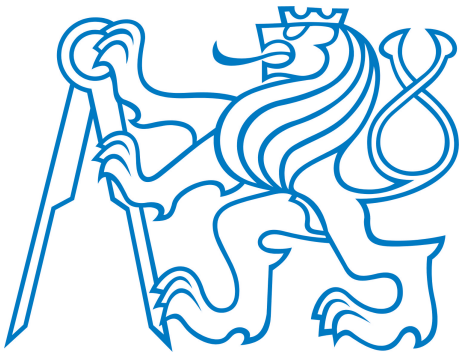
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta elektrotechnická

DIPLOMOVÁ PRÁCE



Bc. Petr Kovář

Hlasové ovládání inteligentní domácnosti

Katedra měření

Vedoucí diplomové práce: Ing. Jan Šedivý, CSc.
Studijní program: Kybernetika a robotika
Studijní obor: Senzory a přístrojová technika

Praha 2017

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta elektrotechnická
Katedra měření

Akademický rok **2016-17**

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student:	**Bc. Petr Kovář**
Studijní program:	**Kybernetika a robotika**
Obor:	**Senzory a přístrojová technika**
Název tématu česky:	**Hlasové ovládání inteligentní domácnosti**
Název tématu anglicky:	**Voice Control of Smart Home**

Pokyny pro vypracování:

1. Prostudujte architekturu a programové vybavení vhodné pro návrh řídicí jednotky inteligentní domácnosti a vyberte vhodné řešení.
2. Prostudujte bezdrátové komunikační technologie vhodné pro komunikaci se senzory a aktuátory a jednu vyberte.
3. Prostudujte metody komunikace mezi mobilním telefonem a řídicí jednotkou inteligentní domácnosti v prostředí internet protokolu verze 4 a 6.
4. Prostudujte možnosti ovládání senzorů a aktuátorů z mobilního telefonu, včetně hlasového ovládání, pro zvolenou řídicí jednotku inteligentní domácnosti.
5. Navrhněte a realizujte vzdálený přístup k řídicí jednotce inteligentní domácnosti z volného internetu i z lokální sítě v prostředí IPv4 a IPv6.
6. Navrhněte a implementujte hlasové ovládání inteligentní domácnosti se zvolenými technologiemi.
7. Navržené řešení otestujte. Zaměřte se jak na testování softwaru, tak na testy z pohledu uživatele.

Seznam odborné literatury:

[1] Goodwin, Steven. Smart Home Automation with Linux. Springer-Verlag New York, 2010. ISBN 978-1-4302-2779-3
[2] Z-Wave.Me Team. Z-Way Developers Documentation
[3] Allen, Grant. Android 4 Průvodce programováním mobilních aplikací. Computer Press Brno, 2013. ISBN 978-80-251-3782-6

Vedoucí diplomové práce:	Ing. Jan Šedivý, CSc. (K 13133)
Datum zadání diplomové práce:	5. ledna 2017
Platnost zadání do[1]:	30. září 2018

L.S.

Prof. Ing. Jan Holub, Ph.D.
vedoucí katedry

Prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 5. 1. 2017

[1] Platnost zadání je omezena na dobu tří následujících semestrů.

Poděkování

Děkuji mému vedoucímu diplomové práce panu Ing. Janu Šedivému, CSc. za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

V Praze dne ....
Podpis autora práce

Prohlášení autora práce

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 ....

Podpis autora práce

Abstrakt

Tato diplomová práce se zabývá návrhem a implementací hlasového ovládání vybrané inteligentní domácnosti prostřednictvím mobilní aplikace v OS Android. Na základě analýzy existujících bezdrátových komunikačních technologií je pro realizaci inteligentní domácnosti zvolena technologie Z-Wave. V práci je popsán proces přípravy řídicí jednotky, která se skládá z Raspberry Pi 3, modulu RaZberry a řídicího softwaru Z-Way. Představen je mechanismus pro automatické připojení mobilní aplikace k řídicí jednotce. Z hlediska komunikace je dále navrženo řešení pro zasílání dat z řídicí jednotky do mobilní aplikace pomocí PUSH notifikací. Klíčovým bodem diplomové práce je hlasové ovládání, pro které byla připravena kompletní architektura. Jedná se především o integraci existujících knihoven a služeb tak, aby výsledná aplikace dokázala porozumět přirozenému jazyku a vést s uživatelem krátké dialogy. V závěru práce je navržené řešení otestováno a dosažené výsledky zhodnoceny.

Klíčová slova

hlasové ovládání, inteligentní domácnost, Z-Wave, Z-Way, Raspberry Pi, RaZberry, Android, PUSH notifikace, rozpoznání řeči, záměr, entita, syntéza řeči, dialog

Abstract

This Master's thesis is focused on the designing and implementation of the voice control of a selected smart home technology through a mobile application on OS Android. Based on an analysis of available wireless communication technologies, Z-Wave technology is chosen for the realization of smart home. The thesis describes the process of preparing control unit, which consists of Raspberry Pi, RaZberry module and Z-Way control software. A mechanism for automatic connection establishment between the mobile application and the control unit is introduced. As for the communication, there is a solution proposed for sending data from the control unit to the mobile application via PUSH notifications. The key part of this thesis is voice control, for which the complete architecture was prepared. This is primarily about integration of existing technologies and services so that the proposed mobile application can understand natural language and manage short dialogs with a user. At the end of the thesis, the proposed solution is tested and the results are evaluated.

Keywords

voice control, smart home, Z-Wave, Z-Way, Raspberry Pi, RaZberry, Android, PUSH notifications, speech recognition, intent, entity, text to speech, dialogue

Obsah

1. Úvod

V posledních letech zažívá velký rozmach tzv. internet věcí (IoT), což jsou běžná zařízení připojená do internetu za účelem výměny, sběru a zpracování dat. Zatímco v roce 2015 bylo k internetu připojených 10 miliard zařízení, dle předpovědí vzroste tato hodnota do roku 2020 na 34 miliard zařízení. Velký potenciál internetu věcí podtrhují také očekávání analytiků, že zákazníci utratí během následujících 5 let až 6 bilionů dolarů za různá IoT řešení. [1]

Internet věcí mimo jiné zahrnuje také zařízení používaná v domácnostech. Již dnes si může zákazník pořídit inteligentní termostat s připojením k internetu a zapnout vytápění chaty dlouho před příjezdem. Pro vyšší bezpečnost zase poslouží IP kamera neustále sledující oblast před domem. Pokud je internet věcí používán správně, získá jeho uživatel nejen vyšší komfort, ale také bezpečnost a možnost úspor.

Společně s přibývajícím počtem zařízení připojených k internetu se zvyšuje poptávka po vhodném rozhraní, které umožní uživateli jednoduše a intuitivně ovládat a monitorovat tato zařízení. Objevují se zařízení s vestavěným webovým serverem, kde jako uživatelské rozhraní poslouží jakýkoli webový prohlížeč. Další možností je ovládací aplikace pro chytrý telefon, tablet, nebo počítač. Odlišný přístup zvolili společnosti Amazon a Google, které nabízejí hlasem ovládané inteligentní asistenty Amazon Echo a Google Home. Jedná se o samostatně stojící zařízení vybavená sadou mikrofonů a reproduktorů, kde veškerá komunikace s uživatelem probíhá pouze prostřednictvím hlasu.

My jsme se v rámci této diplomové práce rozhodli zkombinovat poslední dva uvedené přístupy, tj. mobilní aplikaci a hlasové ovládání. Cílem této práce je návrh a implementace hlasového ovládání vybrané inteligentní domácnosti skrze aplikaci v OS Android. Mobilní aplikaci, kterou vytvoříme v rámci této práce, pojmenujeme HomeVoice a pod tímto názvem se k ní budeme odkazovat v následujícím textu. Představené řešení má především poukázat na nezbytné kroky při vývoji podobných typů aplikací v OS Android. Jedná se o zajištění komunikace s inteligentní domácností (z lokální sítě i přes internet), ovládání/monitorování jednotlivých zařízení a realizaci hlasového ovládání v OS Android.

Práci zahájíme teoretickým rozborem problematiky inteligentních domácností, kde zadefinujeme pojmy jako senzor, aktuátor, řídicí jednotka a ovladač. V úvodu této práce se podíváme také na některé technologie (od protokolů po kompletní ekosystémy), které umožňují vzájemnou komunikaci zařízení v inteligentní domácnosti. Od počátku klademe důraz na bezdrátové komunikační technologie, ale pro srovnání uvedeme také pár drátových komunikačních technologií.

Na základě rozboru komunikačních technologií vybereme pro realizaci inteligentní domácnosti technologii Z-Wave. Jako řídicí jednotka poslouží mikropočítač Raspberry Pi 3 s operačním systémem Raspbian a řídicím softwarem Z-Way. Následuje příprava řídicí jednotky, tj. instalace OS Raspbian a řídicího softwaru Z-Way a také konfigurace SSH přístupu.

V další části prostudujeme možnosti programování vybrané řídicí jednotky, resp. řídicího softwaru Z-Way. Uvedeme jak API určená pro návrh vlastního uživatelského rozhraní (HTTP/JSON volání na vestavěný webserver), tak API pro vývoj rozšiřujících modulů řídicí jednotky (JavaScript). Jak se později ukáže, své využití najdou všechna uvedená API.

Aby mohl potenciální uživatel používat námi navrženou aplikaci HomeVoice bez nutnosti složité konfigurace, představíme návrh mechanismu pro automatické připojování k řídicí jednotce z lokální sítě i z internetu. Zároveň vyřešíme zasílání dat z řídicí jednotky do mobilního zařízení. Zatímco doposud bylo možné pro tento účel používat maximálně SMS zprávy a e-maily, naše řešení umožňuje přenášet data pomocí PUSH notifikací přímo do aplikace HomeVoice.

Jakmile vyřešíme komunikaci s řídicí jednotkou, podíváme se na možnosti použití hlasového ovládání v operačním systému Android. Ačkoli současné verze OS Android disponují hlasovým rozhraním (aplikace Google Now), neumožňují oficiálně integrovat ovládání inteligentní domácnosti. Z tohoto důvodu představíme vlastní řešení hlasového ovládání, které zahrnuje detekci klíčového slova (náhrada za "Ok Google"), rozpoznání řeči, porozumění přirozenému jazyku, zpracování kontextu a syntézu řeči. Cílem práce není podrobný rozbor a vlastní návrh algoritmů pro výše uvedené body, ale integrace existujících technologií (knihoven, služeb) do aplikace HomeVoice.

V závěru této diplomové práce otestujeme navržené řešení hlasového ovládání (aplikace HomeVoice) a zhodnotíme možnosti jeho praktického nasazení. Nejprve představíme několik různých Z-Wave zařízení, která nám umožní důkladné otestování aplikace HomeVoice. Zaměříme se jak na testování softwaru, tak na testy z pohledu uživatele. Dosažené výsledky shrneme a doplníme o poznatky z praktického užívání námi vybrané inteligentní domácnosti.

2. Inteligentní domácnost

2.1 Definice

Inteligentní domácnost (chytrá domácnost) je poměrně široký pojem, který nemá jednu obecnou definici. Existuje mnoho různých definic, z nichž některé jsou velmi podobné, zatímco jiné nemají mnoho společného. [2] Jak docházelo k vývoji technologií, podléhala i definice tohoto pojmu postupným změnám. Dříve byla za inteligentní domácnost označována taková domácnost, která měla telefonní připojku, či analogovou televizi. [3] Dnes mohou být domácnosti vybaveny digitální televizí nebo připojením k internetu, a přesto je nebudeme nazývat inteligentní.

Oxfordský slovník definuje inteligentní domácnost (smart home) jako: "Dům vybavený osvětlením, topením a elektronickými zařízeními, která mohou být ovládána vzdáleně pomocí chytrého telefonu, či počítače.". (překlad autora) [1]

Podle Smart Home Energy je inteligentní domácnost "dům zahrnující pokročilé automatizační systémy, které poskytují obyvatelům sofistikované monitorování a ovládání funkcí budovy. Inteligentní domácnost může například řídit osvětlení, teplotu, multimédia, zabezpečovací, okenní a dveřní operace, stejně jako mnoho dalších funkcí.". (překlad autora) [2]

SmartHomeUSA popisuje inteligentní domácnost následovně: "Inteligentní domácnost je taková, která nabízí svým majitelům komfort, bezpečnost, energetickou efektivnost (nízké provozní náklady) a pohodlí za všech okolností, bez ohledu na to, zda je někdo doma.". (překlad autora) [3]

Jak je patrné z výše uvedených definic, klíčovým znakem inteligentní domácnosti jsou zařízení, která oproti standardnímu vybavení domácnosti nabízí možnost monitorovat a řídit jednotlivé procesy. Jako typický příklad takového zařízení je možné uvést inteligentní žárovku ovládanou chytrým mobilním telefonem s Bluetooth. Ta kromě základního chování (zapnuto/vypnuto) může například plynule měnit svůj jas, či dokonce barvu. [6] Kromě manuálního monitorování a ovládání funkcí inteligentních zařízení uživatelem je vhodné zmínit také automatizaci domácnosti od jednoduchých aplikací s předem definovanou funkcí až po plně automatizované aplikace a sítě zařízení, která vzájemně komunikují a sdílí informace. [3] Jednoduchou aplikaci s předem definovanou funkcí představuje například pohybem spouštěné světlo. Mezi složitější aplikace může patřit síť zařízení, která na základě teploty v interiéru, exteriéru a požadavků

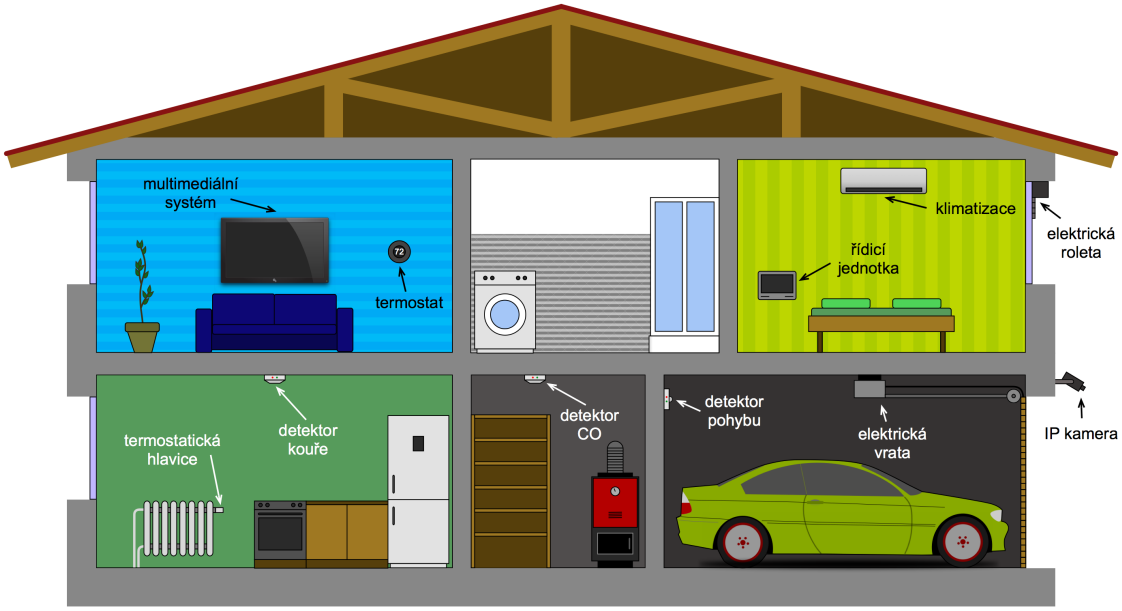
[1] Původní znění: "A home equipped with lighting, heating, and electronic devices that can be controlled remotely by smartphone or computer." [4]

[2] Původní znění: "A home that incorporates advanced automation systems to provide the inhabitants with sophisticated monitoring and control over the building's functions. For example a smart home may control lighting, temperature, multi-media, security, window and door operations, as well as may other functions." [**?**]

[3] Původní znění: "A Smart Home is one that provides its home owners comfort, security, energy efficiency (low operating costs) and convenience at all times, regardless of whether anyone is home." [5]

uživatele automaticky ovládá topení, klimatizaci, či dokonce otevírání oken za účelem větrání. S rozvojem internetu a především mobilních sítí je u inteligentní domácnosti kladen stále větší důraz na vzdálený přístup, ideálně odkudkoli ze světa. Pro mnoho lidí je jistě lákavá představa, že si budou moci ověřit zamčení hlavních dveří a aktivaci zabezpečovacího systému i po odjezdu od domu. Z tohoto důvodu mohou být dnešní inteligentní domácnosti vybaveny tzv. branou, která zajistí propojení sítě inteligentních zařízení s internetem. Důležitou roli při definování pojmu inteligentní domácnost hraje také důvod jejich vytváření. Motivací pro zavádění inteligentních domácností je nejen zvýšení pohodlí a bezpečí obyvatel domácnosti, ale také snaha o dosažení co nejnižších provozních nákladů (spotřeba vody, plynu, elektrické energie).

Možná podoba inteligentní domácnosti je zachycena na obrázku (4.2). V ukázkové domácnosti jsou představeny některé z dnes dostupných typů inteligentních zařízení.



Obrázek 2.1: Příklad podoby inteligentní domácnosti se zaměřením na použitá zařízení. Základ tvoří řídicí jednotka kombinovaná s ovládacím panelem. Ze zařízeních sloužících k zabezpečení domácnosti před nezvanými hosty se zde nachází IP kamera a detektor pohybu. Pro bezpečí obyvatel je dále instalován detektor kouře, detektor oxidu uhelnatého (CO) a detektor zaplavení. Ze zařízení typu HVAC (topení, větrání, klimatizace) se zde nachází inteligentní termostat, termostatická hlavice k topení a klimatizace. Dále je zobrazená inteligentní domácnost vybavena elektrickou okenní roletou a elektrickými garážovými vraty. Závěrem je ještě vhodné zmínit, že součástí inteligentní domácnosti mohou být také různé multimediální systémy. [2]

2.2 Inteligentní zařízení

2.2.1 Úvod

Inteligentní zařízení (anglicky Smart Device) představuje elektronické zařízení, které je vybaveno komunikačním rozhraním pro účely propojení, sdílení a interakce s dal-

šími inteligentními zařízeními, či uživatelem. [7] Zařízení tvořící inteligentní domácnost vzniká obvykle rozšířením funkcionality standardních zařízení použitých v domácnosti tak, aby bylo možné snadné monitorování a ovládání jejich funkcí uživatelem, ale také vzájemné propojení s dalšími inteligentními zařízeními. Mezi zařízení použitá v inteligentní domácnosti patří také řídicí jednotky a zařízení sloužící jako uživatelská rozhraní (ovládací panely, tablety,...). [2]

Dostupnost zařízení, která mají zabudované některé z dnes používaných komunikačních rozhraní přímo od výrobce, je na vzestupu. Kromě toho existují také zařízení, která se stanou inteligentními po doplnění příslušného adaptéru. Hlavní funkcí adaptéru je zapojení zařízení do infrastruktury inteligentní domácnosti. [8] Díky těmto adaptérům je dnes například možné proměnit domácí NAS server na řídicí jednotku pro inteligentní domácnost, nebo rozšířit možnosti obyčejných elektrických garážových vrat o dálkové ovládání odkudkoli ze světa.

Mezi základní typy zařízení používaných v inteligentní domácnosti patří senzory, aktuátory, brány, řídicí jednotky a ovladače. Existuje ale i celá řada komplexních zařízení, která v sobě kombinují dvě a více výše uvedených typů zařízení. [8]

2.2.2 Senzor

Senzor (někdy také nazývaný detektor) představuje zařízení, které snímá fyzikální, biologickou, či chemickou veličinu. Z důvodu usnadnění dalšího zpracování a přenosu této veličiny v rámci sítě inteligentních zařízení, dochází uvnitř senzoru k transformaci na výstupní, tzv. měronosnou veličinu (analogový, či digitální signál). V inteligentní domácnosti je pro zařízení zastávající funkci senzoru vhodné použít spíše označení inteligentní senzor, neboť tato komplexní zařízení obvykle splňují předpoklady inteligentního senzoru, tj. mají možnost obousměrné komunikace, obsahují číslicovou část (ADC) a nabízí diagnostiku a korekci chyb. [9] Senzory určené pro použití v inteligentní domácnosti mohou navíc obsahovat blok pro další zpracování naměřených dat a paměť pro jejich dočasné ukládání.

Pro inteligentní domácnost představují senzory prostředek, kterým získá informace o dění nejen uvnitř domu, ale i v jeho bezprostředním okolí. Naměřená data následně umožní uživatelům domácnosti sledovat průběhy jednotlivých veličin. Velký význam mají tyto informace také pro automatizaci některých procesů. Díky senzorům se dostane inteligentní domácnosti zpětné vazby na akce provedené pomocí aktuátorů. Je tedy zřejmé, že senzory jsou nezbytnou součástí inteligentní domácnosti.

Pokud se porozhlédneme po nabídce zařízení pro inteligentní domácnost, která zastávají funkci senzoru, zjistíme, že najdeme široké spektrum zařízení pro měření teploty, vzdušné vlhkosti, úrovně osvětlení a UV záření. Dále existují senzory pro monitorování spotřeby elektrické energie, vody a plynu (měření aktuální i celkové spotřeby). Pro zajištění bezpečí obyvatel inteligentní domácnosti slouží detektory kouře, pohybu, zaplavení, detektory koncentrace různých plynů v ovzduší, ale také dveřní a okenní senzory, které upozorní na otevření oken, či dveří. Pro úplnost je ještě vhodné zmínit kamerové systémy (často s využitím IP kamer), detektory hluku a meteorologické stanice.

2.2.3 Aktuátor

Dalším významným prvkem inteligentní domácnosti je tzv. aktuátor (v češtině označovaný také jako akční člen). Aktuátor umožňuje inteligentní domácnosti interagovat s okolním světem. Jedná se tedy o zařízení, prostřednictvím kterého může inteligentní domácnost konat akce v reálném světě a ovlivňovat tak své okolí. K tomu obvykle slouží různá mechanická, či elektronická zařízení. [8] Příkladem mechanických zařízení mohou být hydraulické a pneumatické pumpy nebo servomotory. Zástupce elektronických zařízení tvoří např. spínače osvětlení.

Podobně jako v případě senzorů, tak i zde najdeme na trhu mnoho zařízení, která plní funkci akčního členu. Mezi nejvíce zastoupená zařízení patří různé druhy vypínačů a stmívačů osvětlení, tj. zařízení, která umožní inteligentní spínání světel a v případě stmívačů také jejich plynulou regulaci. Elektrické zásuvky umožňují vypínat a zapínat připojená elektrická zařízení na dálku. Pro ovládání teploty v domácnosti slouží termostaty a ovladače klimatizací. V případě potřeby je dokonce možné vybavit každý radiátor v domácnosti termostatickou hlavicí a ovládat tak teplotu zcela nezávisle. Dále je vhodné zmínit dveřní zámky, které pomocí zabudovaných servomotorů zajistí automatické odemykání a zamykání dveří. Existuje také celá řada zařízení, která promění standardní vybavení domácnosti v inteligentní zařízení. Jedná se například o ovladače multimédií, které obvykle pomocí IR umožní inteligentní domácnosti ovládat standardní multimediální přístroje (televize, DVD přehrávač, HiFi systém,...). Dále sem patří ovladače žaluzií, rolet, markýz, či garážových vrat. Na závěr ještě můžeme zmínit dálkově ovladané ventily, které kromě uzávěru přívodu plynu najdou využití např. v zahradních zavlažovačích.

2.2.4 Řídicí jednotka

Častou, ne ovšem nezbytnou součástí inteligentní domácnosti je řídicí jednotka (někdy také označovaná jako HUB). Řídicí jednotka představuje počítač, který získává data ze senzorů a s využitím předem definovaného chování vygeneruje odpovídající reakci, kterou vykonají akční členy. Kromě automatického chování může řídicí jednotka nabízet také manuální ovládání akčních členů pomocí dálkového ovladače. Roli řídicí jednotky zastává většinou počítač s operačním systémem Linux, Windows a v některých případech i OS X. [8]

Řídicí jednotka nemusí sloužit pouze k propojení senzorů a akčních členů, ale také jako brána pro propojení inteligentní domácnosti s internetem. To je nutné zvláště v případech, kdy samotné senzory a akční členy nejsou uzpůsobené pro přímé připojení do volného internetu. Díky tomu bude možné inteligentní domácnost monitorovat a ovládat přes internet i mimo domov.

Řídicí jednotku je možné koupit ve formě zcela připraveného zařízení, které může prakticky ihned po vybalení z krabice začít zastávat svou funkci. Jedná se o zařízení vybavená příslušným komunikačním rozhraním, které umožňuje propojení se senzory a akčními členy v domácnosti. Součástí těchto řídicích jednotek bývá také předinstalovaný software zajišťující jak automatizaci a řídicí funkci, tak také ovládání jednotky prostřednictvím internetu. Druhou možností je pořízení adaptéru, který promění v řídicí

jednotku klasický počítač, notebook nebo třeba NAS server. Zde se od uživatele očekává určitá znalost počítačů, neboť je třeba instalovat příslušné ovladače a řídicí software. Výhodou tohoto řešení je jeho nižší cena (v případě použití již existujícího počítače) a také obvykle vyšší možnost konfigurace a úpravy zařízení.

2.2.5 Ovladač

Ovladač vystupuje jako rozhraní mezi uživatelem a inteligentní domácností, tj. umožňuje uživateli ovládat a ve většině případů také monitorovat funkce inteligentní domácnosti. V současnosti je převážná většina ovladačů konstruována s dotykovým displejem pro snadné a intuitivní ovládání. S rozvojem technologií pro rozpoznávání hlasu není výjimkou ani hlasové ovládání inteligentní domácnosti.

Kromě speciálních ovládacích zařízení, která jsou mnohdy součástí řídicí jednotky, roste v posledních letech trend ovládání inteligentní domácnosti pomocí chytrých telefonů a tabletů. Dnes prakticky neexistuje výrobce inteligentních domácnosti, který by nepřipravil ovládací aplikaci pro chytrý telefon. V kombinaci s mobilním internetem nabízí chytré telefony dříve nemyslitelné způsoby ovládání. [8]

2.3 Technologie

2.3.1 Úvod do komunikačních technologií

Ať už se jedná o senzory, aktuátory, řídicí jednotky, nebo speciální ovladače, vždy je třeba zajistit vzájemné propojení těchto zařízení. Existuje celá řada technologií, které se dnes používají pro zajištění komunikace mezi zařízeními inteligentní domácnosti. V závislosti na použité technologii se liší konstrukce komunikačního rozhraní. Některé technologie navíc vyžadují zavedení infrastruktury, bez které by byla komunikace nemožná.

Základní dělení vychází z použité technologie fyzické přenosové cesty. Fyzické přenosové cesty dělíme na metalické vedení, optické vlákno a bezdrátový přenos, který je možné dále dělit na rádiový a optický. Metalická přenosová cesta představuje obvykle dvojici vodičů ve formě koaxiálního kabelu, krouceného dvoudrátu, či jen vodičů položených paralelně vedle sebe. Optické vlákno využívá absolutního odrazu světla na rozhraní dvou prostředí s odlišným indexem lomu a oproti metalické přenosové cestě nabízí obtížný odposlech a vyšší přenosovou kapacitu. Základ optické bezdrátové přenosové cesty tvoří většinou infračervený přenos (nachází se např. v dálkových ovladačích multimédií). Rádiová bezdrátová přenosová cesta může využívat široké spektrum frekvencí elektromagnetického záření do cca 300 GHz. Od použité technologie se následně odvíjí vlastnosti a omezení. [10]

V inteligentních domácnostech se stále větší oblibě těší bezdrátový rádiový přenos, neboť na rozdíl od metalického vedení a optických vláken nevyžaduje konstrukci speciální infrastruktury. Díky bezdrátovým technologiím dnes existuje cesta, jak proměnit třeba i starší dům v inteligentní domácnost bez nutnosti bourání a větších úprav. Na druhou stranu metalická vedení a optická vlákna najdou své využití při stavbě inteligentní

domácnosti od základů, neboť jsou méně náchylná k rušení. Výhodou je také absence všudypřítomného elektromagnetického záření, což je pro mnoho lidí důležitý faktor při rozhodování.

