/WiFiManager

[56] MDNSResolver. *Github* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://github.com/madpilot/mDNSResolver

[57] MQTT Discovery. *Home Assistant* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://www.home-assistant.io/docs/mqtt/discovery/

[58] ESP8266EX: Datasheet. *Espressif* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf

[59] Zdroj 230 AC-DC 5V 700mA 3.5 W. *Laskarduino* [online]. b.r. [cit. 2020-05-08]. Dostupné z: https://www.laskarduino.cz/zdroj-230-ac-dc-5v-700ma-3-5-w/

Příloha A

Zdrojový kód navrženého detektoru průchodu je obsažen v příloze. Dále je v příloze STL soubor s 3D návrhem použité krabičky. Jsou zde k dispozici i další fotografie zmíněného zařízení. GIT repozitář s přílohou a krátkým komentářem k přiloženým souborům je dostupný z https://github.com/FilipBelohlavek/Indoor_Positioning_System.