2.3.2 Ethernet

Typickým příkladem technologie, která vyžaduje fyzické propojení dvojice zařízení je Ethernet. Standardizace probíhá na fyzické a spojové vrstvě ISO/OSI modelu[4]. Na fyzické vrstvě je Ethernet realizován kroucenými páry metalických vodičů (10Base-T, 10Base-TX, 100Base-T2, 100Base-T4, 1000Base-T), dále koaxiálním kabelem (10Base2, 10Base5), či optickými vlákny (10Base-F, 100Base-FX, 1000Base-SX, 1000Base-LX). [10] Přenosová rychlost leží v rozmězí 10 Mbps až 100 Gbps. V praxi se nejčastěji používá rychlost 100 Mbps. Vzhledem k tomu, že je Ethernet založený na drátovém propojení (metalické vedení, nebo optické kabely), je velmi odolný vnějšímu rušení a tedy i více spolehlivý. Jakmile je ovšem jednou vytvořena síť pomocí Ethernetu, jsou velmi složité její změny. [11] Společně se stoupající velikostí a složitostí sítě navíc dochází k rychlému růstu nákladů na zřízení potřebné infrastruktury (zvlášt v případě starších domů).

2.3.3 ITU-T G.hn

Dalším zástupcem drátové technologie je ITU-T G.9960 (zkráceně G.hn). Tvůrcem tohoto technologického standardu je Mezinárodní telekomunikační institut (ITU) a jedná se o poměrně nový standard (přijatý v roce 2009). Stejně jako v případě Ethernetu, tak i ITU-T G.hn definuje fyzickou a spojovou vrstvu ISO/OSI modelu. Cílem standardu G.hn je sjednocení komunikace elektronických zařízení přes nejrůznější média (elektrická síť, telefonní linka, koaxiální kabely). Domácí síť tvořená pomocí tohoto standardu se skládá z jednotlivých domén, kde každá doména představuje síť zařízení komunikujících pomocí určitého média (např. elektrická síť). O propojení domén využívajících různá média se starají tzv. mosty (bridge). [12] Na rozdíl od technologie Ethernet využívá technologie ITU-T G.hn především existující infrastrukturu.

2.3.4 PLC

PowerLine Communication (PLC) označuje technologie využívající k přenosu dat elektrickou síť. Technologie PLC vychází z úvahy, že většina zařízení v inteligentní domácnosti vyžaduje napájení z elektrické sítě, která tak zároveň může sloužit jako potřebná komunikační infrastruktura. Výhodou tohoto přístupu jsou nízké pořizovací náklady, neboť není třeba budovat kompletní infrastrukturu. Hlavní nevýhody spočívají ve skutečnosti, že elektrická síť není primárně určena k přenosu dat. Díky tomu dochází

[4]Referenční model ISO/OSI poskytuje obecný rámec pro návrh distribuovaných systémů za účelem jejich snadného propojení. Základem ISO/OSI modelu je sedmivrstvý protokolový zásobník, kde každá vrstva využívá služeb vrtsvy pod ní a naopak poskytuje služby vrstvě nad sebou. Nejnižší, fyzická vrstva řeší přenos jednotlivých bitů. Spojová vrstva přenáší bloky dat (rámce) mezi dvěma uzly. Síťová vrstva zajišťuje směrování paketů mezi uzly sítě. Úkolem transportní vrstvy je sjednotit chování vrstev pod ní. Relační vrstva zajišťuje sestavení, řízení a rušení relací. Prezentační vrstva má na starosti potřebné konverze přenesených dat. Nejvyšší, aplikační vrstva obsahuje standardizovaná jádra aplikací. [10]

k zarušení přenosu. Technologie PLC je také náchylná k případnému odposlechu komunikace. Existuje několik standardů definujících pravidla přenosu dat po elektrické síti. Za zmínku stojí například specifikace HomePlug AV a HomePlug GB, které jsou obě v souladu se standardem IEEE 1901. Tento standard zajišťuje vyvážené a efektivní využití elektrické sítě a definuje mechanismy pro vzájemné propojení zařízení tak, aby bylo dosaženo požadované šířky pásma a kvality služby. Mezi další standardy, které definují přenos dat prostřednictvím elektrické sítě, patří CEBus, LonWorks, či protokol X10 (kromě přenosu dat prostřednictvím elektrické sítě definuje i bezdrátový rádiový přenos). [11] [13]

2.3.5 WiFi

Poměrně populární technologií, která dnes nachází využití při stavbě inteligentních domácnosti, je WiFi. Jedná se o bezdrátovou komunikační technologii využívající k přenosu rádiový kanál. Původní specifikaci WiFi definuje standard IEEE 802.11. Postupnou úpravou a přidáváním dodatků se z tohoto standardu následně vyvinuly standardy IEEE 802.11a, 802.11b, 802.11g, 802.11h, 802.11n, 802.11ac a několik dalších. V závislosti na konkrétní specifikaci probíhá komunikace na frekvenci 2.4 GHz, případně 5 GHz v pásmu ISM[5] s přenosovou rychlostí od 1 Mbit/s až po 300 Mbit/s. Uvnitř budov činí dosah WiFi kolem 30 m, na volném prostranství je to až 90 m. Kromě standardních pásem na 2.4 GHz a 5 GHz se v USA můžeme setkat se standardem 802.11y, který pracuje ve frekvenčním pásmu 3650 - 3700 MHz. Výše zmíněné standardy definují požadavky na fyzickou vrstvu ISO/OSI modelu, ale také na podčást spojové vrstvy, tzv. MAC[6]. Řízení přístupu k médiu zajišťuje u WiFi přístupová metoda CSMA/CA[7]. [10]

Pomocí WiFi lze vytvářet dva druhy sítí a to stacionární sítě a tzv. Ad-hoc sítě. Stacionární síť je označení infrastruktury, která obvykle využívá stacionární přístupové body (AP). Jednotlivé uzly (zařízení) následně komunikují prostřednictvím přístupového bodu, který tato zařízení obsluhuje. Součástí přístupového bodu může být také brána pro propojení s jiným médiem (typicky Ethernet pro připojení do pevné sítě). Díky tomu mohou získat připojená zařízení přístup na volný internet. Druhou variantou jsou Ad-hoc sítě, které označují přímé propojení uzlů (P2P) bez nutnosti použít síťovou infrastrukturu. [10] Kromě stacionární sítě, která se již dnes nachází v mnoha domácnostech, najde v inteligentní domácnosti uplatnění i Ad-hoc síť pro přímé propojení dvojice inteligentních zařízení. Stále se ovšem nejedná o ideální řešení pro koncept inteligentní domácnosti, kde obvykle požadujeme, aby jednotlivé uzly byly přímo propojeny s více než jedním dalším uzlem sítě (tzv. smíšená topologie). S podporou smíšené topologie sítě naštěstí přichází standard IEEE 802.11s. [14]

[5] ISM (zkratka z anglického názvu Industrial, scientific and medical) jsou pásma radiové komunikace používaná v průmyslovém, vědeckém a zdravotnickém oboru.

[6] MAC (z anglického Media Access Control) označuje v teorii počítačových sítí tzv. řízení přístupu k médiu. Jedná se o podvrstvu spojové vrstvy ISO/OSI modelu, která zahrnuje mechanismy pro řízení přístupu ke sdílenému médiu za účelem emulace plně duplexní komunikace.

[7] CSMA/CA je mechanismus pro řízení přístupu k médiu, kdy každý uzel připojený k médiu před započetím komunikace naslouchá, zda je médium volné. Uzel následně zvolí náhodný časový interval a pokud je po uplynutí tohoto intervalu médium stále volné, může začít vysílat. V opačném případě se interval prodlužuje o dobu obsazenosti média. [10]

Jak vyplývá z výše uvedeného, mezi hlavní přednosti WiFi jako komunikační technologie pro inteligentní domácnost patří minimální potřeba infrastruktury. Pro zřízení inteligentní domácnosti s technologií WiFi postačí pořízení přístupového bodu s branou do pevné sítě a volného internetu. Již dnes je takový přístupový bod (WiFi modem/router) součástí většiny domácností, které mají přístup k internetu. Oproti dalším bezdrátovým technologiím použitým v inteligentních domácnostech nabízí WiFi větší dosah (lze navíc rozšířit pomocí běžně dostupných WiFi extenderů) a vyšší přenosovou rychlost. Ze seznamu výhod je vhodné jmenovat ještě spolehlivost a bezpečnost komunikace. Proti WiFi mluví cena a také vyšší spotřeba elektrické energie. Obzvlášť vyšší spotřeba je důležitý argument v případě nutnosti použít napájení z baterií. S cílem nabídnout nižší spotřebu elektrické energie vznikla tzv. Low Power WiFi (LP-WiFi), která však stále čeká na širší nasazení. Společně s rozšiřováním WiFi technologie se také objevují problémy se vzájemným rušením přenosů dat. [14] [15] Inteligentní zařízení využívající k přenosu dat WiFi nabízí například společnost Belkin v produktové řadě WeMo[8].

2.3.6 Bluetooth

Bluetooth je další dobře známou bezdrátovou komunikační technologií, která si v poslední době nachází cestu do inteligentních domácností. Technologie Bluetooth je definována standardem IEEE 802.15.1 a vytváří tzv. WPAN[9]. Motivem pro vznik této technologie bylo nahrazení komunikace přes USB a sériovou linku bezdrátovou alternativou. Podobně jako v případě WiFi, tak i Bluetooth pracuje v ISM pásmu 2.4 GHz a v závislosti na konkrétní verzi nabízí přenosovou rychlost v rozmezí od 1.2 Mbit/s až po 24 Mbit/s ve verzi 4.0. Podle vysílacího výkonu se Bluetooth zařízení dále dělí do 3 tříd: class 1 (do 100 mW, dosah až 100 m), class 2 (do 2.4 mW, dosah do 10 m) a class 3 (do 1 mW, dosah do 1 m). Na rozdíl od WiFi se jedná o komplexní protokolový zásobník, který definuje chování od fyzické až po aplikační vrstvu (ne však zcela v souladu s ISO/OSI modelem). Typickou aplikací Bluetooth je vytváření adhoc pikosítí, kde jedno ze zařízení působí v roli tzv. mastera, který může obsluhovat až 7 podřízených zařízení (slave). Kromě toho existuje také podpora propojení jednotlivých pikosítí, čímž vznikne tzv. scatternet ("rozprostřená"síť). Připojení podřízeného zařízení k zařízení master musí předcházet proces párování, díky kterému je možné povolit připojení jenom vybraným zařízením. Zároveň dojde k uložení tohoto párování a při příštím pokusu o připojení již není třeba párovací proces opakovat. [10]

Na rozdíl od WiFi nenabízí Bluetooth zařízení podporu pro přímé připojení volného internetu. K tomu je třeba pořídit odpovídající bránu, což oproti WiFi zvyšuje nároky na použitou infrastrukturu. V porovnání s drátovými komunikačními technologiemi se ale stále jedná o poměrně nízké náklady. Nasazení Bluetooth v inteligentní domácnosti provází podobné nevýhody a omezení jako technologii WiFi. Vzhledem k tomu, že obě tyto technologie pracují v ISM pásmu na 2.4 GHz, není výjimkou jejich vzájemné rušení. Dalším společným problémem je vyšší spotřeba elektrické energie, která

[8]Dostupné například z: http://www.belkin.com/us/Products/home-automation/c/wemo-home-automation/

[9]WPAN (z anglického Wireless Personal Area Network) označuje bezdrátovou osobní síť, která je obvykle tvořena komunikujícími zařízeními (mobil, PDA, notebook) poblíž jejich vlastníka.

znemožňuje dlouhodobý provoz Bluetooth zařízení na baterie. Řešení na tento problém naštěstí přináší Bluetooth verze 4.0, který poskytuje nízkoenergetický přenos, ale také menší datový tok. Oproti WiFi nabízí Bluetooth obvykle také kratší dosah signálu a nižší úroveň zabezpečení. Výhodou je jednoduchost přidání zařízení do inteligentní domácnosti, neboť stačí projít párovací proces mezi Bluetooth zařízením a chytrým telefonem a následně už jen pomocí příslušné aplikace ovládat a monitorovat funkce tohoto zařízení. Příkladem inteligentního zařízení, které používá technologii Bluetooth, je výše zmiňovaná chytrá žárovka MiPow Playbulb[10], či senzory a aktuátory vyráběné společností Elgato[11].

2.3.7 ZigBee

Velmi rozšířená alternativa k technologiím WiFi a Bluetooth se nazývá ZigBee a jedná se o nadstavbu nad standardem IEEE 802.15.4. Stejně jako v případě Bluetooth i ZigBee je určena pro vytváření bezdrátových osobních sítí (WPAN). Samotný standard IEEE 802.15.4 definuje fyzickou vrstvu a podvrstvu MAC z vrstvy spojové. Komunikace standardně probíhá v ISM pásmu na 2.4 GHz s přenosovou rychlostí do 250 kbit/s. V USA je možné se setkat s ZigBee v pásmu ISM 915 MHz a přenosovou rychlostí do 40 kbit/s. V Evropě se jako varianta na nižší frekvencí nabízí komunikace v pásmu 868 MHz s přenosovou rychlostí do 20 kbit/s. ZigBee nabízí poměrně nízký vysílací výkon (pouze 1 mW) a dosah signálu v rozmezí 10 až 30 metrů. Podvrstva MAC dle standardu IEEE 802.15.4 rozlišuje 3 typy zařízení a to koordinátora (udržuje informaci o podobě celé sítě), FFD[12] (může komunikovat s jakýmkoli zařízením v síti a převzít funkci koordinátora) a RFD[13] (typicky koncové zařízení, může komunikovat pouze s FFD). Nadstavba ZigBee definuje chování na síťové a aplikační vrstvě. Směrování paketů je řešeno ad-hoc až při vyslání požavku a teprve tehdy se hledá cesta k cílovému uzlu. Na aplikační vrstvě se pracuje se ZigBee Device objekty a uživatelem definovanými aplikačními objekty. [10]

V inteligentních domácnostech se technologie ZigBee těší velké oblibě. Na rozdíl od výše zmíněných technologií WiFi a Bluetooth je ZigBee navržena tak, aby umožňovala dlouhodobý provoz na baterie. Daň za tuto možnost představuje výrazně nižší přenosová rychlost, než v případě WiFi a Bluetooth. Nejedná se ale o nijak vážný nedostatek, neboť aplikace v inteligentní domácnosti obvykle potřebují přenášet pouze malé bloky dat. Dosah je nižší než v případě WiFi, ale díky možnosti vytvářet různé topologie sítí je možné s dostatečným počtem zařízení zajistit kvalitní pokrytí celé domácnosti. Pro připojení do pevné sítě, resp. volného internetu je opět potřeba příslušná brána. Díky bezdrátové přenosové cestě představuje brána jediné zařízení potřebné pro vytvoření infrastruktury. Velkou výhodou je také nízká cena jednotlivých ZigBee zařízení. Té může výrobce dosáhnout díky přijatelné ceně za samotný ZigBee čipset, která při velkém odběru klesá až na hranici 1 USD. [16] Technologii ZigBee využívá pro svou funkci například známá inteligentní žárovka Philips Hue[14] od stejnojmenného výrobce.

[10]Dostupné například z: http://www.playbulb.com/en/playbulb-smart.html
[11]Dostupné například z: https://www.elgato.com/en/eve/
[12]FFD - zkratka z anglického Full Function Device
[13]RFD - zkratka z anglického Reduced Function Device
[14]Dostupné například z: http://www.philips.cz/c-m-li/osobni-bezdratove-osvetleni-hue

2.3.8 RFID

Technologie RFID (Radio Frequency Identification), která původně vznikla jako náhrada čárových kódů, nachází postupně cestu i do inteligentních domácností. Klasický RFID systém se skládá ze dvou typů zařízení. Prvním typem zařízení je tzv. TAG (RFID prvek), který může být buď aktivní (napájený z baterie), pasivní (energie získána vstřebáním VF pole čtečky), či semi-aktivní (vlastní baterie pouze pro napájení paměti). Druhým typem zařízení je čtečka, která jak název napovídá vyčítá informaci z jednotlivých TAGů. Součástí TAGu může být ID (8 - 16 bajtů), paměť (ROM, RW), případně i senzory. Pro přenos dat se nejčastěji používají bezlicenční pásma na nízkých frekvencích LF (125 - 134 kHz), vysokých frekvencích HF (13.56 MHz), velmi vysokých frekvencích VHF (860 - 930 MHz) a mikrovlnných frekvencích MW (2.4 GHz a 5.8 GHz). V závilosti na použité frekvenci se pohybuje dosah v rozmezí od cca 80 cm až po desítky metrů. [10]

Na rozdíl od výše uvedených bezdrátových technologií nachází RFID uplatnění v inteligentní domácnosti především při identifikaci a sledování osob, či objektů. RFID TAG s požadovanými informacemi (např. jedinečný identifikátor osoby, či objektu) se umístí na sledovaný objekt, případně se jím vybaví osoba, kterou je třeba identifikovat. RFID čtečky umístěné v domácnosti se následně postarají o výčet dat z přítomných RFID TAGů. Díky tomu je možné odemknout dveře pouze autorizovaným osobám, sledovat přítomnost jednotlivých osob v domácnosti, či například kontrolovat obsah inteligentní lednice.

2.3.9 EnOcean

Zajímavou bezdrátovou technologií je EnOcean, pro kterou jsou typické senzory bez potřeby napájení z baterií nebo sítě. Existence takových zařízení je možná díky velmi nízké spotřebě elektrické energie a také díky procesu získávání energie z okolí (tzv. energy harvesting). Energie pro bezdrátovou komunikaci je získána například z pohybu, elektromagnetického záření, nebo rozdílů teplot. Vzhledem k malému odběru v režimu spánku (< 100 nA) a přenosu zprávy o velikosti pouhých 8 bajtů (1 B data, 7 B režie) a době trvání 1 ms dosahuje EnOcean velmi nízké spotřeby energie. I přes nízkou spotřebu nabízí technologie EnOcean dosah až 30 metrů uvnitř budov a 300 metrů ve volném prostoru. Pro přenos dat se používá pásmo 868 MHz (Evropa), 315 MHz (Asie), 902 MHz (USA, Kanada) a 928 MHz (Japonsko). Přenosová rychlost činí 125 kbit/s. Součástí sítě EnOcean senzorů je obvykle také brána sloužící jako rozhraní pro další bezdrátové technologie např. ZigBee, nebo Bluetooth LE. [17]

Velkou výhodou technologie EnOcean je snadná instalace a minimální potřeba údržby. Díky absenci drátového přípojení (pro přenos i napájení) a baterií není potřeba složitá infrastruktura ani dodatečná manipulace (výměna baterií). Senzor tak může být například vestavěn přímo do konstrukce budovy. Vzhledem k velmi krátkému vysílacímu času a typicky dlouhé periodě přenosu dat vychází pravděpodobnost kolize nízko (0.01% pro 100 senzorů vysílajích data jednou za minutu). Jedná se tedy o poměrně spolehlivý způsob přenosu. Seznam zařízení je možné nalézt na strákách EnOcean Alliance[15].

[15]Dostupné z: https://www.enocean-alliance.org/en/products/

2.3.10 Z-Wave

Vedoucí pozici mezi bezdrátovými technologiemi pro inteligentní domácnosti si drží technologie Z-Wave. Jedná se o kompletní řešení zahrnující vše od fyzické až po aplikační vrstvu. Certifikaci Z-Wave aktuálně nese přes 1700 zařízení od více než 450 výrobců, což dle odhadů představuje 60 až 70 procent trhu. Z-Wave poskytuje spolehlivé přenosy malých balíků dat s nízkou latencí a rychlostí přenosu až 100 kbit/s. Propustnost 40 kbit/s (9.6 kbit/s u starších čipů) je vhodná pro většinu řídicích a senzorových aplikací. Protokolový zásobník Z-Wave se skládá z 5 vrstev (fyzická, MAC, síťová, transportní a aplikační vrstva). Přenos probíhá v pásmu 868.42 MHz v Evropě a 908.42 MHz v USA a Kanadě. Z-Wave může využívat také další pásma v závislosti na regulacích dané země. Maximální vzdálenost mezi dvěmi zařízeními činí okolo 30 metrů. Počet uzlů v síti je omezen na 232, ale může být dále navýšen propojením více sítí.

Z-Wave rozlišuje dvě základní třídy zařízení a to řídicí zařízení (Controller) a podřízené zařízení (Slave). Každé zařízení obsahuje identifikátor sítě (Home ID) a unikátní indentifikátor (Node ID), který toto zařízení jednoznačně určuje v dané síti. Vzájemně komunikovat mohou pouze zařízení se stejným identifikátorem sítě (Home ID). Řídicí zařízení mají identifikátor sítě naprogramovaný od výroby a uživatel ho nemůže změnit. Z-Wave síť může obsahovat více řídicích zařízení, ale pouze jedno z nich může být primární a určuje identifikátor sítě. Podřízená zařízení přebírají identifikátor sítě od primárního řídicího zařízení. Proces přidání zařízení do sítě se nazývá inkluze (z anglického Inclusion) a přidávané zařízení musí přijmout přirazené Home ID a Node ID. Odebrání zařízení ze sítě Z-Wave se označuje exkluze (z anglického Exclusion) a odebírané zařízení je uvedeno do továrního nastavení včetně smazání Home ID a Node ID. [18]

Dalším znakem Z-Wave sítě je vytváření smíšené síťové topologie (Mesh). Řídicí uzel nevyžaduje přímé spojení s každým podřízeným uzlem v síti, ale s cílovým uzlem může komunikovat prostřednictvím uzlů ležících mezi nimi. Často je tak možné zajistit spolehlivou komunikaci i pro zařízení, která neleží v přímém dosahu řídicího uzlu. Během procesu inkluze zasílá přidávaný uzel řídicímu zařízení seznam všech uzlů, které má v přímém dosahu. Později může uzel zaslat tento seznam na vyžádání. Řídicí uzel ze získaných informací vytváří směrovací tabulku, která popisuje použitelné cesty pro komunikaci mezi řídicím a podřízeným uzlem. Při přenosu dat zkouší řídicí uzel nejprve přímé spojení. Pokud přímé spojení neuspěje, použije řídicí uzel postupně až tři různé cesty získané ze směrovací tabulky. Jestliže ani potom nedostane odpověď od cílového uzlu, řídicí zařízení oznámí neúspěch. [18]

Vzhledem k použitému pásmu pod 1 GHz nedochází k zarušení Z-Wave komunikace signály s frekvencí 2,4 GHz a 5 GHz. To představuje velkou výhodu hlavně v centrech velkých měst, kde jsou tato pásma obsazena všudypřítomnými sítěmi WiFi. Na rozdíl od konkurenční technologie ZigBee je Z-Wave navrženo jako kompletní řešení. Díky tomu je zaručena vzájemná kompatibilita všech Z-Wave zařízení včetně těch, která budou vyrobena v budoucnosti. Příkladem Z-Wave zařízení je Aeotec MultiSensor 6[16], který umožňuje měřit teplotu, vlhkost, osvětlení, UV záření, pohyb a vibrace.

[16]Dostupné například z: http://aeotec.com/z-wave-sensor

3. Výběr řídicí jednotky

3.1 Požadavky

Základ inteligentní domácnosti tvoří obvykle řídicí jednotka. Existují inteligentní domácnosti i bez řídicí jednotky, ale ve většině případů se jedná o jednoduchá řešení, která zvládají pouze základní úlohy (např. ovládání světel dálkovým ovládáním). Inteligentní domácnost nabízející pokročilejší funkce bude téměr vždy vyžadovat patřičnou řídicí jednotku. Pokud chceme zajistit vzdálený přístup k ovládání inteligentní domácnosti, poslouží řídicí jednotka s vestavěnou branou do internetu.

Při výběru řídicí jednotky hraje důležitou roli technologie použitá pro vzájemnou komunikaci mezi řídicí jednotkou, senzory a aktuátory. Pro každou z výše uvedených komunikačních technologií existuje na trhu určitá skupina kompatibilních zařízení. Cílem této práce je implementovat hlasové ovládání pro co nejširší spektrum funkcí inteligentní domácnosti. Musíme tedy zvolit takovou technologii, která nabízí dostatečnou rozmanitost kompatibilních zařízení. Kromě rozmanitosti je důležitý také celkový podíl skupiny vzájemně komatibilních zařízení dané technologie na trhu. Čím vyšší tento podíl bude, tím větší je šance uvedení navrženého řešení do praxe a tím větší bude také dopad této práce. Budeme tedy hledat takovou řídicí jednotku, která je kompatibilní s co největší skupinou zařízení používajících danou technologii.

Dále požadujeme, aby se jednalo o hotové řešení bez nutnosti vývoje vlastního softwaru řídicí jednotky. Mezi cíle práce nepatří návrh a implementace řídicího softwaru, neboť tato úloha je natolik komplexní, že by vyžadovala realizaci v rámci samostatného projektu. Nezavrhujeme ovšem případnou instalaci řídicího softwaru, či vývoj rozšiřujících modulů. Tato možnost je naopak vítaná z hlediska možného budoucího rozšíření funkcí řídicí jednotky. Abychom obecně mohli řídicí jednotku ovládat pomocí námi vytvořené aplikace, musí řídicí software disponovat vhodným aplikačním rozhraním (tzv. API[1]). Pro daný typ úlohy bude nejvhodnější webové aplikační rozhraní známé pod pojmem REST[2].

Předpokladem pro použití REST je webový server s REST API vestavěný v řídicí jednotce. Z toho plyne další požadavek na řídicí jednotku, kterým je rozhraní pro připojení do sítě (internetu). Při výběru řídicí jednotky se tedy zaměříme na ta zařízení, která disponují WiFi konektivitou, či rozhraním ethernet.

[1] Zkratka z anglického výrazu Application Programming Interface.

[2] REST, zkratka z anglického výrazu Representational State Transfer představuje architekturu rozhraní, která umožňuje jednotný a snadný přístup ke zdrojům (data, stavy aplikace). REST API bývá často používáno v souvislosti s webovými službami, kde se k jednotlivým webovým zdrojům přistupuje prostřednictvím HTTP protokolu. Mezi standardní metody HTTP prokolu patří GET, POST, PUT, OPTIONS a DELETE. Každá z těchto metod určuje, jaká akce má být provedena nad vybranými zdroji. Jednotlivé zdroje jsou při použití REST API rozlišeny jedinečným identifikátorem zvaným URI (např. /light/command/on). V případě webových služeb se URI použije jako přípona základní URL, která definuje použitý protokol a adresu webového serveru. Kompletní URL pak může vypadat následovně: http://api.example.com/light/command/on.

3.2 Volba komunikační technologie

Základní kritérium při výběru řídicí jednotky představuje volba komunikační technologie. Vybírat budeme na základě požadavků stanovených v úvodu této kapitoly z výše popsaných bezdrátových komunikačních technologií.

Mezi nejrozšířenejší z uvažovaných bezdrátových komunikačních technologiích patří Bluetooth a WiFi. S těmito technologiemi se setkáváme každý den ať už v chytrých telefonech, či počítačích. Od roku 2000 bylo certifikováno více než 30 tisíc WiFi zařízení. Certifikaci Bluetooth získalo dokonce více než 30 tisíc výrobců. [19] Pouze malá část z tohoto počtu ovšem připadá na zařízení určená pro inteligentní domácnost. Větší rozšíření těchto technologií v inteligentních domácnostech má pomoci Bluetooth LE a LP-WiFi.

Konkurenční technologii ZigBee aktuálně využívá 400 výrobců, kteří nabízí celkem 1200 certifikovaných zařízení. [20] Z-Wave podporuje více než 450 výrobců na celém světě a dohromady je k dispozici 1700 certifikovaných vzájemně kompatibilních zařízení. Celkový počet prodaných zařízení Z-Wave pak činí 50 milionů kusů. [21] Z-Wave navíc garantuje zpětnou i budoucí kompatibilitu všech Z-Wave certifikovaných zařízení, zatímco u ZigBee nemusí být zařízení různých výrobců navzájem kompatibilní. Pokud chceme použít technologii, která aktuálně nabízí největší skupinu vzájemně kompatibilních zařízení, Z-Wave je ta správná volba.

Další výhody plynou ze specifikace Z-Wave, která pokrývá vše od fyzické až po aplikační vrstvu. Aplikační vrstva pak poskytuje aplikacím rozhraní pro správu sítě Z-Wave zařízení, která zahrnuje přidávání/odebírání jednotlivých Z-Wave zařízení a úpravu směrovacích tabulek. Prostřednictvím tohoto rozhraní je možné také ovládat a konfigurovat jednotlivá zařízení. Z pohledu integrace Z-Wave technologie do řídicí jednotky tak není třeba řešit žádné dodatečné vrstvy protokolového zásobníku, ale pouze řídicí software.

Na základě argumentů uvedených výše jsme se rozhodli použít pro realizaci řídicí jednotky inteligentní domácnosti technologii Z-Wave.

3.3 Volba řídicího softwaru

3.3.1 Funkce řídicího softwaru

Jak již bylo řečeno v kapitole 3.2, Z-Wave specifikuje kompletní protokolový zásobník až po aplikační vrstvu. V praxi to znamená, že se jednotlivé řídicí jednotky liší hlavně v použitém řídicím softwaru. Tento software využívá služeb aplikační vrstvy Z-Wave protokolu a implementuje řídicí funkci inteligentní domácnosti. Řídicí software se stará o získávání dat ze senzorů a o ovládání aktuátorů. Kromě této základní funkce může řídicí software nabízet vestavěný webový server. S využitím klasického webového prohlížeče tak může uživatel nastavovat, ovládat a monitorovat inteligentní domácnost jak z lokální sítě, tak případně i přes internet odkudkoli ze světa. Další pokročilou funkcí, kterou obvykle implementuje řídicí software, je domácí automatizace. Pod tímto pojmem se skrývá proces, kdy řídicí jednotka na základě dat ze senzorů

a předem definovaného chování generuje patřičné reakce prostřednictvím dostupných aktuátorů. Pro Z-Wave je dnes k dispozici několik řídicích softwarů.

3.3.2 Vera UI7

Řídicím softwarem Vera UI7 jsou vybaveny řídicí jednotky společnosti Vera. Tento software zajišťuje kompletní ovládání a monitorování inteligentní domácnosti skládající se ze Z-Wave zařízení. Kromě toho umožňuje reagovat na různé druhy událostí a automaticky měnit scény na základě denní doby, západu/východu slunce, či stavu jednotlivých zařízení. Uživatel může řídicí jednotku s UI7 ovládat přes vestavěný webový server, nebo s využitím aplikace pro Android a iOS. [22]

Vera UI7 je založena na enginu Luup, který kombinuje skriptovací jazyk Lua a průmyslový standard pro ovládání zařízení UPnP[3]. Vyvíjené aplikace (např. pro chytré telefony) mohou s UI7 komunikovat prostřednictvím standardního UPnP protokolu (skládá se z SOAP/XML žádostí). Pro případ, že vývojář nemůže použít UPnP, je UI7 vybaven i REST API s přístupem přes HTTP volání. Jazyk Lua je možné použít také na straně řídicí jednotky pro psaní automatizačních skriptů. Mezi další přednosti UI7 patří Javascript API, které slouží pro vývoj doplňků. API je velmi podrobně zdokumentováno a přichází ve formě knihovny vystavující potřebné objekty a metody/funkce jádra řídicího softwaru. Vzdálený přístup odkudkoli ze světa zajišťuje zdarma společnost Vera přes server cp.mios.com. [23] [24]

Pro použití řídícího softwaru Vera UI7 mluví velký počet možností, jak programovat dodatečné moduly i aplikace přistupující k objektům a funkcím řídicí jednotky. Nevýhodou je uzavřenost tohoto řešení pouze na řídicí jednotky společnosti Vera.

3.3.3 OpenHAB2

OpenHAB2 je open-source řídicí software určený pro integraci různých technologií inteligentních domácností do jednoho řešení. Kromě technologie Z-Wave může OpenHAB2 ovládat také zařízení kompatibilní s EnOcean, Insteon, Samsung, LG, Tesla a mnoho dalších. Základ softwaru OpenHAB2 tvoří Eclipse SmartHome[4] a kompletní řídicí software je napsán v Javě. Díky tomu je možné OpenHAB2 spustit na Windows, Mac OS X i Linuxu. Jedná se o vysoce modulární software, což znamená, že může být rozšířen o celou řadu doplňků. Prostřednictvím doplňků je možné přidat podporu dalších typů zařízení, či například nové uživatelské rozhraní. [26]

Základní uživatelské rozhraní představuje vestavěný webový server, který umožňuje ovládat, monitorovat a konfigurovat připojená zařízení. Dále existují aplikace pro Android, iOS a Windows. Vývojářům je k dispozici REST API, kdy mohou prostřed-

[3]UPnP, zkratka z anglického výrazu Universal Plug and Play, je architektura umožňující spojení dvou síťových zařízení ve stylu klient-klient. Zařízení se po připojení do sítě nakonfiguruje a získá IP adresu. V dalším kroku hledá ostatní zařízení na síti pomocí UPnP Discovery protokolu (založen na HTTP a UDP). Jakmile najde jiné zařízení, musí získat jeho popis. Ten dostane ve formátu XML přes URL z předchozího kroku. Následná komunikace mezi ropoznanými zařízeními probíhá přes URL získaná v popisu zařízení s využitím SOAP/XML. [25]

[4]Více informací k Eclipse SmartHome najdete na https://eclipse.org/smarthome/

nictvím HTTP volání monitorovat, ovládat a konfigurovat zařízení připojená k řídicí jednotce. Vývoj doplňků probíhá v Javě a programování doplňků (včetně instalace vývojového prostředí Eclipse) je velmi dobře popsáno. [26]

Mezi velké výhody softwaru OpenHAB patří množství podporovaných technologií a zařízení. Kromě standardních zařízení inteligentní domácnosti tak může OpenHAB ovládat také domácí audio systém, či třeba komunikovat s elektromobilem Tesla. Za vším stojí velká komunita programátorů, která rozšiřuje tento open-source projekt o další možnosti a funkce. K dispozici jsou také zdrojové kódy pro Android, iOS i Windows aplikaci, což usnadní případnou integraci hlasového ovládání. Jedná se tedy o vhodného kandidáta na použitý řídicí software.

3.3.4 Freedomotic

Další open-source řídicí software se nazývá Freedomotic a jedná se o flexibilní a bezpečné řešení pro realizaci inteligentních systémů pro domácnosti, školy, muzea, kanceláře, atd.. Pomocí širokého množství doplňků je možné ovládat zařízení různých technologií, včetně vlastních zařízení založených na platformě Arduino. Stejným způsobem můžeme integrovat grafická uživatelská rozhraní třetích stran nebo třeba napojení na různé sociální sítě. Vývoj Freedomotic probíhá v Javě, což zaručuje kompatibilitu softwaru s Windows, Linux i Mac OS X. Projekt se aktuálně nachází v pokročilé beta fázi vývoje a datum oficiálního vydání zatím není známo. [27]

Freedomotic je možné nainstalovat s grafickým uživatelským rozhraním, kdy pro ovládání, monitorování a konfiguraci inteligentní domácnosti slouží samostatná aplikace. Tento režim vyžaduje připojení monitoru, klávesnice a myši přímo do řídicí jednotky. Druhou možnost představuje instalace s webovým serverem, který je přístupný prostřednictvím standardního webového prohlížeče. Pro vývoj doplňků slouží Java a vše potřebné je podrobně zdokumentováno. Pokud doplněk projde schvalovacím procesem, stane se součástí standardní verze softwaru Freedomotic. Kromě REST API je pro vývojáře k dispozici i websocket. [28]

Ve prospěch softwaru Freedomotic mluví množství doplňků, které je sice stále nižší, než u softwaru OpenHAB, ale i tak nabízí dostatek možností. Příjemná je podpora vývoje vlastních zařízení založených na platformě Arduino. Mezi zdrojovými kódy nechybí jednoduchá klientská aplikace pro Android. Oproti softwarue OpenHAB však působí Freedomotic méně propracovaně a finální integrace bude vyžadovat více času.

3.3.5 OpenRemote

Open-source platforma OpenRemote se skládá z řídicí serverové aplikace (Controller), klientské aplikace pro iOS/Android (Console) a online služby pro návrh GUI klientských aplikací (Designer). OpenRemote podporuje široké množství technologií pro inteligentní domácností. Kromě Z-Wave nalezneme podporu audio systému SONOS, inteligentních žárovek Phillips HUE, či technologií Insteon a EnOcean. OpenRemote je opět vyvíjen v Javě, což znamená podporu Windows, Mac OS X i Linux. [29]

Základem platformy OpenRemote je serverová aplikace, která se instaluje na řídicí jednotku. Tato aplikace propojuje zařízení inteligentní domácnosti a umožňuje jejich ovládání a monitorování z mobilních klientských aplikací. Pro vývojáře je k dispozici skriptovací jazyk Drool, pomocí kterého mohou rozšiřovat možnosti řídicí jednotky. Další možností je standardní REST API. [29]

Pro uživatele je jistě výhodou snadné vytváření grafického rozhraní klientských aplikací. Z pohledu vývoje hlasového ovládání to ale příliš velký význam nemá, neboť tato úloha bude vyžadovat zásah do kódu klientské aplikace, nebo kompletní přípravu vlastní aplikace. V neprospěch mluví slabší dokumentace hlavně v souvislosti s REST API.

3.3.6 Z-Way

Software Z-Way byl první řídicí software, který v roce 2011 získal certifikát od konsorcia Z-Wave Alliance. V roce 2014 prošel recertifikací s ohledem na pravidla nové verze protokolu Z-Wave Plus. Software je aktuálně otestovaný prakticky se všemi Z-Wave zařízeními dostupnými v Evropě a s většinou zařízení pocházejících z USA. Oproti výše uvedeným možnostem se nejedná o open-source řešení, ale pro zprovoznění je potřeba hardware (Z-Wave vysílač) s platným licenčním klíčem od společnosti Z-Wave.Me. Software Z-Way je aktuálně dostupný pro operační systémy Linux, Windows a Mac OS X. Co se týče hardwaru, na výběr je mezi modulem RaZberry a zařízením UZB stick. Modul RaZberry je určený pro jednodeskový počítač Raspberry Pi s operačním systémem Raspbian. UZB stick je standardní USB zařízení, které promění v řídicí jednotku počítač se systémem Windows, Linux nebo Mac OS X. S hardwarem jiných výrobců nebude platforma Z-Way spolupracovat. [30]

Platforma Z-Way se skládá z následujích částí:

1. Optimalizovaný firmware pro Z-Wave vysílací čip
2. Komunikační zásobník spravující Z-Wave síť
3. Automatizační engine
4. Webový server implementující uživatelské rozhraní (volitelný)

Platforma Z-Way nabízí bezplatnou službu pro vzdálený přístup prostřednictvím stránky find.z-wave.me. Takto je možné využívat funkce řídicí jednotky odkudkoli ze světa. Dále potěší možnost zálohování a následné obnovy inteligentní domácnosti. Pro ladění je možné využít vizualizaci a debug veškeré komunikace mezi jednotlivými uzly. [30]

Základem softwaru Z-Way je Z-Wave core. Toto jádro komunikuje se Z-Wave kompatibilními vysílači prostřednictvím standardního Sigma Designs Serial API. Navíc přidává několik dalších funkcí, např. změnu frekvence. Pro programátory je k dispozici několik různých API, která jsou navzájem částečně propojená. Prostřednictvím těchto rozhraní je možné přímo, či nepřímo komunikovat s jádrem Z-Wave core a používat jeho funkce. Psaní doplňkových modulů je možné prostřednictvím Javascript API a C Library API. Nechybí ani REST API pro napojení vlastních aplikací. K dispozici jsou navíc zdrojové kódy aplikací pro Android a iOS. [31]

Nevýhodou platformy Z-Way je omezení pouze na jejich hardware, jehož nabídka navíc není příliš široká. To je ovšem vykompenzováno optimalizovaným firmware pro Z-Wave vysílací čip. Díky tomu, že se jedná o komerční řešení, je zde předpoklad lepší podpory ze strany vývojářů a snazší instalace potřebného softwaru. Na základě výše uvedeného jsme se rozhodli použít řídicí software Z-Way.

3.4 Volba hardwaru

3.4.1 Z-Way kompatibilní hardware

Technologii (Z-Wave) i řídicí software (Z-Way) již máme vybraný. Dalším krokem je tedy logicky volba hardwaru, na kterém poběží řídicí software. Rozhodování je to relativně jednoduché, neboť s řídicím softwarem Z-Way jsou kompatibilní pouze dvě zařízení a to UZB (na obrázku (3.1)) a RaZberry (obrázek (3.2)). Zatímco UZB pro svou práci vyžaduje standardní počítač se systémem Windows, Linux, či Mac OS X, RaZberry je modul určený pro jednodeskový počítač Raspberry Pi.

Obrázek 3.1: UZB [32]

Obrázek 3.2: RaZberry [33]

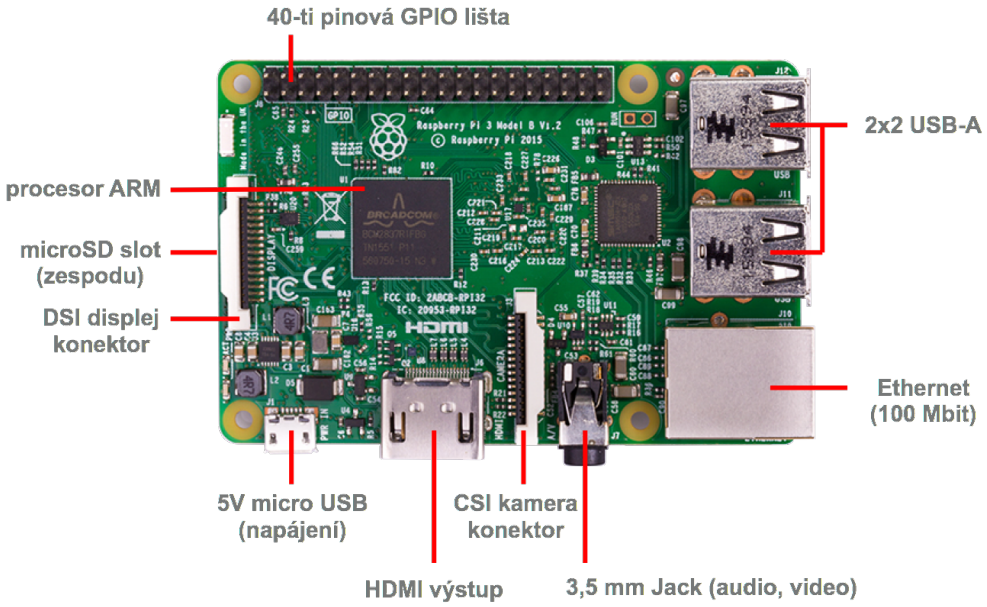
Zařízení UZB je ideální volba, pokud má uživatel doma například server, který běží neustále. Řídicí jednotku může samozřejmě tvořit i standardní počítač, což ale vzhledem k vysoké spotřebě elektrické energie nebude zcela ekonomické řešení. V situaci, kdy uživatel nemá k dispozici potřebný počítač, jsou náklady na jeho pořízení výrazně vyšší, než na pořízení Raspberry Pi. Počítač zamýšlený čistě jako řídicí jednotka bude navíc silně předimenzovaný. Zařízení UZB je možné v České republice pořídit za zhruba 1000 Kč[5].

Modul Razberry je sice dražší (zhruba 1600 Kč[6]), ale jeho cenu vykompenzuje výrazně nižší cena Raspberry Pi oproti standardnímu počítači. Vzhledem k tomu, že nemáme k dispozici vhodný počítač ani server, použití zařízení UZB se nejeví jako vhodná volba. Rozhodli jsme se tedy sestavit řídicí jednotku z minipočítače Raspberry Pi a modulu RaZberry.

[5]Dostupné z https://smarterhome.sk/cs/zwave-pc-adaptery/z-waveme-usb-stick-165.html
[6]Dostupné z https://smarterhome.sk/cs/zwave-pc-adaptery/razberry-2-300.html

3.4.2 Raspberry Pi

Raspberry Pi představuje jednodeskový počítač velikosti kreditní karty s cenou okolo 1 200 Kč[7]. V současnosti je možné koupit již třetí generaci nazvanou Raspberry Pi 3. Srdcem počítače Raspberry Pi 3 je čtyřjádrový procesor ARM taktovaný na frekvenci 1,2 GHz, což znamená dostatek výkonu i pro náročnější úlohy. Procesor doplňuje paměť RAM o kapacitě 1 GB. Součástí počítačů Raspberry Pi není paměť pro uživatelská a systémová data. Za účelem ukládání dat je Raspberry Pi vybaveno rozhraním pro připojení paměťové karty typu microSD. Co se týče konektivity, najdeme na Raspberry Pi 3 čtyři USB porty, HDMI port, 3,5 mm Jack (audio, video) a ethernet (100 Mbit). Poslední verze Raspberry Pi 3 navíc přinesla vestavěný modul WiFi a také podporu Bluetooth 4.1 (včetně Bluetooth LE). To vše při zachování spotřeby pod 2 A.



Obrázek 3.3: Vzhled jednodeskového počítače Raspberry Pi 3 Model B s důrazem na dostupná rozhraní. Díky výstupu HDMI a USB-A portům, je možné připojit monitor, klávesnici a myš a používat Raspberry Pi jako standardní počítač. Toho můžeme s výhodou využít při prvním spuštění a konfiguraci. 40-ti pinová GPIO lišta slouží Raspberry Pi pro interakci z vnějším světem. K této liště můžeme připojit tlačítka, LED diody, ale i výrazně složitější moduly. [35]

Mimo výše zmíněná standardní rozhraní nabízí Raspberry Pi 3 také 40 pinů, z nichž 26 je označovaných jako GPIO[8]. Jednotlivé GPIO piny mohou být nakonfigurovány jako vstupní, výstupní, či jako některá z komunikačních sběrnic (I2C, SPI nebo UART). Každý pin, který je nakonfigurovaný jako vstupní, může být navíc zdrojem přerušení pro ARM procesor. Zbývající piny zpřístupňují napájení (3,3 V a 5 V), zem a ID EEPROM. Raspberry Pi dále umožňuje připojení kamerového modulu[9] prostřednictvím rozhraní CSI a dotykového LCD displeje[10] do DSI konektoru. Jak je vidět, z hle-

[7]Dostupné z https://eshop.minidroid.cz/vsechno-pro-raspberry-pi/470-raspberry-pi-3-model-b.html

[8]Zkratka z anglického výrazu General Purpose Input/Output, který označuje konfigurovatelné vstupně/výstupní piny s obecným účelem.

[9]Dostupné z https://www.raspberrypi.org/documentation/hardware/camera/README.md

[10]Dostupné z https://www.raspberrypi.org/documentation/hardware/display/README.md

diska vstupně/výstupních portů se Raspberry Pi poměrně dost liší od standardních počítačů a nabízí tak dostatek prostoru pro případné experimenty. [34]

Co se týče operačních systémů, nadace Raspberry Pi oficiálně podporuje operační systém Raspbian. Jedná se o systém založený na operačním systému Debian a optimalizovaný pro hardware Raspberry Pi. Raspbian je stejně jako Debian open-source a tedy volně dostupný. Kromě samotného systému nabízí Raspbian přes 35 tisíc doplňujících balíků (Python, Java, Scratch, Mathematica, atd.). Vzhledem k oblibě počítače Raspberry Pi není divu, že existuje podpora i pro mnoho dalších operačních systémů. Mezi operační systémy třetích stran patří Ubuntu, Windows IoT Core, RISC OS, či například OSMC a OpenELEC určený pro mediální centra. I když ze strany Googlu zatím nepřišla oficiální podpora systému Android na Raspberry Pi, není problém nalézt mnoho neoficiální portů. [36]

4. Příprava řídicí jednotky

4.1 Úvod

V předchozí kapitole jsme vybrali hardware a software potřebný pro stavbu řídicí jednotky. Základ řídicí jednotky bude tvořit minipočítač Raspberry Pi s operačním systémem Raspbian. Kompatibilitu s technologií Z-Wave zajistí modul RaZberry. Pro řízení, vzdálený přístup a automatizaci použijeme software Z-Way, který je dodávaný zdarma při koupi modulu RaZberry. V rámci této kapitoly popíšeme postup přípravy řídicí jednotky podle návodů na webu raspberrypi.org a razberry.z-wave.me.

4.2 Instalace OS Raspbian

Prvním krokem při přípravě řídicí jednotky je instalace operačního systému Raspbian na micro SD kartu. Při instalaci se neobejdeme bez dalšího počítače, který je potřebný pro naformátování SD karty, stažení OS a jeho přenos na kartu. Velikost systému Raspbian (s prostředím PIXEL) přesahuje 4 GB, takže pro jeho instalaci je vhodná karta minimálně s kapacitou 8 GB. Čím vyšší rychlost zápisu a čtení bude SD karta mít, tím svižněji instalovaný systém poběží a tím lépe se nám s ním bude pracovat.

Raspberry Pi aktuálně podporuje pouze souborové systémy typu FAT. To znamená, že před instalací je potřeba naformátovat SD kartu na formát FAT16, případně FAT32 (nutné pro micro SD karty s kapacitou větší než 32 GB). V závislosti na operačním systému pomocného počítače bude potřeba využít specializovaný nástroj (např. SD Formatter[1] pro Windows a Mac OS X). [37]

Nejsnazší způsob instalace operačního systému Raspbian představuje použití nástroje NOOBS[2]. Tento instalátor operačního systému Raspbian stáhneme jako archiv ve formátu ZIP ze stránek nadace Raspberry Pi. Následně extrahujeme archivovaná data a přesuneme je na připravenou micro SD kartu. Kartu vložíme do Raspberry Pi, připojíme klávesnici, myš a monitor a zapneme napájení.

Posledním krokem je samotná instalace operačního systému. Procesem instalace nás provede grafické rozhraní instalátoru NOOBS. Před instalací některého z výše zmíněných alternativních systémů je třeba nejprve nastavit WiFi připojení, aby se mohla data stáhnout. Protože operační systém Rapbian je již obsažený v základní verzi instalátoru NOOBS, stačí nám pouze vybrat tento systém z menu a spustit instalaci. Doba instalace závisí na rychlosti použité karty. Ve většině případů se jedná o několik minut, po kterých se Raspberry Pi restartuje a je možné ho začít používat.

[1] Dostupné z https://www.sdcard.org/downloads/formatter_4/

[2] Zkratka z anglického výrazu New Out Of the Box Software. Jedná se o instalátor operačních systémů, který v základní verzi obsahuje Raspbian. Mimo Raspbianu je možné prostřenictvím menu vybrat instalaci několika dalších alternativních systémů (Pidora, LibreELEC, OSMC, RISC OS, Arch Linux a Windows 10 IoT), které se před zahájením instalace nejprve musí stáhnout z internetu. Existuje i verze NOOBS Lite, která na rozdíl od základní verze neobsahuje Raspbian. NOOBS je zdarma ke stažení zde: https://www.raspberrypi.org/downloads/noobs/.

4.3 Nastavení OS Raspbian (SSH)

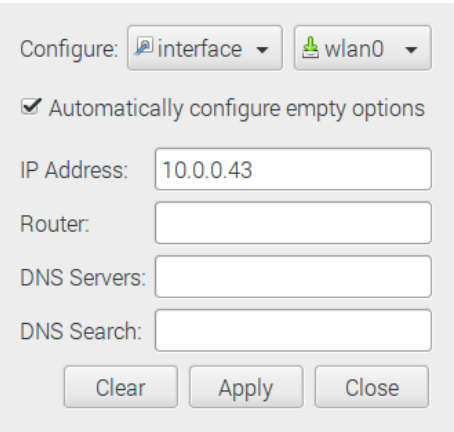
4.3.1 Motivace

Raspberry Pi s nainstalovaným systémem Raspbian se po připojení klávesnice, myši a monitoru příliš neliší od klasického stolního počítače. Veškerá nastavení, instalace softwaru a jeho používání tak můžeme provádět přímo v grafickém uživatelském rozhraní OS Raspbian. V případě řídicí jednotky inteligentní domácnosti není tento přístup zcela ideální. Ne vždy totiž máme možnost připojit k řídicí jednotce všechny potřebné periferie. Jednou z možností, jak ovládat Raspberry Pi z jiného počítače na stejné lokální síti, je protokol SSH[3].

4.3.2 Statická IP adresa

První krok, který zjednoduší přístup k Raspberry Pi z lokální a později i z veřejné sítě, je nastavení statické IP adresy. Standardně je IP adresa dynamicky přiřazována jednotlivým zařízením v lokální síti routerem, resp. vestavěným DHCP serverem. IP adresa je obvykle zapůjčována jednotlivým zařízením pouze na určitou tzv. zápůjční dobu. Proměnná IP adresa komplikuje vzálený přístup, neboť musíme neustále aktualizovat nastavení pro vzdálený přístup. Proto na úvod nastavíme statickou IP adresu. Při volbě statické adresy je třeba držet se v rozsahu IP adres lokální sítě. Doporučuje se nastavit statickou IP adresu podle aktuálně přidělené dynamické adresy.

Nastavení statické IP adresy je dnes možné přes grafické rozhraní zvané PIXEL. V našem případě je Raspberry Pi 3 připojeno do lokální sítě přes WiFi, takže statickou IP adresu nastavíme pro rozhraní wlan0.



Obrázek 4.1: Okno nastavení bezdrátových a drátových sítí v systému Raspbian. Pro nastavení statické IP adresy WiFi připojení zvolíme rozhraní wlan0. Do pole IP Address zapíšeme požadovanou IP adresu. Zbylé položky ponecháme prázdné (nastaví se automaticky).

[3]SSH (Secure Shell) označuje zabezpečený komunikační protokol používaný v obecně nezabezpečených počítačových sítích. Jedná se o protokol transportní vrstvy, který je typicky vystavěný na protokolu TCP/IP. Připojení prostřednictvím SSH funguje na principu klient-server. Při odesílání dat přes SSH dojde k jejich zašifrování. Jakmile data dorazí ke svému cíli, dojde k jejich dešifrování. Nejznámější aplikací je pravděpodobně zabezpečené přihlášení na vzdálený počítač a přístup do jeho příkazové řádky. [38]

4.3.3 SSH přístup

Přístup přes SSH je v operačním systému Raspbian v základním stavu vypnutý. Při prvotním nastavení se tedy nevyhneme připojení klávesnice, myši a monitoru. Následně již bude možné všechny periferie odpojit a přistupovat k Raspberry Pi vzdáleně z jiných počítačů na stejné lokální síti. Základní konfigurační nástroj v systému Raspbian představuje raspi-config. Tento nástroj můžeme vyvolat z terminálu pomocí příkazu `sudo raspi-config`. Otevře se okno nastavení, které umožňuje konfigurovat chování systému. Nastavení SSH se nachází pod položkou rozšířené nastavení (Advanced Settings).

Před dalším použitím je dobré ještě z bezpečnostních důvodů změnit defaultní přihlašovací údaje k Raspberry Pi (defaultní jméno: pi, heslo: raspberry). Tyto přihlašovací údaje slouží také k příhlašení přes SSH a nechceme, aby řídicí jednotku ovládl někdo jiný.

Vše je připraveno, abychom se mohli připojit k Raspberry Pi přes SSH z lokální sítě. Pokud se budeme připojovat ze zařízení s operačním systémem Linux, či Mac OS X, můžeme použít vestavěného klienta v příkazové řádce. Na počítači s Windows poslouží například SSH klient PuTTY. Pro přihlášení z příkazové řádky použijeme příkaz: `ssh pi@10.0.0.43`, kde pi je zvolené uživatelské jméno a za znakem @ následuje IP adresa zařízení, kterou jsme nastavili v předchozím kroku. Následně budeme požádání o vyplnění hesla. Jakmile tak učiníme, objeví se před námi příkazová řádka Raspberry Pi.

4.3.4 SSH přístup bez hesla

Doposud jsme při přihlášení přes SSH museli pokaždé zadat zvolené heslo. Aby po nás nebylo při každém pokusu o přihlášení požadováno heslo, nastavíme si bezheslový přístup vygenerováním páru veřejný/privátní SSH klíč.

Klíče se generují na zařízení (počítači), se kterým se plánujeme vzdáleně přihlašovat k Raspberry Pi. Vygenerované SSH klíče se ukládají do složky `~/.ssh`. Výpis existujících klíčů získáme příkazem: `ls ~/.ssh`. Pokud se ve složce již nachází pár klíčů (id_rsa.pub, id_dsa.pub), můžeme použít tyto existující klíče a přeskočit proces generování, nebo je smazat a vytvořit nové.

Pro vygenerování klíčů slouží příkaz: `ssh-keygen -t rsa -C raspi3`. Parametr raspi3 je pouze komentář popisující daný pár klíčů a je tedy na každém, jakou hodnotu zde nastaví. Defaultní adresář pro uložení klíčů potvrdíme stiskem Enter. Passphrase ponecháme prázdnou a opět stiskneme Enter. Tím máme klíče vygenerované. Pokud si nyní vypíšeme obsah adresáře `~/.ssh`, najdeme v něm složku id_rsa obsahující privátní klíč našeho zařízení (počítače) a soubor id_rsa.pub s veřejným SSH klíčem.

Nyní je třeba zkopírovat veřejný SSH klíč (id_rsa.pub) z počítače na Raspberry Pi. Pomocí SSH se přihlásíme k Raspberry Pi a vytvoříme adresář `~/.ssh` (pokud již neexituje) příkazem: `install -d -m 700 ~/.ssh`, který vytvoří adresář a nastaví mu potřebná práva.

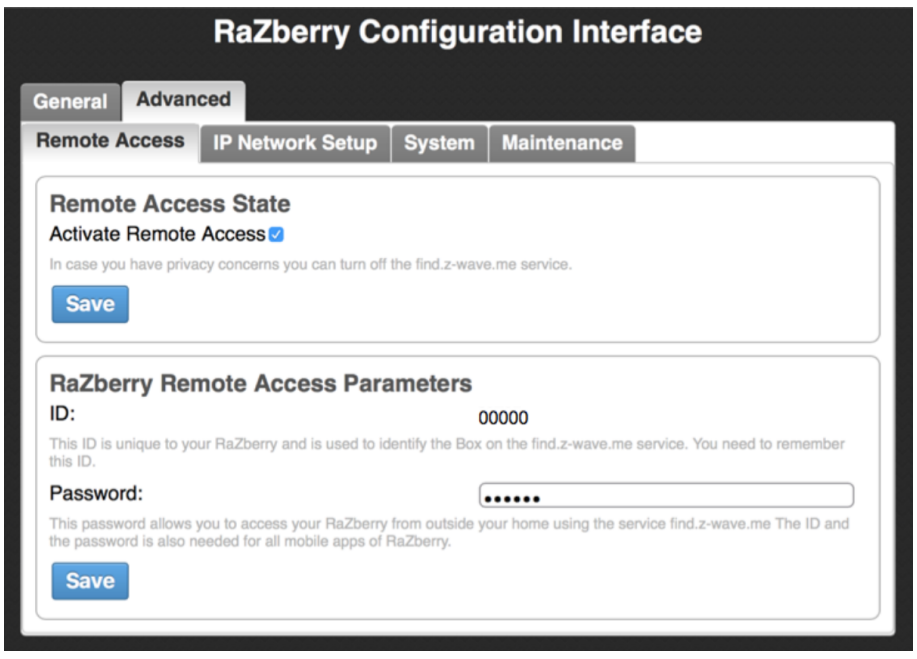
Můžeme zkopírovat veřejný klíč z počítače na Raspberry Pi. K tomu použijeme příkaz: `cat ~/.ssh/id_rsa.pub | ssh pi@10.0.0.43 'cat >> .ssh/authorized_keys'`. Tentokrát po nás ještě bude Raspberry požadovat ověření heslem. Poté se již můžeme k řídicí jednotce (Raspberry Pi) přihlašovat vzdáleně přes SSH bez nutnosti zadávání hesla.

4.4 Instalace softwaru Z-Way

Posledním krokem při přípravě řídicí jednotky je instalace řídicího softwaru Z-Way. Výrobce se snažil postup instalace co nejvíce zjednodušit a proto připravil instalační skript, který se postará o instalaci a konfiguraci řídicího softwaru Z-Way. Pro stažení skriptu a instalaci softwaru musí být Raspberry Pi připojeno k internetu. Skript stáhneme a spustíme příkazem `wget -q -O - http://razberry.z-wave.me/install | sudo bash`. Po dokončení instalace a konfigurace můžeme k řídicímu softwaru přistupovat prostřednictvím vestavěného webového serveru z lokální sítě.

4.5 Nastavení softwaru Z-Way

Abychom mohli k funkcím řídicí jednotky přistupovat také z internetu, musíme povolit vzdálený přístup, zjistit si identifikátor řídicí jednotky a nastavit heslo. Základní konfiguraci softwaru Z-Way je možné provést pomocí služby běžící na portu 8084. IP adresu řídicí jednotky můžeme zjistit z příkazové řádky pomocí příkazu `ifconfig`, či pomocí jiného počítače a služby find.z-wave.me.

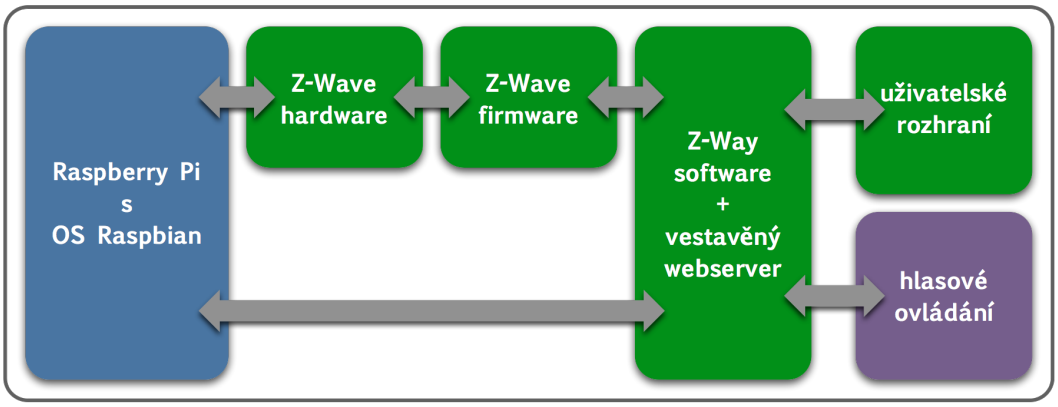


Obrázek 4.2: Okno nastavení softwaru Z-Way na portu 8084. Zde je možné zapnout vzdálený přístup k řídicí jednotce (přes internet) a nastavit přístupové heslo. Jedinečný identifikátor řídicí jednotky je vygenerován automaticky při instalaci softwaru. Tento identifikátor je nezbytný pro autorizaci při vzdáleném přístupu. V nastavení je dále možné provádět upgrade firmwaru, či například povolit vzdálenou podporu.

5. Ovládání řídicí jednotky

5.1 Struktura řídicí jednotky

V předchozích kapitolách jsme vybrali softwarové a hardwarové vybavení vhodné pro návrh řídicí jednotky. Základ řídicí jednotky tvoří počítač Raspberry Pi s operačním systémem Raspbian. Kompatibilitu s technologií Z-Wave zajišťuje modul RaZberry od společnosti Z-Wave.Me, který se připojuje na GPIO piny Raspberry Pi. RaZberry obsahuje certifikovaný Sigma Designs Z-Wave vysílač/přijímač s upraveným firmwarem. Použitý řídicí software také pochází od společnosti Z-Wave.Me a nese označení Z-Way. Kompletní struktura námi připravené řídicí jednotky je představena na obrázku (5.2).



Obrázek 5.1: Struktura řídicí jednotky založené na počítači Raspberry Pi s operačním systémem Raspbian (vyznačeno modře). Hardwarové a softwarové vybavení od společnosti Z-Wave.Me je vyznačeno zeleně a skládá se z modulu RaZberry (hardware a firmware Z-Wave), řídicího softwaru Z-Way s vestavěným webserverem a uživatelského rozhraní (webová aplikace). Hlasové ovládání (vyznačeno fialově) tvoří další typ uživatelského rozhraní a v celkovém řešení tak zastává podobnou funkci jako zmíněná webová aplikace. Podle [45]

Jak je patrné z obrázku (5.2), řídicí jednotka obsahuje všechno potřebné vybavení a je tak připravena k použití koncovým uživatelem. Ten může ovládat a monitorovat svou inteligentní domácnost prostřednictvím webové aplikace s grafickým uživatelským rozhraním. Pro pokročilé uživatele je navíc k dispozici rozhraní Expert UI, které mimo jiné umožňuje konfiguraci Z-Wave sítě. Námi připravená řídicí jednotka tedy představuje hotové řešení, které je samo o sobě zcela funkční. Uvažované hlasové ovládání nabídne uživateli další možnost, jak interagovat s řídicí jednotkou. Hlasové ovládání si neklade za cíl nahradit existující uživatelské rozhraní ve formě webové aplikace.

5.2 Dostupná API

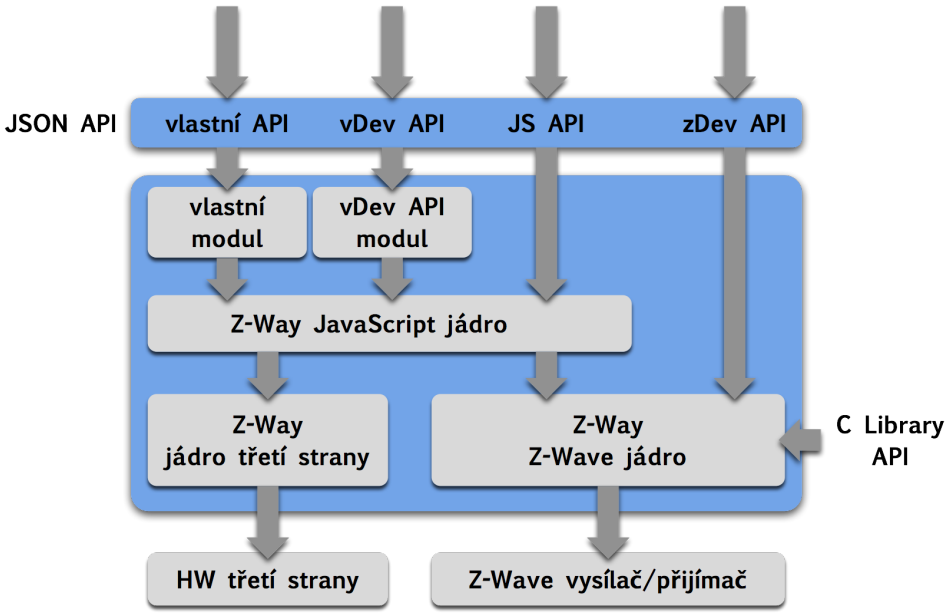
5.2.1 Architektura softwaru Z-Way

Pro pochopení možností ovládání řídicí jednotky je třeba nejprve prostudovat architekturu softwaru Z-Way. Nejdůležitější částí softwaru Z-Way je jádro Z-Wave. Toto jádro používá standardní Sigma Designs Serial API, prostřednictvím kterého komunikuje s kompatibilním Z-Wave hardwarem. Kromě toho nabízí jádro také některé další

funkce, například změnu frekvence Z-Wave vysílače/přijímače. Sigma Designs Serial API je dostupné pouze pro vlastníky Sigma Designs Development Kit[1]. [31]

Další významnou částí Z-Way softwaru je Z-Way JavaScript jádro. Zatímco jádro Z-Wave slouží pro konfiguraci Z-Wave sítě a komunikaci s jednotlivými zařízeními, Z-Way JavaScript jádro se nachází o úroveň výše a přidává další možnosti a funkce (automatizace, vyšší úroveň logiky). Implementace Z-Way JavaScript jádra je rozdělena do tzv. modulů, které implementují široké spektrum aplikací nad jádrem Z-Wave. Kromě toho může Z-Way JavaScript jádro ovládat i technologie třetích stran (např. EnOcean, či jakoukoli technologii založenou na protokolu HTTP). V případě potřeby je možné nahradit Z-Way JavaScript jádro vlastním řešením. [31]

Součástí softwaru Z-Way je dále vestavěný webový server. Veškerá komunikace probíhající mezi uživatelským rozhraním a softwarem Z-Way využívá tohoto webserveru. Za tímto účelem poskytuje webserver JSON[2] API, které přes HTTP/JSON volání provádí požadované akce na straně řídicího softwaru.



Obrázek 5.2: Architektura řídicího softwaru Z-Way s důrazem na dostupná API. Podle [31]

Software Z-Way nabízí několik API pro programování vlastních aplikací. Funkce jádra Z-Wave jsou přístupné přímo prostřednictvím Z-Wave Device API (zDev API). Pro programátory jsou k dispozici dvě verze zDev API.

1. **JSON API**: Funkce zDev API mohou být použity přes JSON API implementované vestavěným webserverem. Tuto verzi zDev API mohou používat rozličná uživatelská rozhraní (např. Expert UI).

[1]Dostupné z http://z-wave.sigmadesigns.com/design-z-wave/embedded-development-kits/

[2]Zkratka JSON z anglického výrazu JavaScript Object Notation označuje formát dat. Jedná se o jednoduchý formát, který je dobře čitelný pro člověka a jednoduchý z hlediska zpracování a generování pro programy. Základem formátu JSON je seřazený seznam párů klíč-hodnota. Více informací je možné nalézt na http://www.json.org.

2. **C Library API**: Všechny funkce zDev API jsou dostupné přes volání funkcí knihovny jazyka C. Toto API slouží pro vývoj aplikací, které poběží po boku softwaru Z-Way na stejné řídicí jednotce.

Software Z-Way je dodáván s celou řadou modulů, které umožňují snadné zprovoznění domácí automatizace. S využitím těchto modulů implementuje Z-Way také Virtual Device API (vDev API). Toto API využívá informací poskytnutých zDev API a na jejich základě vytváří model virtuálních zařízení. Díky tomu je možné reprezentovat zařízení v Z-Wave síti sjednoceným způsobem tak, aby k nim mohli snáze přistupovat vyšší vrstvy prostřednictvím jednotného JSON rozhraní. Díky tomuto sjednocení je vDev API vhodné pro návrh vlastního uživatelského rozhraní. [31]

Kromě dodávaných modulů je možné pro software Z-Way vyvíjet také vlastní moduly. Ty obvykle implementují pokročilé funkce (alarm, notifikace, automatizace osvětlení) a využívají služeb JavaScript jádra. Za účelem vývoje vlastních modulů je na straně řídicí jednotky k dispozici JavaScript běhové prostředí, ve kterém jsou vytvořené JavaScript moduly spouštěny. Veškeré funkce JS API jsou navíc dostupné i na straně klienta (např. webový prohlížeč), což nabízí zajímavé možnosti testování a ladění. [31]

Jak již bylo řečeno, implementace hlasového ovládání představuje další formu uživatelského rozhraní. Jistě se tedy bude hodit jednotný přístup k jednotlivým fyzickým zařízením a funkcím, který poskytuje Virutal Device API (vDev API). Dále se tedy podíváme na podrobnější popis tohoto rozhraní. Kromě vDev API se blíže seznámíme také s JavaScript API (pro případný návrh modulů na straně řídicí jednotky) a zDev API (umožní spravovat Z-Wave síť - přídávání/odebírání zařízení).

5.2.2 zDev API

Z-Wave Device API zprostředkovává základní funkce jádra Z-Wave. Tyto funkce můžeme rozdělit do dvou kategorií.

1. **Správa Z-Wave sítě**: Tato kategorie zahrnuje přidávání zařízení do Z-Wave sítě a jejich případné odebrání. Dále je pomocí těchto funkcí možné spravovat směrovací tabulku a udržovat tak vytvořenou Z-Wave síť přehlednou a stabilní.

2. **Ovládání jednotlivých zařízení**: Tyto funkce umožňují ovládat jednotlivá fyzická zařízení na dané Z-Wave síti.

V našem případě neimplementujeme uživatelské rozhraní jako součást řídicí jednotky, ale jako aplikaci na mobilním telefonu. Z existujících verzí zDev API nás tedy zajímá verze označená JSON API, která umožňuje volat tyto funkce přes webserver vestavěný v softwaru Z-Way.

Základem JSON API jsou HTTP volání ve formě `http://<ADRESA>:8083/<URL>`, kde `<ADRESA>` je IP adresa řídicí jednotky a `<URL>` je v případě zDev API tvořeno jako `/ZWaveAPI/Run/<příkaz>`. Pro vykonání příkazu se používá HTTP metoda POST (metoda GET je aktuálně také funkční, ale v budoucnu bude pravděpodobně zastaralá). Pokud chceme zapnout zařízení (přepínač) s ID 2, můžeme použít následující URL.

```
/ZWaveAPI/Run/devices[2].instances[0].Basic.Set(255)
```

Protože se chystáme pro ovládání zařízení použít vDev API, u zDev API se zaměříme především na příkazy správy Z-Wave sítě. Seznam nejdůležitějších funkcí, které bychom mohli použít, je uveden níže.

1. `controller.AddNodeToNetwork(M)` - Spustí (M=1), či zastaví (M=0) proces přidání nového zařízení do Z-Wave sítě. Proces inkluze je automaticky ukončen po 20 sekundách.

2. `controller.RemoveNoteFromNetwork(M)` - Spustí (M=1), či zastaví (M=0) proces odebrání zařízení ze Z-Wave sítě. Proces exkluze je automaticky ukončen po 20 sekundách.

3. `controller.SetDefault()` - vrátí řídicí jednotku do továrního nastavení

Více informací o dalších příkazech a funkcích dostupných přes zDev API můžeme nalézt v Z-Way dokumentaci pro vývojáře[3], či v oficiálním prohlášení[4] na stránkách z-wave.me.

5.2.3 vDev API

Nejvhodnějším rozhraním pro návrh uživatelského ovládání je vDev API. Jednotlivá připojená fyzická zařízení jsou mapována na virtuální zařízení tak, aby byl zjednodušen a sjednocen přístup k jejich datům a funkcím. Pokud má fyzické zařízení více různých funkcí (např. měření teploty a měření vlhkosti), každá z těchto funkcí se namapuje na samostatné virtuální zařízení. Tím se vDev API odlišuje od zDev API. Níže jsou uvedeny všechny funkce vDev API.

1. **Seznam virtuální zařízení**: Poskytuje virtuální namapování všech fyzických zařízení na dané Z-Wave síti.

2. **Funkce fyzických zařízení**: Zobrazí všechny funkce daného fyzického zařízení jako jednotlivá virtuální zařízení.

3. **Kontextové informace**: Umožňuje koncovému uživateli přidávat kontextové informace jako například uživatelské profily, seznam místností, aplikace třetích stran, atp..

4. **Správa událostí**: Zpracovává události v řídicí jednotce uživatelsky přívětivým způsobem tak, aby mohli být například použity pro upozornění (notifikace) v uživatelském rozhraní.

Podobně jako v případě zDev API i vDev API je možné používat jak vzdáleně (JSON API přes webový server), tak lokálně (pomocí jazyka JavaScript). V případě přístupu přes JSON API je URL adresa tvořena jako `/ZAutomation/api/v1/<příkaz>`. JSON API verze podporuje metody POST (vytváření záznamů), PUT (aktualizování záznamů), GET (získání záznamů/dat, nebo jejich seznamu) a DELETE (odstranění existujících záznamů).

[3]Dostupné z https://github.com/Z-Wave-Me/Z-Way-Manual/raw/master/zwayDev.pdf
[4]Dostupné z https://forum.z-wave.me/viewtopic.php?f=3417t=19742

Velmi podrobný popis vDev API včetně ukázkových kódů je k dispozici i přes platformu Apiary[5]. Zde je možné nelézt nejen podporované příkazy/funkce, ale také očekáváné struktury JSON odpovědí. To velmi usnadní použití tohoto API. Pro představu uvedeme několik základních příkazů, bez kterých se při implementaci uživatelského rozhraní neobjedeme.

Seznam všech virtuálních zařízení včetně jejich parametrů získáme pomocí následujícího příkazu (metoda GET). Vrácený seznam zařízení obsahuje pouze ta zařízení, která byla aktualizována od okamžiku definovaného parametrem `<čas>`[6].

`/ZAutomation/api/v1/devices?since=<čas>`

Pro každé virtuální zařízení bude vrácený seznam obsahovat JSON strukturu popisující toto zařízení. Příklad takové struktury pro zařízení typu teploměr může vypadat následovně.

```
{
    creationTime: 1461244852,
    creatorId: 29,
    deviceType: "sensorMultilevel",
    h: 1303283136,
    hasHistory: false,
    id: "ZWayVDev_zway_2-0-49-1",
    location: 2,
    metrics: {
        probeTitle: "Temperature",
        scaleTitle: "°C",
        level: 24,
        icon: "temperature",
        title: "Temperature Door - Living Room",
        modificationTime: 1463128674,
        lastLevel: 24
    },
    permanently_hidden: false,
    probeType: "temperature",
    tags: [],
    visibility: true,
    updateTime: 1464684789
}
```

Zdrojový kód 5.1: Popis virtuálního zařízení ve formátu JSON

Nebudeme zde popisovat význam jednotlivých polí, které tato JSON struktura obsahuje. Více informací je možné získat přes výše zmíněnou platformu Apiary. Za zmínku ovšem stojí pole s klíčem `deviceType` a `probeType`[7]. Tato dvě pole totiž poměrně jasně definují o jaký typ zařízení se jedná. Z uvedeného příkladu se tak například můžeme dočíst, že jde o víceúrovňový senzor měřící teplotu.

[5] Dostupné z http://docs.zwayhomeautomation.apiary.io/

[6] Časový okamžik je uváděn v počtu sekund, které uplynuly od roku 1970.

[7] Více zde: http://docs.zwayhomeautomation.apiary.io/#reference/devices/virtual-device

Při vyčítání hodnot ze senzorů stačí z JSON popisu zařízení extrahovat hodnotu pod klíčem `level` ve struktuře označené `metrics`. Pokud chceme měnit nastavení daného virtuálního zařízení (např. přejmenování), upravíme příslušnou hodnotu a aktualizujeme příslušné zařízení zasláním JSON popisu metodou PUT na URL níže.

`/ZAutomation/api/v1/devices/<id>`

Ovládání aktuátorů probíhá lehce odlišně. K dispozici zde máme sadu příkazů, které je možné volat metodou GET pro aktuátor s daným identifikátorem. URL takového volání vypadá následovně.

`/ZAutomation/api/v1/devices/<id>/command/<příkaz>`

Seznam podporovaných příkazů je uvedený v příloze B.

5.2.4 JavaScript API

JavaScript API (JS API) nabízí stejnou sadu funkcí jako zDev API a navíc přidává možnost spustit nad těmito funkcemi/daty JavaScript kód. Existují dvě možnosti využití JavaScript API.

1. **JSON API**: Funkce JavaScript API mohou být použity přes JSON API implementované vestavěným webserverem. Takto je možné spouštět JavaScript kód prostřednictvím standardního webového prohlížeče.
2. **JS API pro moduly**: JavaScript kód může být také trvale uložen a spouštěn přímo na řídicí jednotce. V tomto případě mluvíme o tzv. modulech.

Na rozdíl od výše uvedených API nabízí JavaScript API téměr nekonečné možnosti z hlediska vývoje (automatizace, loggování dat, zabezpečení domácnosti, předpověď počasí,...). Veškeré moduly, které jsou součástí Z-Way softwaru, jsou navíc open-source a je možné se jimi inspirovat při vývoji vlastních modulů.

V našem případě budeme JavaScript API používat maximálně pro psaní vlastních doplňkových modulů. Stručný přehled vytváření takových modulů uvedeme v následující sekci.

5.3 Automatizační subsystém

5.3.1 Sběrnice událostí

Automatizační subsystém softwaru Z-Way umožňuje psát moduly pomocí jazyka JavaScript. Základem automatizačního subsystému (a tedy i modulů) je sběrnice distribuující události (Event Bus). Tyto události může generovat samotná Z-Wave síť, nebo mohou vzniknout na základě interakce uživatele (rozsvícení světla, či například zaslání e-mailu). Ke sběrnici mají přístup všechny moduly a to jak pro čtení, tak pro generování událostí. Moduly typicky nepřijímají veškeré události, ale filtrují je podle svého určení. Modul Automatické osvělení tak například odchytává pouze události generované senzory s binárním výstupem (senzor pohybu). Tímto způsobem si tedy jednotlivé moduly navzájem vyměňují data.

5.3.2 Module.json

Module.json definuje vlastnosti/konfiguraci modulu a je zpracován automatizačním subsystémem. Jedná se o soubor ve formátu JSON, který popisuje název, verzi, ikonu, autora, kategorii, repozitář se zdrojovými kódy, konfigurovatelné položky, atp.. Pole s největším významem včetně popisu jsou uvedena níže. [31]

1. **singleton**: Určuje, zda může být spuštěna jedna, nebo více instancí.
2. **moduleName**: Představuje název modulu.
3. **schema**: Popisuje parametry (nezbytné i volitelné), které je nutné konfigurovat pro vytvoření instance modulu.
4. **options**: Přidává popisy parametrů a nápovědu pro zobrazení v grafickém uživatelském rozhraní.
5. **defaults**: Struktura popisující defaultní nastavení modulu.

5.3.3 index.js

Soubor index.js, který je potomkem třídy AutomationModule, definuje chování modulu a jedná se tak o klíčovou třídu každého modulu. Níže je uveden nejjednodušší příklad třídy index.js, tzv. Hello, World!.

```
1  function SampleModule (id, controller) {
2  SampleModule.super_.call.init(this, id, controller);
3      this.greeting = "Hello, World!";
4  }
5
6  inherits(SampleModule, AutomationModule);
7  _module = SampleModule;
8
9  SampleModule.prototype.init = function () {
10     this.sayHello();
11 }
12
13 SampleModule.prototype.sayHello = function () {
14     debugPrint(this.greeting);
15 }
16
17 SampleModule.prototype.stop = function () {
18     SampleModule.super_.prototype.stop.call(this);
19 }
```

Zdrojový kód 5.2: Příklad třídy index.js podle [31]

První část uvedené třídy index.js (řádky 1 až 7) představuje kontruktor a vytváří instanci modulu. Tato část by měla být víceméně stejná pro každý modul. Hlavní rozdíly přichází v metodě init, která je volaná při vytváření instance modulu. Zde se definuje chování modulu.

6. Komunikace s řídicí jednotkou

6.1 Požadavky

Aby mohlo mobilní zařízení komunikovat s řídicí jednotkou, je třeba splnit několik předpokladů. Obě zařízení, tj. řídicí jednotka i mobilní zařízení, se musí nacházet buď na stejné lokální počítačové síti, či musí být obě připojena do internetu. Jen tak zaručíme, že bude možné mezi mobilním zařízením a řídicí jednotkou navázat spojení. V obou případech budou zařízení komunikovat s využitím rodiny protokolů TCP/IP[1]. Z toho plynou požadavky i na mobilní zařízení, které musí podporovat tento typ komunikace. Bavíme se tedy především o tzv. chytrých telefonech vybavených typicky operačním systémem Android, iOS, či Windows Phone.

6.2 Internet protokol

6.2.1 Úvod

Pro pochopení komunikace je třeba nejprve porozumět pojmu IP adresa. Jedná se o jedinečný identifikátor, který je přidělen každému zařízení připojenému do počítačové sítě (internetu). Na základě této IP adresy vysílá IP protokol do počítačové sítě zprávy zvané datagram. IP adresa tedy určuje cílové zařízení, na které má být IP datagram směrován. [10]

V případě řídicí jednotky jsme IP adresu nastavili staticky, jak bylo popsáno v jedné z předchozích kapitol. Vzhledem k tomu, že mobilní zařízení připojujeme obvykle k různým sítím, nevolíme u něj IP adresu staticky, ale necháme ji dynamicky přiřazovat. O dynamické přiřazování IP adres se obvykle stará router podporující DHCP[2].

6.2.2 IPv4

V současnosti je nejpoužívanější formát IP adres tzv. IPv4. Jedná se o původní formát adres repzrezentovaných 32 bitovým číslem. K dispozici je tak 2^{32} IPv4 adres ve formátu IPv4, což představuje zhruba $4,3 \cdot 10^9$ adres. Odečteme-li vyhrazené adresy, dostáváme něco okolo $3,7 \cdot 10^9$ adres. [10]

IPv4 adresa se dnes skládá ze tří částí: číslo sítě, číslo podsítě a číslo síťového rozhraní. V závislosti na tom, kolik bitů připadá každé z těchto částí rozlišujeme několik tříd IP adres. Kromě jednotlivých tříd byl později přidán i tzv. CIDR, který umožňuje umístit hranici mezi číslem sítě a číslem podsítě mezi konkrétní dva bity. Díky tomuto dělení je možné určit jedinečné umístění IP adresy v rámci internetu. [10]

[1] Rodina TCP/IP představuje sadu protokolů, které se používají pro komunikaci v rámci počítačových sítí. IP (Internet protokol) označuje základní protokol síťové vrstvy. TCP (Transmission Control Protocol) je protokol transportní vrstvy, který zajišťuje spolehlivost spojení.

[2] DHCP, zkratka z anglického výrazu Dynamic Host Configuration Protocol, představuje síťový protokol, který zajišťuje dynamické přirazování IP adres zařízením připojeným do počítačové sítě.

Běžný uživatel se s IP adresou ve formátu IPv4 setká v podobě, kdy jsou čtyři decimální čísla (v rozsahu 0 až 255) oddělena tečkami. Každé z těchto čtyř čísel odpovídá jedné osmici bytů z 32 bitové reprezentace. IP adresa ve formátu IPv4 tak může vypadat například následovně.

```
192.168.0.21
```

6.2.3 IPv6

Ačkoli byl počet IPv4 adres v počátcích internetu dostatečný, postupem času začaly IP adresy docházet. Proto se objevil formát IPv6, kde každá IP adresa je reprezentována 128 bitovým číslem. K dispozici je tak 2^{128} IPv6 adres, což představuje zhruba $3,4 \cdot 10^{38}$ adres. Jedná se o dostatečný počet, aby každé zařízení, které je dnes připojené do internetu, mohlo mít svou jedinečnou IPv6 adresu. Přestože byl tento formát uveden již v roce 1996, v současné době má IPv6 adresu pouze okolo 10% zařízení a zdá se, že v blízké budoucnosti nezíská převahu nad IPv4. Podobně, jako v případě IPv4, i u IPv6 se používá notace CIDR pro rozdělení IPv6 adresy na číslo sítě a číslo podsítě. [10]

V textové podobě má tvar osmi čtveřic hexadecimálních číslic, které jsou odděleny tečkami. Každá z těchto čtveřic odpovídá 16 bitům a leží v rozsahu 0000 až ffff. Možná podoba IP adresy ve formátu IPv6 je uvedena níže.

```
2001:0db8:85a3:08d3:1319:8a2e:0370:7344
```

6.3 Problém s NAT

Jedním z mechanismů, který řeší problém s nedostatkem IPv4 adres, je tzv. NAT (Network Address Translation). Tento mechanismus využívá skutečnosti, že zařízení, která nejsou připojena do internetu, mohou mít stejnou IP adresu (mluvíme o tzv. privátní IP adrese). V případě, že se chce zařízení připojit do internetu, zajistí NAT překlad privátní IP adresy do veřejné IP adresy (jedinečné v rámci internetu). Veřejná IP adresa se uděluje pouze dočasně ze seznamu veřejných IP adres, kterými disponuje zařízení zajišťující překlad. Takovým zařízením je ve většině domácností router. Díky NAT je tedy dnes možné přistupovat k internetu z více zařízení, než jaký je počet IPv4 adres. [10]

Použití překladu adres s sebou přináší také několik nevýhod. Zařízení s privátní IP adresou (umístěné v lokální síti za NATem) není přímo přístupné z internetu, neboť IP datagramy nebudou směrovány na privátní IP adresu tohoto zařízení. Místo toho budou datagramy směrovány na zařízení s veřejnou IP adresou o stejné hodnotě. V případě řídicí jednotky inteligentní domácnosti se jedná o poměrně vážný problém, neboť není možné přímo adresovat toto zařízení z internetu a tak vzdáleně ovládat a monitorovat inteligentní domácnost. Kvůli NAT také nemusí fungovat některý software. K problémům může dojít kvůli absenci překladu IP adres obsažených v aplikačních datech (překládají se pouze adresy v hlavičkách protokolů).

7. Přístup z lokální sítě

7.1 Úvod

O přístupu z lokální sítě mluvíme, pokud se obě zařízení (řídicí jednotka i mobilní telefon) nachází na stejné lokální fyzické síti. Jedná se tak o situaci, kdy se k řídicí jednotce připojujeme z mobilního telefonu, který se nachází v této inteligentní domácnosti. Lokální přístup představuje jednodušší případ, neboť zúčastněná zařízení nejsou umístěná za NATem a mohou tak spolu komunikovat napřímo přes TCP/IP. Ani jedno ze zařízení přitom nemusí být připojeno do internetu. Pro vzájemnou komunikaci postačí infrastruktura lokální (domácí) sítě.

Pokud existuje možnost připojit se k řídicí jednotce lokálně, měli bychom ji upřednostnit před připojením přes internet. Díky tomu, že data mezi mobilním telefonem a řídicí jednotkou neproudí přes internet, ale pouze na lokální síti, je tento typ připojení logicky mnohem bezpečnější. Dále, protože jsou datagramy směrovány pouze v rámci domácí sítě, očekáváme kratší odezvy v komunikaci a tedy rychlejší přenos dat v obou směrech.

Pro přístup k řídicí jednotce z mobilního telefonu na lokální síti potřebujeme znát IP adresu řídicí jednotky a příslušný port. Jen tak je možné navázat spojení s webserverem vestavěným v řídicí jednotce a ovládat/monitorovat inteligentní domácnost. V našem případě známe obě tyto hodnoty, neboť IP adresu řídicí jednotky jsme již dříve nastavili staticky a port je dán použitým JSON API. Momentálně používáme internet protokol verze 4, ale operační systém Raspbian i řídicí software Z-Way podporují také internet protokol verze 6. Přestože jsou níže uvedené postupy zaměřené na IPv4, lze je s drobnými úpravami použít analogicky i na IPv6.

7.2 Skenování zařízení na síti

7.2.1 Motivace

Ačkoli známe IP adresu i port řídicí jednotky, pro běžného uživatele mohou být tyto údaje neznámé a těžko zjistitelné. Pro takového uživatele by bylo tedy poměrně obtížné nakonfigurovat aplikaci hlasového ovládání tak, aby se dokázala připojit k dané řídicí jednotce. Vzhledem k tomu, že JSON API používá defaultní port 8083, můžeme tento port nastavit fixně v kódu aplikace a nemusíme jeho konfigurací obtěžovat uživatele.

Co se týká IP adresy, zde je řešení o něco složitější. IP adresu dopředu neznáme, neboť je přidělena automaticky přes DHCP při připojení řídicí jednotky do domácí počítačové sítě, či je nastavena uživatelem staticky. V případě dynamicky přidělované adresy navíc nemusí být tato adresa stálá, ale může se v průběhu času měnit. Jednou z možností, jak tento problém vyřešit, je skenování zařízení na síti a automatická identifikace řídicí jednotky. Jedná se tedy o postupné procházení všech zařízení na lokální síti a zjišťování, zda tato zařízení odpovídají na daném portu (8083) a zda se jedná o požadovanou řídicí jednotku.

7.2.2 Určení rozsahu IP adres

Abychom jednotlivá zařízení identifikovali, je potřeba znát, v jakém rozsahu IP adres se mohou připojená zařízení nacházet. Vycházet přitom můžeme ze znalosti dělení IP adresy na číslo sítě, číslo podsítě a číslo síťového rozhraní. Zatímco číslo sítě a číslo podsítě zůstává v rámci lokální sítě neměnné, číslo síťového rozhraní se liší pro každé připojené zařízení. V prvním kroku je tedy třeba nalézt číslo sítě, číslo podsítě a rozsah čísel síťového rozhraní.

Předpokládejme, že se obě zařízení nachází na stejné lokální počítačové síti, neznáme IP adresu řídicí jednotky a chceme se k této řídicí jednotce připojit z aplikace v mobilním telefonu. Protože je mobilní telefon připojen do sítě, bude mít jistě přidělenou IP adresu, která má číslo sítě a číslo podsítě totožné s příslušnými čísly IP adresy řídicí jednotky. IP adresy těchto dvou zařízení se budou lišit pouze v čísle síťového rozhraní.

Číslo sítě a číslo podsítě můžeme určit z IP adresy mobilního telefonu na základě třídy IP adresy, resp. hodnoty CIDR. Jakmile známe pozici hranice oddělující číslo síťového rozhraní, víme také, kolik bitů na toto číslo připadá. Postoupnou iterací přes všechny kombinace těchto bitů získáme celkový rozsah čísel síťových rozhraní.

Výše popsaný postup si ukážeme na příkladu. Uvažujme, že mobilní telefon s aplikací hlasového ovládání má následující IP adresu:

```
192.168.0.21
```

Dále jsme zjistili, že se jedná o často používanou třídu IP adres C, resp. CIDR = 24, což znamená, že na číslo síťového rozhraní připadá 8/32 bitů. To představuje $2^8 = 256$ různých čísel síťových rozhraní. Po odečtení vyhrazených adres a vyloučení IP adresy mobilního telefonu dostáváme celkem 253 možných IP adres. Skenování zařízení na dané lokální síti bude probíhat v rozsahu IP adres níže.

```
192.168.0.1 - 192.168.0.254
```

7.2.3 Identifikace řídicí jednotky

Nyní tedy víme, přes které adresy je potřeba síť skenovat. V dalším kroku ověříme, zda je některá z těchto IP adres přidělena řídicí jednotce. Protože víme, že řídicí jednotka obsahuje vestavěný webový server odpovídající na portu 8083, vyřadíme ze seznamu všechny IP adresy, u kterých není možné navázat spojení na tento port.

Takto získáme seznam všech zařízení na lokální síti, která odpovídají na portu 8083. Přestože takových zařízení moc nebude (pravděpodobně získáme jen jednu IP adresu odpovídající naší řídicí jednotce), předpokládejme, že objevíme i jiná zařízení odpovídající na portu 8083. Zde můžeme použít některou z funkcí JSON API a porovnat odpověď HTTP volání s očekáváním. Nabízí se například příkaz s následujícím URL, který vrátí odpověď `OK`, pokud se jedná o řídicí jednotku.

```
/ZAutomation/api/v1/status
```

7.2.4 Implementace skenování v OS Android

Výše uvedený mechanismus budeme implementovat jako součást aplikace HomeVoice pro OS Android. Vývoj probíhá v Javě s využitím Android SDK verze 24 (určena pro Android 7.0), neboť se jedná o nejnovější verzi API v době zahájení návrhu.

Proces skenování je inspirován implementací v ukázkové aplikaci Z-Way-Android[1], kterou poskytuje společnost Z-Wave.Me. Hlavní rozdíl spočívá v odlišném vyčítání informací o síti a síťových rozhraních (používáme standardní metody Android API namísto volání systémových příkazů). Další rozdíl představuje princip využití funkce skenování. Zatímto ukázková aplikace skenuje síť pouze při konfiguraci aplikace (napovídá uživateli možnou IP adresu řídicí jednotky), v našem případě skenujeme síť na pozadí při každé změně sítě, přístupové údaje konfigurujeme automaticky a uživatel tak nepřijde vůbec do styku s použitou IP adresou.

Základ skenovacího procesu představuje zjištění IP adresy a CIDR sítě. Níže je uvedena metoda, která pro dané síťové rozhraní zjistí název síťového rozhraní, použitou IP adresu a CIDR za pomocí standardních metod Android API.

```
1  private static NetInfo getIntfInfo(NetworkInterface ni) {
2      if (ni == null) return new NetInfo();
3      for(InterfaceAddress ia : ni.getInterfaceAddresses()) {
4          if(!ia.getAddress().isLoopbackAddress()) {
5              NetInfo info = new NetInfo();
6              info.m_intf = ni.getDisplayName();
7              info.m_ip   = ia.getAddress().getHostAddress();
8              info.m_cidr = ia.getNetworkPrefixLength();
9              return info;
10         }
11     }
12     return new NetInfo();
13 }
```

Zdrojový kód 7.1: Implementace funkce pro zjištění parametrů síťového rozhraní

Jak je patrné z uvedeného kódu, pro dané síťové rozhraní vracíme parametry první IP adresy, která není typu loopback (vnitřní smyčka). Pokud toto síťové rozhraní nedisponuje žádnou IP adresou, parametry rozhraní nevyplňujeme, aby se předešlo dalšímu zpracování tohoto rozhraní. Z vybraných rozhraní se následně vybere první rozhraní, které má validní IP adresu (jiná než 0.0.0.0).

Skenování sítě může být obzvlášť při větším počtu připojených zařízení poměrně časově náročné. Vyhneme se tedy implementaci skenování v hlavním vlákně aplikace, což by mohlo negativně ovlivnit odezvu uživatelského rozhraní. Ačkoli bychom mohli vytvořit další vlákno a provést skenování v něm, využijeme již připravenou třídu `AsyncTask`, která je určená pro tento typ úloh. Teprve v této třídě vytváříme pracovní vlákna, která asynchronně zkouší připojení k IP adresám v daném rozsahu.

[1] Dostupné z https://github.com/Z-Wave-Me/Z-Way-Android

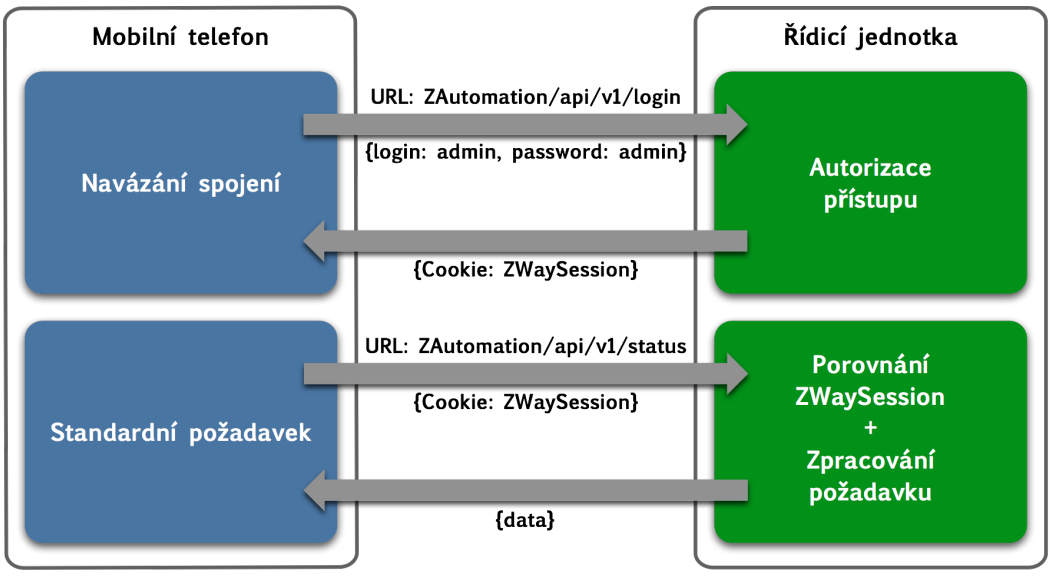
Ze známé IP adresy a hodnoty CIDR dále určíme rozsah adres, přes které je třeba skenovat. Procházení jednotlivých IP adres se rozdělí mezi pracovní vlákna, která pak asynchronně zkouší navázat spojení (Socket) na danou IP adresu a port (8083). Pokud nedojde ke spojení do 500 ms, vyřadí se příslušná IP adresa ze seznamu. Implementace funkcí popsaných v tomto odstavci je převzata z ukázkové aplikace Z-Way-Android a proto zde nebude více rozvedena.

Podstatný rozdíl přichází při zpracování získaných informací, tj. IP adres, které mohou příslušet naší řídicí jednotce. Zatímco v ukázkové aplikaci je tento seznam použit pouze jako nápověda pro uživatele, který musí manuálně vybrat adresu řídicí jednotky, v našem případě je řídicí jednotka identifikována automaticky. Za tímto účelem jsme si vytvořili třídu `ZWayAuthQueue`. Jedná se o třídu implementující FIFO[2] frontu, která testuje jednotlivé IP adresy. Jako test se používá proces autorizace přístupu k řídicí jednotce na lokální síti (podrobně popsán v další sekci). Pokud se autorizace podaří, našli jsme řídicí jednotku. Jestliže se autorizace nezdaří pro žádnou z navrhovaných IP adresy, řídicí jednotka se zřejmě nenachází na lokální síti a pokusíme se navázat spojení přes internet.

7.3 Autorizace na lokální síti

7.3.1 Princip autorizace

Abychom předešli zneužití řídicí jednotky kýmkoli, kdo získá přístup do naší lokální sítě, může být řídicí jednotka chráněna uživatelským jménem a heslem. Tento bezpečnostní prvek je součástí softwaru Z-Way a nedovolí nám používat funkce JSON API, pokud náš přístup není autorizovaný.



Obrázek 7.1: Princip autorizace přístupu v softwaru Z-Way s využitím sessions.

[2]FIFO (zkratka z anglického výrazu first in, first out) označuje v našem případě metodu pro zpracování prvků ve frontě, kde nejdříve se zpracuje prvek, který byl přidán do fronty jako první.

Pro autorizaci přístupu k funkcím řídicí jednotky využívá software Z-Way principu tzv. sessions. V prvním kroku zašle klient (v našem případě mobilní telefon) pomocí metody POST přihlašovací údaje (login a password ve formátu JSON) na URL `/ZAutomation/api/v1/login`. Pokud jsou přihlašovací údaje správné, řídicí jednotka vytvoří identifikátor nazvaný **ZWaySession**, který jednoznačně identifikuje komunikaci pocházející od daného klienta. Tento identifikátor je následně zaslán klientovi a obě strany si ho uloží. V případě webového prohlížeče bývá takový identifikátor označován jako Cookie. Při zasílání dalších požadavků se již klient nemusí autorizovat pomocí přístupových údajů, ale stačí, když v hlavičce volání uvede vygenerovaný identifikátor **ZWaySession**.

7.3.2 Implementace autorizace v OS Android

Ačkoli princip autorizace zní poměrně jednoduše, samotná implementace v OS Android je o poznání složitější. Je třeba vytvořit a nakonfigurovat REST klienta, prostřednictvím kterého budeme volat funkce JSON API. Tento klient musí podporovat volání protokolu HTTPS a také zachycení, uložení a následné použití identifikátoru **ZWaySession**. Ukázková aplikace Z-Way-Android sice všechny výše popsané body obsahuje, ale toto řešení je dnes zastaralé a během testování se jevilo nespolehlivě. Rozhodli jsme se tedy implementovat jednotlivé body s využitím moderních knihoven.

1. **OkHttp**[3]: HTTP klient pro Android a Javu od společnosti Square, který zjednodušuje a optimalizuje použití HTTP volání. OkHttp poskytuje implementaci synchronních i asynchronních volání, šifrovacích protokolů SSL a TLS a je kompatibilní s knihovnou Retrofit. Jedná se o open-source knihovnu distribuovanou pod licencí Apache 2.0.

2. **Retrofit**[4]: REST klient pro Android a Javu od společnosti Square, který zjednodušuje výměnu dat mezi klientem a serverem. I v tomto případě se jedná o open-source knihovnu distribuovanou pod licencí Apache 2.0.

3. **Gson**[5]: Java knihovna od společnosti Google, která převádí Java objekty do jejich JSON reprezentace a naopak. Jedná se open-source knihovnu, která je distribuována pod licencí Apache 2.0.

V prvním kroku inicializujeme HTTPS klienta pomocí knihovny **OkHttp**. Vzhledem k tomu, že apriorně neznáme IP adresu řídicí jednotky a obecně ani adresu serveru pro vzdálený přístup, povolíme klientovi připojení k serveru s jakoukoli IP adresou, či jménem (hostname). Během konfigurace přiřadíme HTTP klientovi i správce souborů Cookies, který uloží **ZWaySession** a umožní pozdější použití tohoto identifikátoru.

Následně připravíme REST klienta s využitím **Retrofit**. Při inicializaci předložíme REST klientovi IP adresu řídicí jednotky, JSON převodník vytvořený pomocí knihovny **Gson** a HTTPS klienta z předchozího kroku. Získáme tím nejen snazší definování jednotlivých HTTP volání, ale také automatický převod přijatých dat ve formátu JSON na námi vytvořeného datové modely (Java objekty) a naopak.

[3]Dostupné z http://square.github.io/okhttp/
[4]Dostupné z http://square.github.io/retrofit/.
[5]Dostupné z https://github.com/google/gson.

Jakmile máme vytvořeného REST klienta, můžeme přistoupit k samotné autorizaci. Protože se jedná o první z mnoha použití tohoto klienta v rámci námi vyvíjené aplikace, podíváme se blíže, jak se taková volání realizují.

Každé volání musí mít definovanou HTTP metodu (GET, POST, PUT, DELETE a HEAD) a relativní URL v podobě tzv. anotace (uváděna symbolem @). Dále je možné URL dynamicky upravovat pomocí pole @Path, přidávat dotazy @Query, či například tělo volání @Body. V neposlední řadě je možné definovat také typ návratového objektu. Níže je uvedena definice volání pro autorizaci přístupu k řídicí jednotce.

```
1 public interface LocalAuthRequest {
2
3     @POST("/ZAutomation/api/v1/login")
4     Call<Object> auth(@Body LocalAuth localAuth);
5
6 }
```

Zdrojový kód 7.2: Definice HTTP volání pro lokální autorizaci s využitím knihovny Retrofit.

V těle autorizačního volání je předáván objekt `LocalAuth`, který v nejjednodušším případě obsahuje pouze pole `login` a `password`. Objekt `LocalAuth` je během realizace volání automaticky převeden na datovou strukturu ve formátu JSON.

```
1  public class LocalAuth {
2
3      public String login;
4      public String password;
5
6      public LocalAuth (String login, String password) {
7          this.login = login;
8          this.password = password;
9      }
10
11 }
```

Zdrojový kód 7.3: Definice objektu LocalAuth s členy login a password.

Pokud byla autorizace úspěšná, správce Cookies přiřazený HTTP klientovi obsahuje identifikátor **ZWaySession**.

8. Přístup z internetu

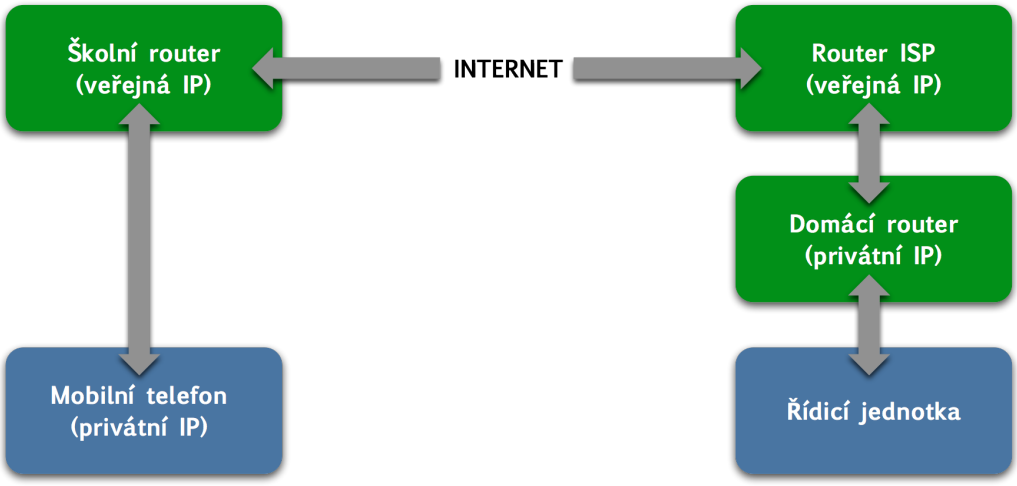
8.1 Úvod

V případě, že se mobilní telefon a řídicí jednotka nenachází na stejné lokální síti, postup popsaný v předchozí kapitole není až na několik výjimek použitelný. V praxi k této situaci dochází, pokud například chceme ovládat inteligentní domácnost ze školy, práce, či odkudkoli ze světa. Aby bylo vůbec možné navázat mezi zařízeními spojení, je třeba, aby byla obě připojena k internetu. Oproti přístupu z lokální sítě očekáváme pomalejší přenos a obecně i nižší bezpečnost.

8.2 Struktura vzdáleného přístupu

Zásadní problém při realizaci vzdáleného přístupu představuje možná struktura připojení mobilního telefonu a řídicí jednotky k internetu, kdy řídicí jednotka nemá přidělenu veřejnou IP adresu. Řídicí jednotka se v tomto případě nachází za jednou, či více vrstvami NAT, což znemožňuje přímý přístup k řídicí jednotce tak, jak je to možné na lokální síti. Níže rozebereme jednotlivé případy a jejich možná řešení.

V případě, že má alespoň náš domácí router veřejnou IP adresu, můžeme na tomto routeru nastavit přesměrování příslušných portů (8083, 8084) na naši řídicí jednotku a následně pro přístup používat veřejnou IP adresu domácího routeru. Horší situace nastává, pokud nemá náš domácí router statickou veřejnou IP adresu, ale tato adresa je dynamicky přidělována, tj. mění se v čase. Zde můžeme použít služby DynDNS, která zajistí přístup k zařízení s měnící se IP adresou prostřednictvím stabilního doménového jména. Existuje mnoho poskytovalů služby DynDNS.



Obrázek 8.1: Možná struktura připojení mobilního telefonu a řídicí jednotky k internetu při připojení ze školní sítě. Mobilní telefon i řídicí jednotka má privátní IP adresu a veřejná adresa je pouze zapůjčována mechanismem NAT. Ani domácí router nemá typicky veřejnou IP adresu, ale pouze privátní adresu v rámci lokální sítě poskytovatele internetu (ISP).

Typicky ani domácí router nemá přidělenu veřejnou IP adresu, ale nachází se na lokální síti poskytovatele internetu. Veřejnou IP adresu pak poskytovatelé internetu nabízí pouze za poplatek (např. O2 nabízí veřejnou IP adresu pro domácí router za 210 Kč bez DPH za měsíc). V tomto případě již nepomůže ani přesměrování portů, ani použití služby DynDNS, ale je třeba zvolit jiné řešení.

8.3 Služba find.z-wave-me

Společnost Z-Wave.Me nabízí vlastní službu pro vzdálený přístup k řídicí jednotce prostřednictvím stránky find.z-wave.me. Jedná se o volně dostupné řešení, které nám umožní jednoduše přistupovat k řídicí jednotce přes internet. Abychom mohli tuto službu použít, je třeba povolit vzdálený přístup v softwaru Z-Way a také musíme znát jedinečný identifikátor řídicí jednotky. Tento identifikátor je automaticky vygenerován při instalaci Z-Way softwaru a umožní službě pro vzdálený přístup rozlišit naši řídicí jednotku.

Při použití vzdáleného přístupu se mobilní telefon připojuje pomocí protokolu HTTPS k serveru find.z-wave.me podobně, jako k řídicí jednotce na lokální síti. Na základě identifikátoru řídicí jednotky zajistí server find.z-wave.me spojení s příslušnou řídicí jednotkou. Ačkoli podrobný popis funkce této služby není k dispozici, dle reakcí Z-Way komunity komunikuje tento server s řídicí jednotkou přes SSH tunel[1].

Obrázek 8.2: Princip vzdáleného přístupu přes službu find.z-wave.me

Vzhledem k tomu, že se jedná o hotové a spolehlivé řešení bez viditelných problémů, rozhodli jsme se nehledat další možná řešení, ale použít pro vzdálený přístup službu find.z-wave.me. Tato služba je navíc zdarma a hlavně nevyžaduje dodatečné nastavení mobilního telefonu, řídicí jednotky, ani domácí sítě. To předurčuje tuto službu pro použití běžnými uživateli bez hlubších znalostí počítačových sítí.

8.4 Autorizace přes internet

8.4.1 Princip autorizace

Při použití služby find.z-wave.me je autorizace přístupu přes internet velmi podobná autorizaci na lokální síti. I zde se používá protokol HTTPS a princip vytváření session. Na základě přístupových údajů (identifikátor řídicí jednotky, uživatelské jméno a heslo) vygeneruje server find.z-wave.me i řídicí jednotka příslušné Cookies, které umožní další komunikaci. Na rozdíl od lokálního přístupu se k identifikátoru **ZWaySession** přidává

[1] Více info na https://forum.z-wave.me/viewtopic.php?f=3422t=21890

i druhý identifikátor **ZBW_SESSID** (vygenerovaný službou find.z-wave.me). Další rozdíl spočívá v použité URL, kde pro autorizaci přes internet slouží URL `/zboxweb`.

8.4.2 Implementace autorizace přes internet v OS Android

Velká část implementace lokální autorizace je sdílena i s autorizací přes internet. Používáme zde naprosto stejného HTTPS klienta jako v případě autorizace na lokální síti. Konfigurace REST klienta se liší pouze v použité adrese řídicí jednotky (místo lokální IP adresy použijeme adresu serveru find.z-wave.me). Dále musíme rozšířit správce souborů Cookies, aby kromě **ZWaySession** ukládal také identifikátor **ZBW_SESSION**.

Další podstatný rozdíl je v definici autorizačního volání, kde se používá URL `/zboxweb` a přihlašovací údaje jsou namísto struktury JSON zakódovány do URL. Parametr `login` se předává ve formátu `identifikátor/uživatelské_jméno`, kde `identifikátor` jednoznačně rozlišuje řídicí jednotku v rámci služby find.z-wave.me. Pro přihlášení se jako parametr `act` používá fixní řetězec "login".

```
1 public interface RemoteAuthRequest {
2
3     @FormUrlEncoded
4     @POST("/zboxweb")
5     Call<Object> auth(@Field("act") String act,
6                       @Field("login") String login,
7                       @Field("pass") String password);
8
9 }
```

Zdrojový kód 8.1: Definice HTTP volání pro autorizaci přes internet.

Pokud se autorizace povedla, bude správce souborů Cookies obsahovat oba identifikátory, tj. **ZWaySession** a **ZBW_SESSID**. Takto připraveného REST klienta můžeme následně používat pro další volání stejně, jako v případě lokální autorizace.

9. PUSH notifikace

9.1 Motivace

Doposud jsme se zabývali tím, jak se z mobilního telefonu připojit k řídicí jednotce. Většina existujících uživatelských rozhraní používá pouze tento typ komunikace, kdy na základě interakce uživatele zašle dané rozhraní požadavek řídicí jednotce. Pokud chce uživatel monitorovat inteligentní domácnost v reálném čase, musí příslušné uživatelské rozhraní aktivně a periodicky vyčítat požadovaná data z řídicí jednotky. V případě mobilního telefonu by tak ovládací aplikace musela trvale běžet na pozadí, což má negativní vliv na spotřebu elektrické energie a vede k rychlejšímu vybíjení zařízení.

Vzhledem k výše uvedenému jsme se rozhodli navrhnout a implementovat vlastní řešení, kdy bude řídicí jednotka schopna iniciovat komunikaci a zasílat data do mobilního telefonu. Díky tomu můžeme prostřednictvím telefonu upozornit uživatele na události v době, kdy se skutečně staly, a ne až tehdy, kdy o tato data požádá uživatel. Možnost okamžitě informovat uživatele jistě najde uplatnění v krizových situacích, jako je například neoprávněné vniknutí do domu, požár, únik vody/plynu, atp.. V kontextu chytrých mobilních telefonů se takto zasílaná data (krátké informační zprávy) označují jako PUSH notifikace.

9.2 Firebase Cloud Messaging

Vhodný nástroj pro zasílání PUSH notifikací představuje služba Firebase Cloud Messaging (FCM) od společnosti Google. Díky tomuto nástroji je možné bezplatně doručovat krátké zprávy na mobilní telefony s operačním systémem Android a iOS, ale také do webových prohlížečů. Velikost zasílaných zpráv je omezena na 4 KB, což je dostatečná hodnota pro námi zamýšlené použití. Jednotlivé zprávy je možné zasílat na konkrétní zařízení, skupinu zařízení, či všechna zařízení, která jsou přihlášena k odběru daného typu zpráv.[47]

Implementace FCM se skládá typicky ze 3 částí.

1. **FCM server**: Server poskytovaný společností Google, který přijímá zprávy ze serverové aplikace a distribuuje je mezi příslušné klientské aplikace. Pro zasílání zpráv jsou k dispozici HTTP a XMPP[1] API.

2. **Klientská aplikace**: Klient běžící na iOS, Android, či v rámci webového serveru (JavaScript), který zajišťuje zpracování přijatých zpráv a obsahuje logiku přihlašování k odběru zpráv. Tuto část implementuje uživatel.

3. **Serverová aplikace**: Aplikace, která se stará o generování zpráv a jejich zasílání na FCM server přes HTTP, či XMPP. Tuto část implementuje uživatel.

[1]XMPP, zkratka z anglického výrazu Extensible Messaging and Presence Protocol, představuje sadu technologií určených pro komunikaci v reálném čase za pomocí XML. XMPP nachází uplatnění při vývoji aplikací pro chatování, audio/video hovory, atp.. Více informací o XMPP je možné nalézt zde: https://xmpp.org/about/technology-overview.html.

Než budeme moci přistoupit k návrhu serverové a klientské aplikace, musíme nejdřív založit účet v konzolové aplikaci[2] služby Firebase, kde následně vytvoříme nový projekt. Není potřeba žádná složitá konfigurace, stačí zadat pouze název projektu a zemi původu.

Aby mohla naše serverová aplikace komunikovat s FCM serverem, musí být veškeré žádosti autorizovány. Autorizační klíč **Web API Key** je vytvořen automaticky při založení konkrétního projektu a jeho hodnotu získáme prostřednictvím konzolové aplikace. Dále musíme ještě vygenerovat konfigurační soubor **google-services.json** pro klientskou aplikaci. V případě aplikace pro OS Android stačí vyplnit název balíčku mobilní aplikace[3] a připravený soubor stáhnout pro další použití.

9.3 Firebase modul pro Z-Way

9.3.1 Struktura Firebase modulu

Firebase serverová aplikace poběží v našem případě přímo na řídicí jednotce. Našim cílem je naimplementovat serverovou aplikaci jako rozšiřující modul automatizačního subsystému v softwaru Z-Way, který bude odchytávat vybrané události ditribuované po sběrnici (Event Bus). Modul zachycené události zpracuje, vygeneruje jejich popis a ve vhodné podobě ho odešle na FCM server. Modul navrhneme co nejobecněji tak, aby bylo možné jeho použití i dalšími moduly a klientskými aplikacemi.

V prvním kroku musíme zvolit, který z dostupných protokolů (HTTP, XMPP) použijeme pro komunikaci s FCM serverem. Zatímco přes HTTP je možné odesílat zprávy pouze jedním směrem (server → klient), XMPP umožňuje zasílat zprávy i ve směru opačném. Protože komunikaci ve směru od mobilního zařízení do řídicí jednotky jsme již vyřešili v předchozích kapitolách, spokojíme se s HTTP API.

Dále se musíme rozhodnout, která data bude moci uživatel v rámci tohoto modulu parametrizovat. Na základě prostudování nástroje FCM a existujících modulů automatizačního subsystému jsme určili následující sadu parametrů.

1. **Web API Key**: Autorizační klíč, kterým se autorizuje serverová aplikace při komunikaci s FCM serverem. Pomocí tohoto parametru bude možné vybrat, se kterým projektem v rámci služby Firebase má serverová aplikace komunikovat.

2. **Device ID**: Identifikátor klientské aplikace na konkrétním zařízení umožní adresovat notifikace na vybrané zařízení. Device ID musí vygenerovat aplikace, která bude následně přijímat PUSH notifikace.

3. **Filtr událostí**: Seznam událostí, které má modul zachytávat a jejichž popis má zasílat na FCM server. Konkrétní hodnoty je třeba přizpůsobit modulům, jejichž události chceme odchytávat. Název události je třeba zjistit pro konkrétní modul (např. modul Security Zone[4] generuje událost `security.intrusion.alarm`).

[2]Dostupné z https://console.firebase.google.com/
[3]Název balíčku rozlišuje konkrétní mobilní aplikaci a má následující podobu: com.yourapp.android.
[4]https://github.com/maros/Zway-SecurityZone

Výše uvedené parametry je třeba nastavit při inicializaci modulu, tzn. vytvoření instance. Následně je možné tyto parametry měnit pouze reinicializací příslušné instance. Kromě toho modul vygeneruje také vlastní virtuální zařízení, které umožní další ovládání jeho funkcí. V našem případě nakonfigurujeme virtuální zařízení jako přepínač, který umožní zapínat/vypínat zasílání notifikací.

9.3.2 Registrace k odběru událostí

Ze samotné implementace stojí za zmínku přihlášení k odběru událostí, neboť tento postup není příliš dobře zdokumentován. Každá ze seznamu událostí je zaregistrována pomocí příkazu `self.controller.on(nazev_udalosti, obsluha_udalosti)`, kde argument `obsluha_udalosti` představuje funkci volanou v případě zachycení daného typu události.

```
1 _.each(self.config.events,function(eventName) {
2     self.controller.on(eventName, self.handler);
3 });
```

Zdrojový kód 9.1: JS implementace registrace k odběru konkrétních událostí

9.3.3 Zasílání zpráv na FCM server

V případě zachycení události je zavolána příslušná metoda, která zajistí zaslání zprávy na FCM server. Jak již bylo řečeno, pro komunikaci s FCM serverem jsme zvolili HTTP protokol. Implementace zasílání notifikací pomocí tohoto protokolu je uvedena níže. Používáme HTTP metodu POST na URL `fcm.googleapis.com/fcm/send`. V hlavičce je dále třeba uvést **Web API Key**, v těle volání **Device ID** a také samotný obsah zasílané zprávy.

```
1  http.request({
2      method: 'POST',
3      url: 'https://fcm.googleapis.com/fcm/send',
4      headers: {
5          'Content-Type': 'application/json',
6           Authorization: 'key=' + self.config.api_key
7      },
8      data: JSON.stringify({
9          to: self.config.device_id,
10         data: {
11             text: notice.message
12         }
13     }),
14     timeout: 10000,
15     async: true
16 });
```

Zdrojový kód 9.2: JS implementace zasílání notifikací na server FCM

9.4 Firebase v OS Android

Ke zprovoznění zasílání PUSH notifikací schází ještě příprava klientské aplikace. Funkci klienta naimplementujeme v aplikaci HomeVoice, která zprostředkovává uživateli hlasové ovládání a komunikuje s inteligentní domácností. V rámci této práce chceme nastínit hlavně koncept zasílání dat z řídicí jednotky do mobilního telefonu pomocí PUSH notifikací. Nebudeme se tedy zabývat prokročilým zpracováním přijatých zpráv, ale jejich obsah zobrazíme jako standardní notifikaci v OS Android.

Aby mohla námi navrhovaná mobilní aplikace přijímat notifikace prostřednictvím služby Firebase, musíme přidat příslušné knihovny k projektu této aplikace. K dispozici je několik knihoven, které poskytují podporu různých nástrojů služby Firebase (analýza, mobilní reklama, databáze, atp.). Poslední verze knihovny pro zasílání zpráv a notifikací nese označení `com.google.firebase:firebase-messaging:10.2.4` a pro přidání využijeme standardní nástroj Gradle[5]. Dále je třeba k projektu přidat výše zmíněný konfigurační soubor **google-services.json**, díky kterému bude možné adresovat zprávy do naší aplikace.

Pro zpracování přijatých zpráv implementujeme službu na pozadí (Service), která bude zavolána v případě přijetí zprávy ze serveru FCM. Toho dosáhneme pomocí tzv. filtru záměrů[6], kdy na základě systémové události `com.google.firebase.MESSAGING_EVENT` dojde ke spuštění navrhované služby a předání obsahu zprávy. Tato služba je potomkem třídy `FirebaseMessagingService` a klíčová je pro nás implementace metody `onMessageReceived(RemoteMessage aMessage)`, ve které zpracujeme přijatou zprávu z FCM serveru.

Jak již bylo řečeno, nad obsahem zprávy neprovádíme žádné prokročilé zpracování, ale její obsah pouze zobrazíme formou standardní notofikace v OS Android. K tomu využijeme standardní systémovou třídu `NotificationManager`.

```
1  public class FCMService extends FirebaseMessagingService {
2
3      @Override
4      public void onMessageReceived(RemoteMessage aMsg) {
5
6          if (aMsg().size() > 0) {
7              /*
8               * Zobrazeni notifikace v OS Android
9               */
10         }
11
12     }
13
14 }
```

Zdrojový kód 9.3: Android implementace přijetí zprávy z FCM serveru.

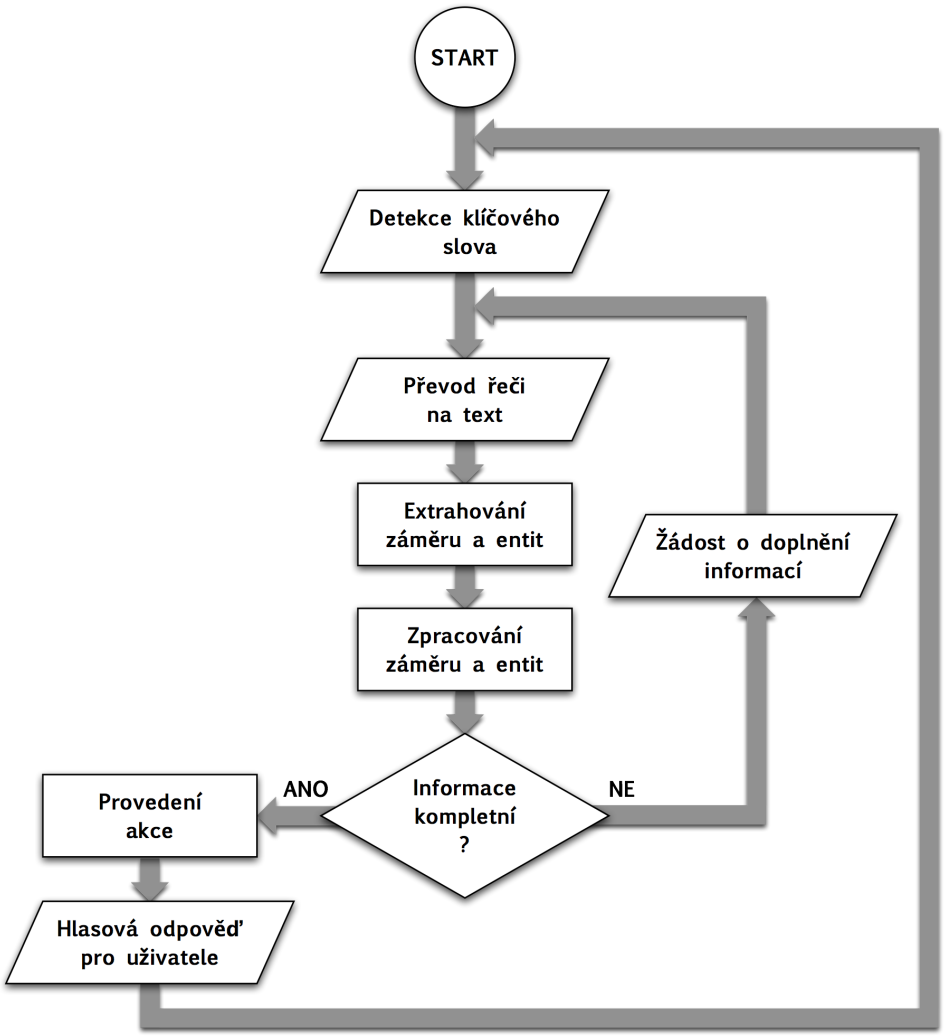
[5]Více informací o nástroji Gradle je možné nalézt zde: https://gradle.org.

[6]Více informací o filtrech záměrů naleznete v knize Android 4 Průvodce programováním mobilních aplikací [48] v kapitole Filtry záměrů.

10. Hlasové ovládání

10.1 Architektura hlasového ovládání

V této kapitole představíme hlavní přínos této práce, což je hlasové ovládání. Budeme se držet standardního schématu použitého v OS Android, kde ke spuštění hlasového zadávání slouží klíčové slovo (Ok Google). Následně je hlasový vstup uživatele převeden na text. Protože požadujeme, aby aplikace rozuměla přirozenému jazyku, musíme z převedeného textu extrahovat záměr a entity (parametry). Ty zpracujeme v kontextu inteligentní domácnosti a pokud máme všechny potřebné informace pro provedení příkazu, zajistíme jeho vykonání a poskytneme uživateli hlasovou odpověď. V případě, že informace nejsou kompletní, aplikace požádá pomocí hlasového výstupu uživatele o jejich doplnění. Princip celého hlasového ovládání je vyznačen ve vývojovém diagramu.



Obrázek 10.1: Vývojový diagram hlasového ovládání zobrazuje podstatné kroky v řetězci hlasového ovládání a jejich vzájemné provázání. Jak je patrné, jedná se o nikdy nekončící proces, který může být zastaven pouze vnějším zásahem.

10.2 Detekce klíčového slova

10.2.1 Integrace do aplikace Google Now

Prvním krokem v řetězci hlasového ovládání je detekce klíčového slova. Aplikace HomeVoice tak nebude neustále naslouchat příkazům uživatele, ale aktivuje se pouze tehdy, pokud zaznamená klíčové slovo. Tento princip úspěšně používá jak aplikace Google Now v OS Android se svou frází "Ok Google", tak konkurenční iOS s "Hey, Siri". My bychom se tohoto přístupu logicky chtěli držet také. V ideálním případě se nám podaří integrovat hlasové ovládání inteligentní domácnosti přímo do aplikace Google Now.

Abychom mohli integrovat ovládání inteligentní domácnosti do aplikace Google Now, musíme nejprve pochopit princip funkce této aplikace. Pro spuštění aplikace slouží již zmíněná fráze "Ok Google", případně je možné aplikaci Google Now spustit kliknutím na její ikonu. Následně aplikace naslouchá uživateli a převádí zachycenou řeč na text. Jakmile uživatel domluví, z textu je extrahován záměr a příslušná data. Zde opět přichází na řadu filtry záměrů, neboť o zpracování získaného záměru se postará aplikace, která má tento záměr zaregistrován (v případě více takových aplikací musí uživatel jednu vybrat).

Ačkoli dříve bylo možné nadefinovat si vlastní záměry, dnes se musíme spokojit pouze se systémovými záměry[1]. Jedná se o záměry pro ovládání širokého spektra funkcí telefonu (kalendář, fotoaparát, e-mail, hudba, mapy, atp.). Mezi záměry bohužel chybí ty, které by se hodily pro ovládání inteligentní domácnosti. Oficiálním způsobem dnes tedy není možné propojit hlasové ovládání naší intelignentí domácnosti s vestavěnou aplikací Google Now.

Během průzkumu aplikací hlasového ovládání jsme narazili na aplikaci AutoVoice[2], která umožňuje do aplikace Google Now přidat vlastní příkazy. Abychom mohli aplikaci AutoVoice používat, musíme ji nejprve povolit v nastavení přístupnosti[3]. Zřejmě tedy musí existovat nějaký neoficiální způsob, jak bychom mohli integrovat ovládání inteligentní domácnosti do Google Now, a tento způsob využívá usnadnění přístupu v OS Android.

Dle vlákna[4] na StackOverflow pracuje aplikace AutoVoice následovně:

1. Spuštění aplikace záměrem AccessibilityService (např. změna obsahu obrazovky)
2. Vyhodnocení, zda záměr vyvolala aplikace Google Now
3. Nalezení prvku na obrazovce s přepisem řeči
4. Vyčtení finálního přepisu příkazu z příslušného prvku

[1] Výpis všech standardních systémových záměrů je možné nalézt zde: https://developer.android.com/guide/components/intents-common.html.

[2] Dostupné z https://play.google.com/store/apps/details?id=com.joaomgcd.autovoice

[3] Nastavení přístupnosti slouží k usnadnění přístupu v OS Android. Je zde možné nastavit automatické předčítání textu na obrazovce, zvýšení kontrastu, zapnout titulky, atp..

[4] Custom commands for Google Now: http://stackoverflow.com/questions/38482626/custom-commands-for-google-now

5. Ukončení Google Now v případě podpory daného příkazu

Implementaci výše uvedeného postupu jsme vyzkoušeli a ověřili požadované chování. Tento přístup je opravdu možné použít pro integraci ovládání inteligentní domácnosti do aplikace Google Now. Vzhledem k tomu, že se jedná o neoficiální postup, který navíc pro svou funkci využívá fixně definované názvy balíku Google Now a prvků na obrazovce, může se tento přístup stát nefunkční s jakoukoli budoucí aktualizací OS Android. Z toho důvodu jsme se rozhodli pokračovat v hledání jiného řešení.

10.2.2 Detekce pomocí PocketSphinx

Protože oficiální integrace ovládání inteligentní domácnosti do aplikace Google Now není momentálně možná, budeme detekci klíčového slova implementovat samostatně. V dnešní době existuje velké množství knihoven a online služeb, které je možné využít pro detekci klíčového slova. Není tedy třeba implementovat zpracování zvuku za účelem detekce klíčového slova od základu, ale využijeme některý z dostupných nástrojů.

Pro výběr konkrétní knihovny jsme si stanovili několik kritérií. V první řadě chceme zajistit detekci klíčového slova i bez připojení k internetu, aby za žádných okolností nezůstala interakce uživatele bez odezvy. Vyhneme se tedy použití online služeb a knihoven, které vyžadují připojení k internetu. Dále požadujeme, aby daná knihovna byla volně dostupná, tj. bez jakýchkoli poplatků a nejlépe open-source.

Na základě průzkumu a dřívějších zkušeností jsme se rozhodli implementovat detekci klíčového slova pomocí knihovny **PocketSphinx** [49], která spadá pod projekt CMU Sphinx. Jedná se o multiplatformní open-source knihovnu psanou v jazyce C, která nabízí efektivní algoritmy pro zpracování řeči. Díky tomu je vhodná pro použití i na mobilních zařízeních s omezenými zdroji. Zpracování řeči navíc probíhá offline a tedy není potřeba připojení k internetu.

Knihovna PocketSphinx nabízí široké spektrum funkcí (převod řeči na text, ohodnocení výslovnosti, detekce klíčového slova, atd.). Pro svou práci využívá PocketSphinx tzv. jazykové modely, které obsahují gramatický, statistický a fonetický popis daného jazyka. K dispozici jsou jazykové modely pro anličtinu (britská a americká), francouzštinu, ruštinu a několik dalších. V případě potřeby je navíc možné další jazykové modely natrénovat a přidat tak například češtinu. Velikost standardního jazykového modelu je okolo 15 MB.

V našem případě předložíme knihovně PocketSphinx nejpoužívanější jazykový model, kterým je model americké anligčtiny[5]. Z hlediska nabízených funkcí nás samozřejmě nejvíc zajímá detekce klíčového slova. To můžeme sestavit z jakýchkoli slov dostupných v daném jazykovém modelu. My jsme pro testovací účely zvolili aktivační frázi **"Ok Smart Home"**.

[5]Aktuální verze všech oficiálních jazykových modelů je možné stáhnout zde: https://sourceforge.net/projects/cmusphinx/files/Acoustic%20and%20Language%20Models/

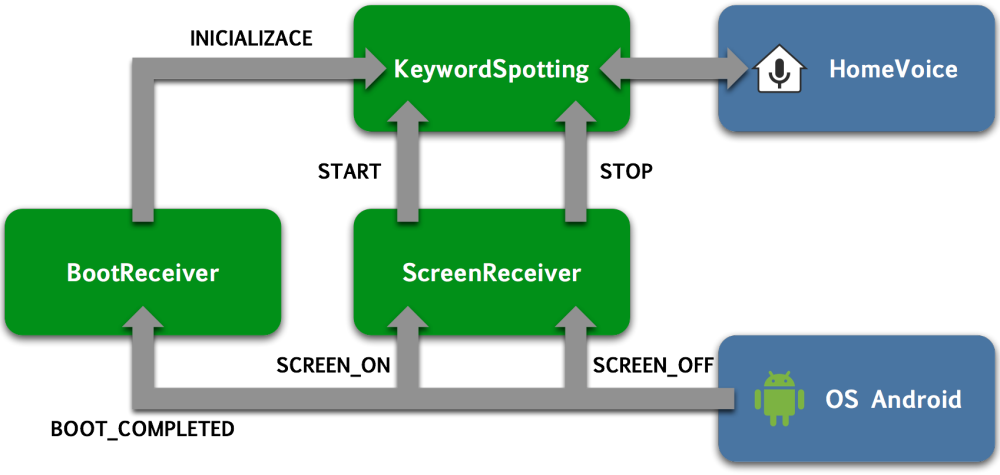
10.2.3 Implementace PocketSphinx v OS Android

Použití knihovny PocketSphinx v rámci našeho projektu nám usnadní dodávaná ukázková aplikace[6], která prezentuje použití detekce klíčového slova, a převod řeči na text v kontextu číslic a počasí. Tato aplikace využívá mikrofon vestavěný v zařízení.

Ukázková implementace knihovny PocketSphinx pro OS Android představuje jednoduchou aplikaci s GUI (tzv. Activity), což pro naše účely není příliš vhodné. Cílem je navrhnout detekci klíčového slova jako službu na pozadí, která poběží nezávisle na aplikaci HomeVoice. Z důvodu úspory elektrické energie zároveň požadujeme, aby detekce klíčového slova byla aktivní pouze tehdy, pokud je rozsvícena obrazovka zařízení. Aby služba pro detekci klíčového slova byla vždy připravena, musíme ještě zajistit její automatické spuštění po startu mobilního zařízení.

Implementace služby pro detekci klíčového slova (`KeywordSpotting`) vychází z příkladu v ukázkové aplikaci. Samotnou detekci tak nebudeme dopodrobna rozebírat, ale zaměříme se na propojení této služby se systémem Android a aplikací HomeVoice.

Pro spuštění služby `KeywordSpotting` při zapnutí zařízení a její následnou aktivaci a deaktivaci podle stavu obrazovky použijeme tzv. Broadcast receivery. Ty umožňují odchytit a zpracovat události generované systémem. V našem případě implementujeme `BootReceiver` registrovaný pro událost `BOOT_COMPLETED`[7], která značí načtení operačního systému. Zapnutí a vypnutí obrazovky zařízení signalizuje událost `SCREEN_ON`[8], resp. `SCREEN_OFF`[9]. Přes filtr záměrů si tyto události zaregistruje `ScreenReceiver`. Napojení Broadcast receiverů na službu `KeywordSpotting` zachycuje obrázek (10.3).



Obrázek 10.2: Schéma vzájemného propojení služby `KeywordSpotting` s OS Android a aplikací hlasového ovládání HomeVoice. Inicializaci služby zajistí `BootReceiver` při zapnutí systému. O aktivaci/deaktivaci se stará `ScreenReceiver` při zapnutí, resp. vypnutí obrazovky. Po detekci klíčového slova spustí služba `KeywordSpotting` aplikaci HomeVoice.

[6] Dostupná z https://github.com/cmusphinx/pocketsphinx-android-demo.
[7] Úplný název v OS Android: `android.intent.action.BOOT_COMPLETED`
[8] Úplný název v OS Android: `android.intent.action.SCREEN_ON`
[9] Úplný název v OS Android: `android.intent.action.SCREEN_OFF`

Jakmile služba `KeywordSpotting` detekuje frázi **"Ok Smart Home"**, zajistí spuštění aplikace HomeVoice prostřednictvím příslušného záměru. V parametru tohoto záměru současně přenese informaci o detekci klíčového slova, na základě které aktivuje aplikace HomeVoice ihned po startu převod řeči na text. Služba `KeywordSpotting` se tímto okamžikem deaktivuje a o opětovnou aktivaci se postará až aplikace HomeVoice.

10.3 Převod řeči na text

10.3.1 Požadavky

Po detekci klíčového slova přichází na řadu převod řeči na text. V této fázi je již aplikace HomeVoice spuštěna a očekává hlasový vstup od uživatele. Typicky se jedná o jednoduchý příkaz, či dotaz trvající v řádu několika sekund. Jakmile uživatel domluví, je třeba tento hlasový vstup převést na text. To umožní jeho další zpracování. Z uvedeného vyplývá, že musíme vyřešit nejen nahrávání zvuku a převod řeči na text, ale také detekci ukončení hlasového zadávání, jakmile uživatel přestane mluvit.

Stejně jako v případě detekce klíčového slova, tak ani zde nemusíme veškeré funkce navrhovat a implementovat od základu, ale využijeme některé z existujících knihoven, či služeb. Opět požadujeme, aby se jednalo o volně dostupné řešení, nejlépe open-source, které budeme moci použít bez dodatečných nákladů. Abychom mohli interagovat s uživatelem i bez připojení k internetu, preferujeme knihovny a služby pracující v offline režimu. Na rozdíl od detekce klíčového slova zde ovšem netrváme na offline režimu za každou cenu, neboť i aplikace Google Now v některých případech vyžaduje připojení k internetu. Velký důraz klademe také na spolehlivost detekce a hlavně dosaženou přesnost převodu řeči na text (tj. co nejnižší míru chybovosti).

10.3.2 Rozpoznání řeči pomocí PockeSphinx

Nabízí se využití knihovny **PocketSphinx**, která je open-source, pracuje v offline režimu a již je součástí projektu HomeVoice (včetně hlasového modelu). Na základě předchozí zkušenosti s přesností převodu řeči na text jsme se rozhodli tuto knihovnu nepoužít. V rámci projektu Household Intelligent Assistant[10], který předcházel této diplomové práci, podávala knihovna **PocketSphinx** velmi nestabilní výkony, navíc silně závislé na použitém mikrofonu. Naši zkušenost potvrzují také výsledky měření uvedené v [50], kde naměřili v průměru 46,4% chybně rozpoznaných slov. Systém pro rozpoznání řeči od společnosti Google v průměru chybně určil pouze 10,6% slov.

10.3.3 Rozpoznání řeči pomocí RecognitionService

Vzhledem k tomu, že jsme zamítli použití knihovny **PocketSphinx** pro převod řeči na text, musíme nalézt jiné řešení. Rozhodli jsme se podívat, jaké možnosti pro převod řeči na text nabízí společnost Google, neboť dle výše zmíněného průzkumu dosahují tyto systémy malé míry chybovosti. Systém pro rozpoznání řeči od společnosti Google je navíc aktivně používán v rámci služeb a produktů této společnosti, což zaručuje

[10] Video projektu Household Intelligent Assistant: https://www.youtube.com/watch?v=xOZluGa_lwc

jeho neustálé testování, údržbu a vývoj. Kromě služby **Google Cloud Speech API**[11] (online, placená - $0.006 za 15 sekund zpracovaného audio záznamu) přichází v úvahu služba **RecognitionService**[12].

Jedná se o službu určenou pro převod řeči na text, která je součástí OS Android (od API verze 8). Defaultně vyžaduje služba **RecognitionService** připojení k internetu. Pokud ji chceme používat offline, musíme do Android zařízení stáhnout příslušný jazykový model[13]. Možnost offline rozpoznání řeči bohužel nepodporují všechna zařízení a obecně tedy předpokládáme nutnost připojení zařízení k internetu. Jak již bylo řečeno, tento nedostatek jsme ochotni tolerovat, neboť v těchto případech vyžaduje připojení k internetu i aplikace Google Now.

Služba **RecognitionService** nabízí při převodu řeči na text dva různé přístupy.

1. **RecognizerIntent**: Pro spuštění služby RecognitionService z aplikace slouží vždy záměr `RecognizerIntent`. Pokud zavoláme tento záměr standardním způsobem (tj. přes `startActivityForResult()`[14]), objeví se systémové dialogové okno, které informuje uživatele o aktivním hlasovém vstupu. Jakmile uživatel domluví, řeč je automaticky převedena na text a ten je navrácen zpět do původní aplikace přes metodu `onActivityResult()`.

2. **SpeechRecognizer**: I zde slouží pro spuštění služby RecognitionService záměr `RecognizerIntent`. V tomto případě je ale tento záměr volán přes instanci třídy `SpeechRecognizer`. Díky tomu se vyhneme zobrazení systémového dialogového okna, takže z pohledu uživatele se zdá, že převod řeči na text zajišťuje přímo aplikace. Přes `SpeechRecognizer` je navíc možné monitorovat převod řeči na text průběžně.

Rozhodli jsme se použít druhý z uvedených přístupů, tj. s vytvořením instance třídy `SpeechRecognizer`. Zpracování řeči tak zdánlivě probíhá přímo v naší aplikaci a uživateli můžeme postupně zobrazovat přepis řeči tak, jak je zrovna zachycen v nahrávaném zvuku. Při převodu řeči opět předpokládáme použití americké angličtiny.

10.3.4 Implemenetace RecognitionService v OS Android

V aplikaci HomeVoice implementujeme převod řeči na text v samostatné třídě zvané `SpeechRecognition`. Dochází zde k inicializaci instance třídy `SpeechRecognizer` a prostřednictvím metody `startRecognition()` je zavolán záměr `RecognitionIntent` a tím zahájen proces převodu řeči na text. Přenos finálního textu i průběžných výsledků zpět do hlavní aktivity aplikace HomeVoice zajišťuje `OnSpeechRecognitionListener`.

[11] Více informací je možné nalézt zde: https://cloud.google.com/speech/

[12] Dostupné z https://developer.android.com/reference/android/speech/RecognitionService.html

[13] Podrobný návod, jak používat převod řeči na text bez připojení k internetu, je možné nalézt zde: http://stackandroid.com/tutorial/how-to-enable-offline-speech-to-text-in-android/

[14] Více informací o spouštění aktivity pomocí záměru a přenosu dat (výsledků) je možné nalézt zde: https://developer.android.com/training/basics/intents/result.html

10.4 Extrahování záměru a entit

10.4.1 Princip funkce

Z předchozího kroku máme k dispozici vstupní data od uživatele ve formě přepisu jeho řeči. Nyní potřebujeme určit, co tímto vstupem uživatel zamýšlel, tj. jaký je význam získaného přepisu. Nechceme se omezovat pouze na konkrétní sadu příkazů a dotazů, ale předpokládáme obecný vstup v přirozeném jazyce. Přepis řeči samozřejmě nemusí být dokonalý a některá slova mohou být rozpoznána chybně. Potřebujeme tedy navrhnout dostatečně robustní zpracování.

Zde přichází na řadu extrahování záměru (značíme `INTENT`) z přepisu řeči, tedy nalezení úmyslu uživatele. V kontextu vyvíjené aplikace si pod pojmem záměr můžeme představit označení konkrétní akce, např. `SET_LIGHT`, či `GET_TEMPERATURE`. Pokud dokážeme určit záměr, můžeme na něj náležitým způsovem reagovat. V případě záměru `SET_LIGHT` tak zajistíme rozsvícení/zhasnutí světla, výčet teploty a její předání uživateli provedeme při rozpoznání záměru `GET_TEMPERATURE`.

Kromě záměru jsou pro další zpracování podstatné také tzv. entity, které mohou být obsažené v přepisu řeči. Jedná se o parametry blíže specifikující úmysl uživatele. V případě záměru `SET_LIGHT` například potřebujeme vědět, zda chce uživatel světlo rozsvítit, či zhasnout. Za tímto účelem si nadefinujeme entitu `ON_OFF`, která nabývá dvou hodnot (`ON`, `OFF`). Pokud se naše inteligentní domácnost skládá z více místností, chtěli bychom umožnit ovládání světla v konkrétní místnosti. Nabízí se definování entity `LOCATION`, jejíž obsahem je textový řetězec nesoucí název konkrétní místnosti/lokace.

Extrahování záměru a entit z přepisu řeči si ukážeme na konkrétním příkladu ovládání inteligentní domácnosti. Předpokládejme, že chce uživatel rozsvíti světlo umístěné v ložnici. Bez ohledu na použitý jazyk (čeština, angličtina, atd.) existuje mnoho způsobů, jak tento úmysl může vyjádřit:

1. Turn on the light in bedroom.
2. Could you turn the light on in bedroom?
3. Turn the bedroom light on, please.

Ať už uživatel použije kteroukoli z výše uvedených vět, vždy očekává, že dojde k rozsvícení světla v ložnici. V každém z těchto případů bychom tak měli dostat stejný záměr (`INTENT`) a stejný seznam entit, na základě čehož budeme schopni zajistit provedení uživatelem požadované akce.

`INTENT`	`SET_LIGHT`
`ON_OFF`	`ON`
`LOCATION`	`bedroom`

Jak je již určitě zřejmé, každá akce v inteligentní domácnosti vyžaduje definování záměru, který ji jednoznačně rozlišuje od ostatních. Musíme také vytvořit množinu všech entit a jejich možných hodnot, pomocí kterých budeme jednotlivé záměry parametrizovat. Seznam záměrů a entit použitých v aplikaci HomeVoice obsahuje příloha C.

10.4.2 Extrahování záměru a entit pomocí Wit.ai

Mechanismus pro extrahování záměru a entit nebudeme navrhovat vlastními silami, neboť se jedná o problematiku, která by svým rozsahem pokryla samostatnou práci. Místo toho přistoupíme k integraci některého z existujících nástrojů do aplikaci HomeVoice. Stejně jako v případě detekce klíčového slova a převodu řeči na text, tak i zde požadujeme volně dostupný nástroj (knihovna, služba), který můžeme použít bez poplatků. Již jsme slevili z nároku, aby všechny komponenty aplikace HomeVoice pracovaly v offline režimu s odůvodněním, že ani aplikace Google Now nedokáže za každé situace pracovat bez připojení k internetu.

Provedli jsme průzkum dostupných nástrojů a vybrali 3 služby, na které se blíže podíváme. Jedná se o službu **LUIS**[15] od společnosti Microsoft, **Api.ai**[16] od společnosti Google a **Wit.ai**[17] ve vlastnictví společnosti Facebook. Všechny tyto služby nabízejí funkce pro zpracování přirozeného jazyka (extrahování záměru a entit) a jsou používány při realizaci konverzace s uživatelem. Ačkoli jsou všechny uvedené služby velmi podobné, několik rozdílu přeci jenom najdeme.

Co se týče ceny, nejlépe si vede služba **Wit.ai**, která je dostupná zcela zdarma pro komerční i nekomerční účely. Stejně je na tom i služba **Api.ai**. Nejhůře dopadla služba **LUIS**, která je k dispozici zdarma pouze do limitu 10 tisíc volání měsíčně. To je pro testování dostatečná hodnota, ale při komerčním nasazení by bylo třeba zvolit placenou variantu ($0,75 za 1000 volání). Z dalšího výběru tedy **LUIS** vyloučíme.

Z hlediska výkonu jsou na tom služby **Wit.ai** a **Api.ai** velmi podobně. Praktické srovnání těchto služeb poskytuje [51]. Obě služby také nabízí Android SDK[18], což usnadňuje jejich použití v rámci aplikace HomeVoice. Z pohledu těchto kritérií je tedy poměrně jedno, zda zvolíme **Wit.ai**, či **Api.ai**.

Přestože žádná z uvedených služeb není výrazně lepší, rozhodli jsme se použít **Wit.ai**. S touto službou jsme se již seznámili v rámci projektu Household Intelligent Assistant, což nám umožní lépe využít její potenciál.

10.4.3 Trénování služby Wit.ai

Prvním krokem při integraci nástroje Wit.ai do aplikace HomeVoice je registrace na stránkách této služby, založení příslušného projektu a hlavně natrénování extrakce záměru a entit v konkrétní doméně (ovládání inteligentní domácnosti). Pro registraci do služby Wit.ai je potřeba GitHub, či Facebook účet. Abychom mohli založit projekt, musíme zadat jeho název, vybrat jazyk (v našem případě angličtina) a zvolit, zda chceme natrénovaný model zveřejnit, nebo ponechat uzavřený. Po založení je automaticky vygenerován klíč **Server Access Token**, prostřednictvím kterého budeme později autorizovat jednotlivá volání.

[15] Více informací o službě LUIS je k dispozici zde: https://www.luis.ai/.

[16] Služba Api.ai je dostupní prostřednictvím https://api.ai.

[17] Zde se nachází stránky služby Wit.ai: https://wit.ai.

[18] SDK, zkratka z anglického výrazu Software Development Kit, představuje obvykle sadu nástrojů pro vývojaře, které umožňují vývoj určitého typu softwaru.

Než přistoupíme k natrénování služby Wit.ai, nadefinujeme nejprve záměry a entity uvedené v příloze C. Nemusíme definovat všechny uvedené entity, neboť mnoho z nich je již ve službě Wit.ai předpřipravených. Wit.ai tak například automaticky rozpozná v předloženém textu lokaci (entita `location`), volbu zapnuto/vypnuto (entita `on_off`), číslo (entita `number`) a mnoho dalších.

V případě námi definovaných entit musíme zvolit, jakým způsobem se má daná entita z textu extrahovat. Na výběr je ze tří možností, které se navíc mohou vzájemně kombinovat.

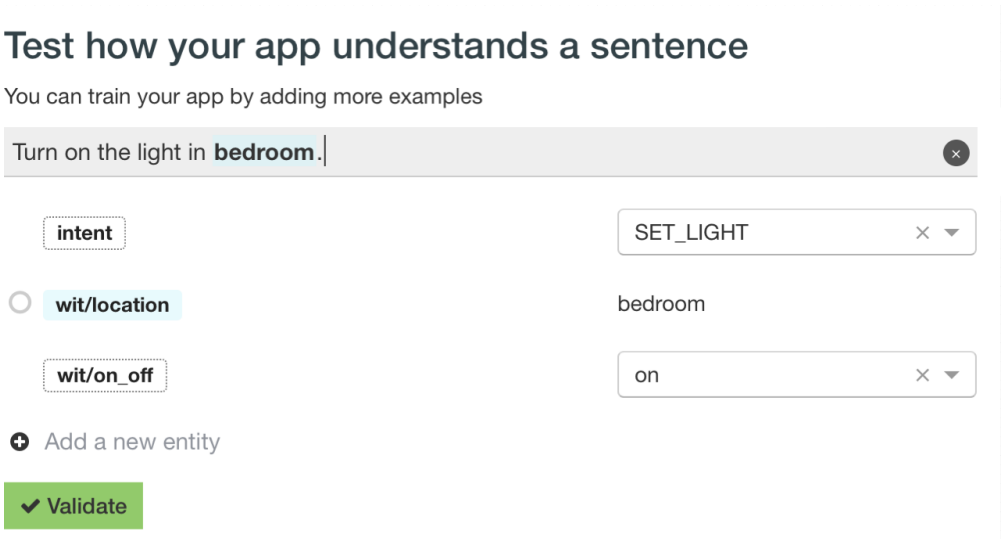
1. **keywords**: Daná entita může nabývat pouze hodnot z předem definovaného seznamu slov. Tato strategie se hodí například při extrahování názvu státu.

2. **free-text**: Pro extrakci příslušné entity je využíváno vzájemného vztahu slov ve větě. Příkladem použití této strategie je extrakce názvu ve větě.

3. **trait**: Strategie nepoužívá pro extrahování entity seznam slov, ani vzájemných asociací mezi slovy, ale pohlíží na daný text jako celek. Tento přístup je vhodný například pro určení nálady uživatele.

Ačkoli z popisu jednotlivých strategií je poměrně jasné, v jakých situacích tu kterou strategii zvolit, v praxi se vyplatí jednotlivé strategie otestovat a vybrat podle nejlepších výsledků. Námi použitá nastavení jsou taktéž uvedena v příloze C.

Jakmile jsou všechny záměry a entity nadefinovány, můžeme se pustit do samotného trénování. K tomu slouží formulář, jež je součástí webového rozhraní služby Wit.ai. Věty, které chceme Wit.ai naučit, zadáváme postupně do příslušného pole. Wit.ai se pro jednotlivé věty pokusí určit záměr a extrahovat entity. Uživatel následně provede potřebné korekce. Seznam trénovacích vět je uveden v příloze D.



Obrázek 10.3: Rozhraní pro trénování služby Wit.ai. Pro zadanou text se Wit.ai snaží automaticky učit záměr a obsažené entity. Na uživateli je kontrola a případná úprava (změna záměru, přidání entit, atp.). Jakmile uživatel klikne na tlačítko Validate, dojde k přetrénování naučeného modelu a během pár sekund je možné ho používat.

10.4.4 Implementace Wit.ai v OS Android

Službu Wit.ai jsme natrénovali a nyní nastal čas integrovat ji do aplikace HomeVoice. Původně jsme zamýšleli využít pro integraci Wit Android SDK[19]. Dle popisu je toto SDK určeno pro experimentální účely a není doporučeno jeho komerční použití. Další možností, jak komunikovat se službou Wit.ai, je použití standardních HTTP volání. Z výše uvedených důvodů jsme se rozhodli integrovat Wit.ai do aplikace HomeVoice s využitím HTTP API.

Pro extrahování záměru a entit je třeba zaslat požadovaný text pomocí HTTP metody GET na níže uvedenou URL. Z důvodu autorizace volání nesmíme opomenout přidání klíče **Server Access Token** do hlavičky HTTP volání.

`https://api.wit.ai/message`

Strukturu odpovědi ve formátu JSON ukážeme na konkrétním příkladu.

```
{
    "msg_id":"0EJP3WxW8sJwu1XCi",
    "_text":"Turn on the lights in bedroom",
    "entities":{
        "location":[{
            "suggested":true,
            "confidence":0.99820353956834,
            "value":"bedroom",
            "type":"value"
        }],
        "intent":[{
            "confidence":0.99905166516171,
            "value":"SET_LIGHT"
        }],
        "on_off":[{
            "confidence":0.97320705333132,
            "value":"on"
        }]
    }
}
```

Zdrojový kód 10.1: Příklad odpovědi ze služby Wit.ai pro větu "Turn on the lights in bedroom". Dle očekávání je v odpovědi zahrnut záměr `SET_LIGHT` a dvojice entit `location` a `on_off`. Za povšimnutí stojí také parametr `confidence`, který značí míru jistoty extrahování.

I zde bychom mohli využít REST klienta Retrofit, který již v rámci aplikace HomeVoice zajišťuje komunikaci s řídicím softwarem Z-Way. V případě služby Wit.ai jsme tohoto klienta nepoužili, neboť integrace této služby do aplikace HomeVoice předcházela implementaci komunikace s řídicí jednotkou a tehdy jsme ještě nebyli blíže seznámeni s REST klientem Retrofit. HTTP volání a převod JSON odpovědí na příslušné datové struktury realizujeme pomocí vlastního klienta.

[19]Dostupné z https://github.com/wit-ai/wit-android-sdk.

10.5 Zpracování záměru a entit

10.5.1 Úvod

Další krok při realizaci hlasového ovládání představuje zpracování záměru a entit, které jsme extrahovali v rámci předchozí kapitoly. Ačkoli již známe úmysl uživatele (záměr), zatím netušíme, zda je možné zajistit patřičnou reakci (provedení akce, odpověď na dotaz). Abychom tak mohli učinit, je třeba zpracovat extrahovaný záměr a entity v kontextu konkrétní inteligentní domácnosti. Musíme určit, zda máme k dispozici všechny potřebné informace, tzn. zda uživatel přesně specifikoval, co chce provést (např. rozsvícení konkrétního světla). Pokud tyto informace nemáme, musíme uživatele náležitým způsobem požádat o jejich doplnění. Současně bychom v této fázi měli rozhodnout, zda je možné požadovaný záměr uživatele splnit (např. pokud uživatele zajímá teplota, ale inteligentní domácnost neobsahuje žádný senzor teploty). Opět je třeba vygenerovat patřičnou odpověď pro uživatele.

Vzhledem k tomu, že je implementace zpracování záměru a entit silně závislá na konkrétní relizaci inteligentní domácnosti (v našem případě řídicí software Z-Way), musíme navrhnout vlastní řešení. Ovládané a monitorované funkce inteligentní domácnosti rozdělíme do skupin (ovládání světel, monitorování teploty, atp.) a tyto skupiny naimplementujeme jako samostatné moduly.

10.5.2 Kontext modulu

Každý z modulů jsme založili na tzv. kontextu, v rámci kterého jsou udržovány informace získané od uživatele prostřednictvím záměru a entit. Hlavním cílem kontextu je uložení těchto informací a následné poskytnutí těchto informací během fáze zpracování kontextu. Životní cyklus kontextu modulu se skládá ze tří fází.

1. **inicializace**: Jakmile uživatel zahájí interakci s aplikací HomeVoice, v každém z modulů dojde k inicializaci kontextu pomocí extrahovaného záměru a entit. Ve stejný okamžik jsou vytvářeny kontexty ve všech dostupných modulech.

2. **aktualizace**: Pokud nejsou informaci získané během inicializace kompletní, je třeba vést s uživatelem dialog a aktualizovat příslušný kontext. Aktualizace kontextu probíhá pouze v modulu, který zažádal o doplňující informace.

3. **smazání**: Ke smazání všech vytvořených kontextů dochází, jakmile je ukončen dialog s uživatelem, tj. buď došlo k provedení požadované akce, nebo není možné v dialogu dále pokračovat.

Jak je patrné, kontext daného modulu vzniká krátce potom, co uživatel osloví aplikaci HomeVoice pomocí klíčového slova "Ok Smart Home". Existence všech kontextů je omezena pouze na dobu vedení dialogu, tj. dokud nepřejde aplikace zpět do režimu detekce klíčového slova. Pomocí vedení dialogu je kontext aktualizován a tím získává aplikace více informací.

Instance jednotlivých kontextů jsou na sobě nezávislé a mají pevnou vazbu na konkrétní modul. Nezbytnou součástí každého kontextu je záměr, tj. úmysl uživatele. Díky této

informaci můžeme později rozhodnout, zda jednotlivé moduly dokáží zpracovat daný kontext. Další položky v kontextu se odvíjí od konkrétního modulu, resp. od skupiny modulů. Jako příklad si uvedeme kontext modulů, které se starají o ovládání a monitorování inteligentních zařízení (`LightModule`, `TemperatureModule`, `HumidityModule`, `IlluminationModule`,...).

1. **záměr**: Úmysl uživatele, např. `GET_TEMPERATURE`, nebo `SET_LIGHT`, představuje základ každého kontextu.
2. **hodnota**: Umožňuje zvolený záměr doplnit o hodnotu. Pro záměr `SET_LIGHT` se například jedná o jednu z hodnot `ON/OFF`, číslo, nebo barvu.
3. **dotaz**: V případě dotazů se uživatel může ptát na hodnotu `VALUE`, počet zařízení `COUNT`, či seznam zařízení `LIST`.
4. **seznam lokací**: Specifikace místností (obecně oblastí), rámci kterých má být požadovaná akce provedena (např. `bedroom`, `garden`).
5. **seznam názvů**: List názvů zařízení (`green light`) umožňuje v rámci inteligentní domácnosti rozlišit konkrétní zařízení.
6. **seznam typů**: Definice typů zařízení, které jsou podporovány pro daný záměr (`binarySwitch`, `multilevelSwitch`,...).

Většina hodnot je nastavena přímo s využitím extrahovaného záměru a entit (záměr, hodnota, dotaz, lokace, názvy), což znamená, že jsou tyto entity uloženy do příslušného kontextu bez dalšího zpracování. Z uvedených hodnot je pouze **seznam typů** nastaven nepřímo. Tento seznam je generován na základě záměru a hodnoty. Například pro záměr `SET_LIGHT` a hodnotu `ON` se bude jistě jednat o zařízení typu `binarySwitch` a `multilevelSwitch`.

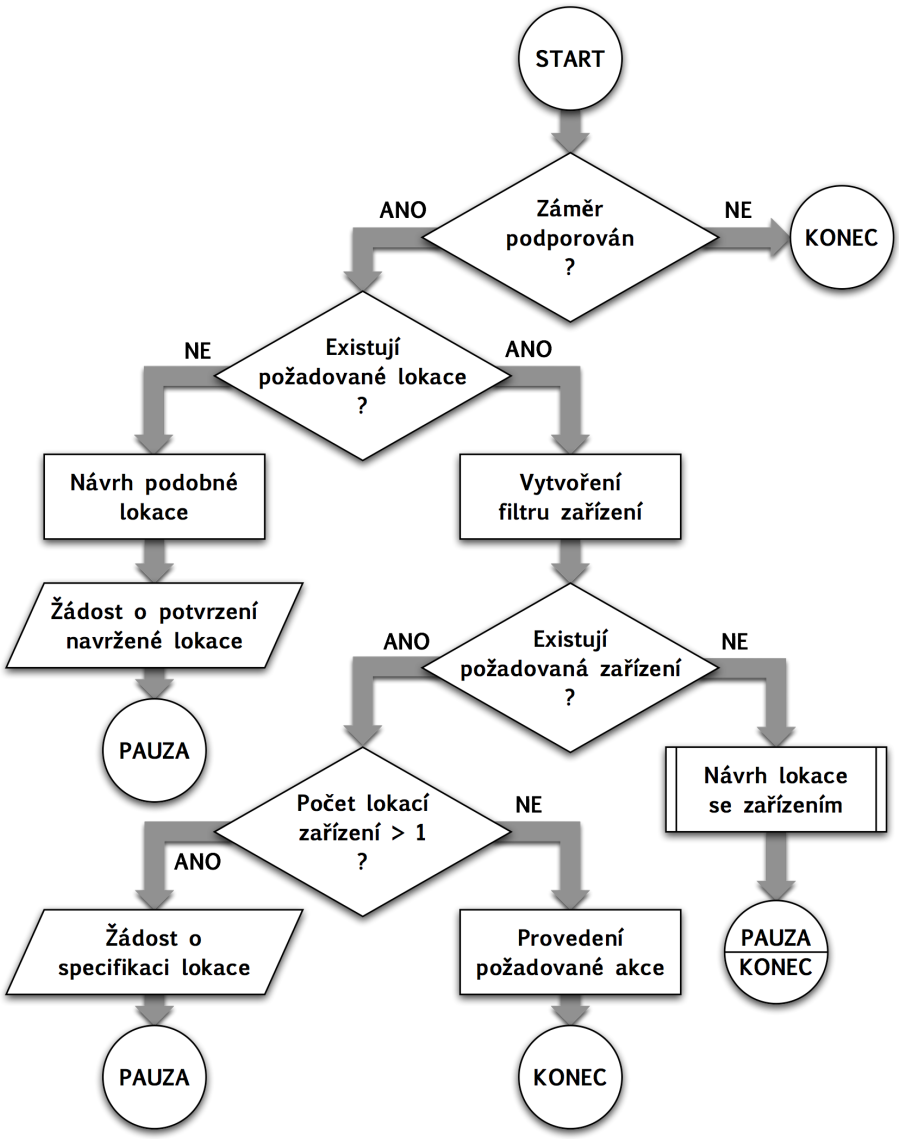
10.5.3 Logika zpracování kontextu

Ke zpracování kontextu dochází vždy po jeho inicializaci, či aktualizaci. Kdykoli tedy uživatel vznese požadavek, upraví staré informace, případně doplní nové, dojde k novému zpracování kontextu. V této fázi se plánují jednotlivé kroky v dialogu s uživatelem a určuje se další postup. Jedná se o klíčovou část hlasového ovládání, která vytváří pocit inteligentního asistenta na straně aplikace HomeVoice.

V rámci tohoto zpracování se rozhodne, zda je možné provést požadovanou akci, tj. jestli máme k dispozici všechny potřebné informace a jestli je vůbec možné požadavek uživatele splnit. Na základě těchto zjištění jsou provedeny patřičné reakce, ať už ve formě dotazu o doplnění informací, žádosti o potvrzení navržených informací, nebo realizace konkrétní akce.

Logika zpracování kontextu se obecně může lišit pro různé moduly, ale prakticky je velmi podobná, či dokonce stejná pro určité kategorie modulů. Ať už se uživatel ptá na teplotu, nebo vlhkost, v obou případech se bude jednat o stejný způsob vedení dialogu (rozdíly nalezneme pouze v názvech veličin). Můžeme říci, že velmi podobný dialog bude správně fungovat i v případě ovládání zařízení.

Níže je uvedena konkrétní podoba zpracování kontextu modulů, které zajišťují ovládání a monitorování inteligentních zařízení. Snažili jsme se o návrh dostatečně robustního mechanismu, který bude beze změny pracovat pro senzory i aktuátory, pro inicializaci i aktualizaci kontextu.



Obrázek 10.4: Vývojový diagram zpracování kontextu v rámci modulů, které zajišťují ovládání a monitorování zařízení inteligentní domácnosti. V prvním kroku se rozhoduje, zda je požadovaný záměr podporován příslušných modulem. Pokud ne, modul ukončí zpracování kontextu a přenechá ho dalším modulům. Dále se kontroluje, zda je požadavek v pořádku, tj. jestli existují lokace uvedené v požadavku. Pokud ano, jsou vybrána zařízení, nad kterými je možné provést požadovanou akci. Další krok závisí na tom, kolik takových zařízení existuje. Jestliže žádné zařízení neodpovídá požadavkům, pokusí se systém navrhnout např. zařízení v jiné lokaci. Nakopak, pokud je zařízení více a nachází se v různých lokacích, je na uživateli specifikace konkrétní lokace.

10.6 Provedení akce

10.6.1 Úvod

Pokud proběhlo zpracování kontextu úspěšně, tj. máme všechny potřebné informace a úmysl uživatele je možné splnit, nastal čas na provedení příslušné akce. O realizaci se v tomto případě stará modul, který dříve zpracoval kontext. Ačkoli se obecně nemusí jednat pouze o akce související s inteligentní domácností (např. dotaz na čas, počasí, atp.), v této práci se zaměříme pouze na ovládání a monitorování zařízení v inteligentní domácnosti, konkrétně ovládání světel a monitorování teploty a vlhkosti.

Abychom mohli požadovanou akci provést, neobejdeme se bez reference (ve formě identifikátoru) na konkrétní zařízení. Tu jsme naštěstí získali v rámci předchozího kroku (zpracování záměru a entit). Následně se aplikace HomeVoice spojí s řídicí jednotkou a prostřednictvím vDev API zajistí provedení požadované akce (získání hodnoty ze senzoru, či nastavení hodnoty aktuátoru). Většina kódu zajišťujícího provedení akce je převzatá z ukázkové aplikace Z-Way-Android.

10.6.2 Získání hodnoty ze zařízení

Pro vyčtení hodnoty (stavu) z konkrétního zařízení je třeba zažádat řídicí jednotku o popis tohoto zařízení. Při použití vDev API a známém identifikátoru (`id`) je možné popis zařízení získat ve formátu JSON pomocí následujícího příkazu:

```
/ZAutomation/api/v1/devices/<id>
```

V námi navržené aplikaci výše uvedený postup není potřeba, neboť máme k dispozici datový model všech zařízení na straně aplikace HomeVoice. Tento model je navíc aktualizován na pozadí aplikace s periodou 1 sekunda, takže máme neustále k dispozici aktuální data (maximálně jednu sekundu stará).

Pro odečtení hodnoty z příslušného zařízení tak stačí najít dle získaného indentifikátoru model zařízení a z něj vyčíst požadovaná data. Výhodou tohoto přístupu je rychlost získání dat, neboť není třeba při každém dotazu navazovat spojení s řídicí jednotkou.

10.6.3 Nastavení hodnoty aktuátoru

V případě nastavení hodnoty zařízení typu aktuátor se již nevyhneme přímé komunikaci s řídicí jednotkou. Pro ovládání zařízení opět použijeme vDev API, které dispnuje sadou příkazů určených pro tento účel. Jedná se vždy o stejné URL, které se liší identifikátorem zařízení, příkazem a sadou argumentů. Seznam podporovaných příkazů je uveden v příloze B.

```
/ZAutomation/api/v1/devices/<id>/command/<příkaz>
```

Implementace v naší aplikaci používá REST klienta, kterého jsme vytvořili během autorizace přístupu k řídicí jednotce. Tento REST klient má tedy uloženy veškeré autorizační klíče ve formě Cookies a není třeba ho znovu autorizovat. Následně při realizaci příslušného volání stačí zadat identifikátor zařízení a požadovanou hodnotu.

10.7 Syntéza řeči

10.7.1 Pořadavky

Poslední krok řetězce hlasového ovládání představuje syntéza řeči. Jedná se o protiklad rozpoznání řeči, kde naopak dochází k převodu textu na řeč. Díky tomu může aplikace předávat informace uživateli prostřednictvím hlasu. Syntéza řeči najde uplatnění jak při odpovídání na dotazy/příkazy uživatele, tak při žádosti o doplnění informací a schválení návrhu.

Cílem této práce není návrh vlastního systému pro syntézu řeči. Místo toho se zaměříme na existující řešení (knihovny, služby), které pro zadaný text vygenerují zvuk řeči. Opět požadujeme, aby byla použitá knihovna, či služba volně dostupná (bez poplatků). Jako výhodu vidíme schopnost vybraného řešení pracovat bez připojení k internetu, tj. v offline režimu. Na této podmínce ale netrváme, neboť extrahování záměru a v některých případech i převod řeči na text pracuje pouze online.

10.7.2 Syntéza řeči pomocí TextToSpeech

Při hledání řešení pro syntézu řeči nemusíme chodit daleko, neboť se nabízí použití knihovny **TextToSpeech**, která je součástí Android SDK od API verze 4. Jedná se o volně dostupnou knihovnu s podporou více než 30-ti jazyků (včetně češtiny). Jednotlivé jazykové modely je navíc možné stáhnout do zařízení a používat tak syntézu řeči i bez připojení k internetu. U vybraných jazyků je možné dále volit mezi mužským a ženským hlasem.

Protože knihovna **TextToSpeech** splňuje všechny výše uvedené požadavky, rozhodli jsme se ji použít pro realizaci převodu textu na řeč. Knihovna navíc disponuje velmi jednoduchým rozhraním, což ji předurčuje pro rychlé nasazení v rámci Android aplikací.

10.7.3 Implementace TextToSpeech v OS Android

Jak již bylo řečeno, použití TextToSpeech v aplikacích pro OS Android je velmi jednoduché. Základ implementace tvoří incializace, v rámci které nastavíme požadovaný výstupní jazyk (v našem případě americká angličtina). Následně již stačí pomocí metody `speak()` přidávat do fronty text, který má být převeden na řeč. Syntetizovaná řeč je automaticky přehrána prostřednictvím vestavěných reproduktorů v mobilním zařízení.

V aplikaci HomeVoice chceme navíc sledovat průběh převodu textu na řeč, abychom ho mohli synchronizovat s textem vypisovaným na obrazovku. K tomu slouží posluchač `UtteranceProgressListener`, který umožňuje detekovat začátek převodu, konec převodu a případné chyby.

11. Testovací domácnost

11.1 Výběr testovacích zařízení

Inteligentní domácnost by nenabízela příliš funkcí, kdybychom k řídicí jednotce nepřipojili různé senzory a aktuátory. Abychom mohli otestovat hlasové ovládání pro co nejširší spektrum funkcí, musíme vybrat vhodné zastoupení inteligentních zařízení. Díky zaručené kompatibilitě všech Z-Wave vyrobených zařízení máme opravdu velký výběr. V případě ekosystému Z-Wave se navíc jedná o hotové řešení, kde jednotlivá zařízení jsou připravena k použití ihned po vybalení z krabice. Nevyžadují tedy žádnou dodatečnou úpravu, ale před pvním použitím je stačí pouze přidat do Z-Wave sítě.

11.2 Aeon Labs MultiSensor 6

Nezbytnou součástí inteligentní domácnosti jsou senzory, tj. zařízení měřící různé fyzikální veličiny. V případě domácnosti se nabízí měření teploty, vlhkosti, či třeba úrovně osvětlení. Všechny tyto veličiny a k tomu i pár dalších měří zařízení MultiSensor 6[1] od společnosti Aeon Labs. Jak je patrné z obrázku, jedná se o malé zařízení, které v sobě kombinuje několik různých senzorů a vysílač Z-Wave. Kromě měření teploty (v rozsahu -10 až +50°C s přesností ±0,5°C), vlhkosti (v rozsahu 20% až 80% RH s přesností ±5% RH) a úrovně osvětlení (rozsah 0 až 1000 Lux) monitoruje MultiSensor také intenzitu UV záření. Aby toho nebylo málo, obsahuje MultiSensor také detektor pohybu PIR a senzor vibrací pro případ, že by se někdo pokoušel toto zařízení odnést. [40]

Obrázek 11.1: Zařízení MultiSensor 6 od společnosti Aeon Labs monitoruje pohyb, vibrace, teplotu, vlhkost, úroveň osvětlení a UV záření. [39]

Zařízení MultiSensor 6 může být napájeno externě 5 V DC přes micro USB konektor, či z baterií (2 × CR123A baterie, 3V). Výrobce udává výdrž 24 měsíců při použití baterií o kapacitě 1500 mAh. Při napájení z baterií může zařízení MultiSensor 6 pravidelně hlásit stav baterií řídicí jednotce. V případě potřeby tak bude uživatel včas upozorněn na nutnost vyměnit baterie. Aby bylo dosaženo nízké spotřeby energie, nedochází k měření a odesílání dat kontinuálně, ale v předem nastavených pravidelných intervalech. [40]

[1] Dostupné z https://smarterhome.sk/cs/bezdratove-senzory-a-detektory/aeonlabs-multisenzor-6-171.html za cenu 1608 Kč.

Jedná se o ideální zařízení pro testovací domácnost, neboť v jednom těle skrývá 6 různých senzorů. Výhodou je také skutečnost, že zatímco některé senzory měří více úrovní (teplota, vlhkost, úroveň osvětlení a UV záření), jiné poskytují binární výstup (pohyb, vibrace). Rozmanitost zaručuje také charakter měřených dat, kde teplota, vlhkost, úroveň osvětlení a UV záření je informativního charakteru na rozdíl od detekce pohybu a vibrací, což má bezpečnostní charakter.

11.3 Popp CO detektor

Další zasoupení třídy senzorů představuje detektor CO (oxid uhelnatý) od společnosti Popp[2]. Tento senzor s podporou technologie Z-Wave monitoruje úroveň oxidu uhelnatého v ovzduší a v případě, že překročí definovanou hranici, zajistí vhodné varování. Pro tento účel je Popp CO detektor vybaven sirénou (85 dB/3 m). Informace o překročení definované hranice je také v binární podobě předána příslušné Z-Wave řídicí jednotce. Jako krizovou situaci vyhodnotí detektor stav, kdy

1. úroveň CO je nad hodnotou 43 ppm po dobu 60 minut
2. úroveň CO je nad hodnotou 80 ppm po dobu 10 minut
3. úroveň CO je nad hodnotou 150 ppm po dobu 2 minut

Popp CO detektor je napájen z integrované baterie, u které výrobce udává výdrž až 10 let. Tato baterie není vyměnitelná, neboť s koncem životnosti baterie přichází i konec životnosti senzoru oxidu uhelnatého. Integrovaná baterie napájí samotný senzor a v případě potřeby i sirénu. Pro napájení Z-Wave modulu slouží 2xAAA baterie. Informace o stavu měnitelných AAA baterií se opět posílá do řídicí jednotky. [41]

Obrázek 11.2: Zařízení CO detektor od společnosti Popp monitoruje úroveň oxidu uhelnatého a v případě nebezpečí varuje uživatele vestavěnou sirénou, či zašle upozornění příslušné Z-Wave řídicí jednotce. [41]

Popp CO detektor je zařazen mezi zařízení testovací inteligentní domácnosti, neboť nabízí zcela odlišný přístup oproti zařízení MultiSensor 6 z hlediska důležitosti naměřených dat (větší riziko pro uživatele).

[2]Dostupný z https://smarterhome.sk/cs/bezdratove-senzory-a-detektory/popp-co-detektor-294.html za cenu 2380 Kč.

11.4 Fibaro zásuvka

Třídu senzorů máme díky zařízení MultiSensor 6 a Popp CO detektor zastoupenou dostatečně. Dále je potřeba vybavit inteligentní domácnost aktuátory. Typickým příkladem aktuátoru s binárním vstupem je inteligentní zásuvka od firmy Fibaro[3]. Pomocí tohoto adaptéru s podporou Z-Wave je možné jakoukoli zásuvku v domácnosti proměnit na inteligentní bez potřeby úprav a složité instalace. Adaptér stačí pouze vložit do zásuvky a následně připojit elektrický přístroj.

Hlavní funkcí inteligentní zásuvky je zapínání/vypínání napájení připojeného spotřebiče. Výrobcem uváděná maximální zátěž je omezena na 2500 W. Na vyšší zátěž, či překročení maximální teploty (55°C) upozorní zásuvka blikáním. Velikost zátěže je také průběžně indikována různou barvou vestavěných LED. Kromě toho umožňuje tato inteligentní zásuvka také monitorovat aktuální příkon připojeného spotřebiče i celkovou spotřebu elektrické energie. [42]

Obrázek 11.3: Inteligentní zásuvka od společnosti Fibaro umožňuje spínat zátěž, monitorovat aktuální příkon i celkovou spotřebu elektrické energie. [42]

Vzhledem k tomu, že pro svou funkci vyžaduje Fibaro zásuvka zapojení do elektrické sítě, je přes ní i napájena. Adaptér Fibaro tedy nevyžaduje dodatečné napájení z baterií. Na rozdíl od výše uvedených zařízení, která jsou napájena z baterií, zastává Fibaro zásuvka v Z-Wave síti i funkci opakovače. [42]

Fibaro zásuvku jsme vybrali jako součást naší inteligentní testovací domácnosti, neboť se jedná o aktuátor s binárním vstupem a širokým spektrem využití. Do zásuvky můžeme připojit jakýkoli elektrický spotřebič a následně řídit jeho napájení. Nabízí se ovládání lampičky, některého z kuchyňských spotřebičů, či zapínání elektrického topení.

11.5 Eurotronic Comet termostatická hlavice

Dalším zařízením, která reprezentuje třídu aktuátorů, je termostatická hlavice Eurotronic Comet[4]. Jedná se o inteligentní hlavici na topení, která otevírá a zavírá ventil radiátoru pomocí elektromotoru. Komunikaci s řídicí jednotkou zajišťuje technologie Z-Wave. Požadovanou teplotu je možné nastavit v rozsahu 6 až 28°C také přes ovládací

[3]Dostupné z https://smarterhome.sk/cs/zwave-bezdratove-zasuvky/fibaro-zasuvka-typ-f-fgwpf-102-zw5-312.html za cenu 1742 Kč.

[4]Dostupné z https://smarterhome.sk/cs/bezdratove-termostaty/eurotronic-comet-termostaticka-hlavice-188.html za cenu 1319 Kč.

kolečko. Informace o nastavení termostatické hlavice se zobrazuje na vestavěném displeji. Kromě ovládání teploty nabízí hlavice také několik automatických funkcí (detekce otevřeného okna, či ochrana proti zamrznutí). [43]

Pro instalaci nejsou potřebné žádné specializované nástroje. Stačí odšroubovat stávající klasickou hlavici a nahradit ji touto inteligentní termostatickou hlavicí. Po instalaci se provede automatické rozpoznání rozsahu radiátorového ventilu. Hlavice je tak kompatibilní s ventily většiny klasických výrobců (Heimeier, Danfoss, Honeywell,...). Napájení termostatické hlavice Comet zajišťují dvě baterie typu AA. Stejně jako u výše uvedených zařízení, tak i v případě hlavice Comet je stav baterií odesílán přes Z-Wave do řídicí jednotky. [43]

Obrázek 11.4: Termostatická hlavice Comet od společnosti Eurotronic umožňuje ovládat teplotu topení přes Z-Wave. [43]

Termostatická hlavice Comet vnáší do testovací inteligentní domácnosti další rozmanitost, neboť se jedná o aktuátor s víceúrovňovým vstupem. Vytápění navíc představuje další oblast, která nachází uplatnění v inteligentních domácnostech. Jistě bychom tedy v rámci navrhovaného hlasového ovládání měli uvažovat i řízení teploty v domácnosti.

11.6 Aeon Labs LED žárovka

Poslední na seznamu zařízení, které tvoří testovací inteligentní domácnost, je LED žárovka od společnosti Aeon Labs[5]. Tato žárovka umožňuje kromě zapínání/vypínání přes Z-Wave také plynulou regulaci jasu (až do maximálního světelného toku 850 lm). Přitom je možné regulovat teplotu vyzařovaného světla v rozmezí 2580 až 7050 K. Aby toho nebylo málo, jedná se o vícebarevnou žárovku s možností výběru až z 16 milionů barev. Při příkonu 9 W se jedná o ekvivalent klasické 70 wattové žárovky. Žárovka od společnosti Aeon Labs je kompatibilní s objímkami E26 a E27 a může tedy nahradit většinu standardních žárovek v domácnosti bez nutnosti úprav. Výrobce udává životnost žárovky 50 tisíc hodin. [44]

[5]Dostupné z https://smarterhome.sk/cs/zwave-bezdratove-stmivace/aeon-labs-led-ziarovka-154.html za cenu 1607 Kč.

Inteligentní žárovku od společnosti Aeon Labs jsme vybrali jako součást testovací inteligentní domácnosti, neboť se jedná o aktuátor s více ovládanými funkcemi (jas, barva). Její použití nevyžaduje žádnou speciální přípravu, tak jako stmívače instalované do zdi. Pro použití inteligentní žárovky mluví také skutečnost, že osvětlení je zřejmě jedna z nejběžnější oblastí ovládaných v inteligentní domácnosti.

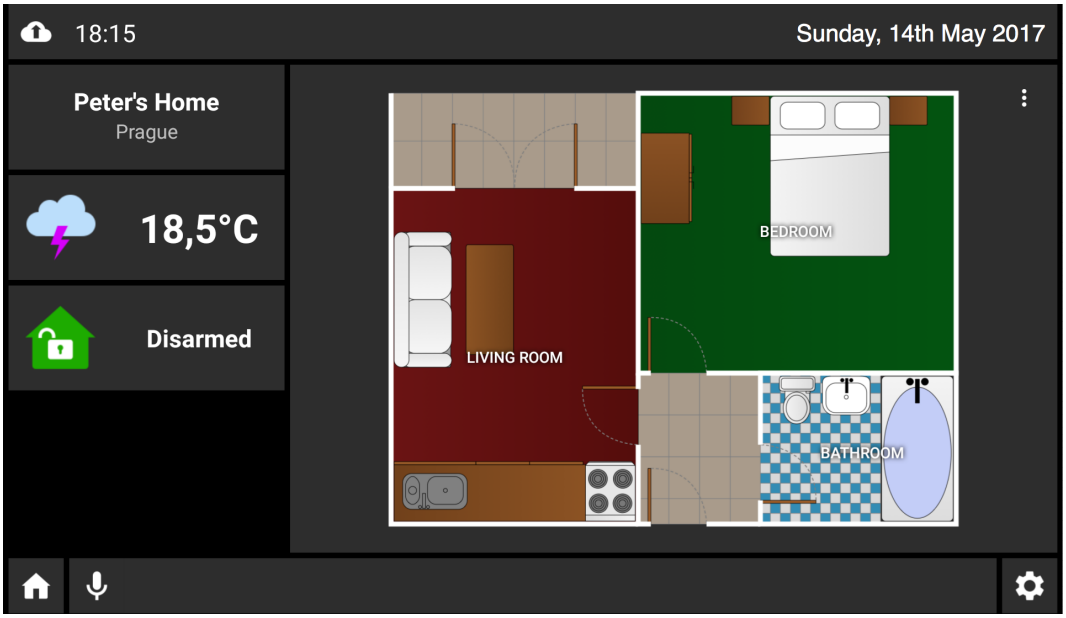
Obrázek 11.5: Vícebarevná LED žárovka od společnosti Aeon Labs představuje umožňuje plynulou regulaci jasu a výběr až z 16 milionů barev. [44]

12. Testování navrženého řešení

12.1 GUI aplikace HomeVoice

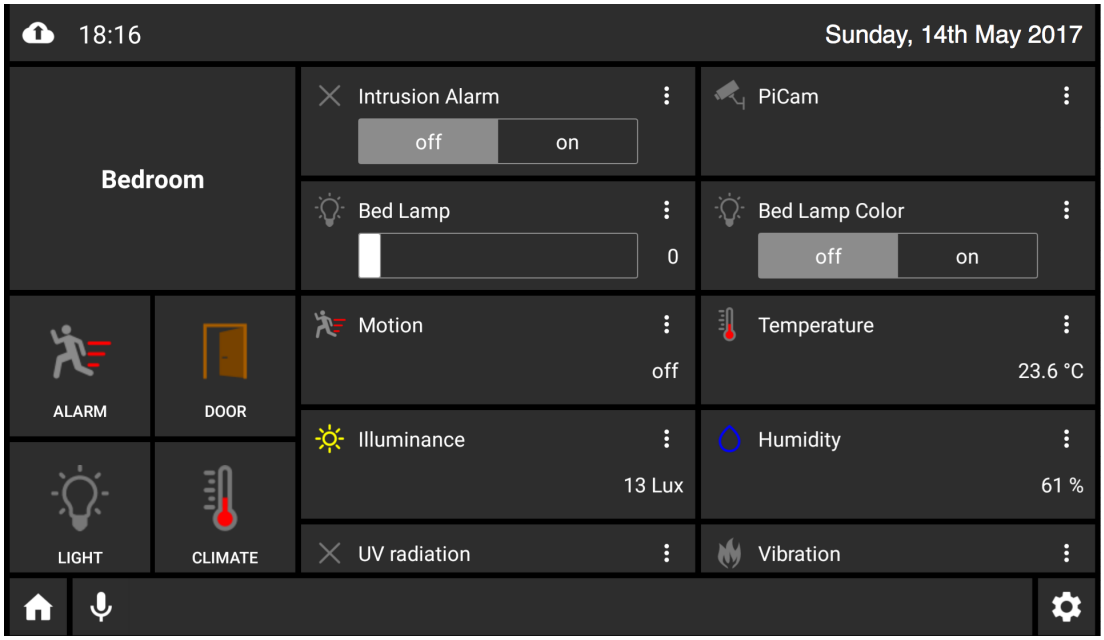
V předchozích kapitolách jsme podrobně rozebrali implementaci hlasového ovládání vybrané inteligentní domácnosti. Nyní nadešel čas námi navržené řešení otestovat. Abychom si zjednodušili vývoj a následně i testování, navrhli jsme aplikaci HomeVoice včetně plnohodnotného grafického uživatelského rozhraní. Mnoho prvků v rozhraní má charakter indikátoru, který nám umožní sledovat stav připojení k řídicí jednotce, probíhající detekci klíčového slova, či například přepis řeči na text. Prostřednictvím grafického uživatelského rozhraní můžeme také ovládat a monitorovat zařízení v inteligentní domácnosti.

Z hlediska grafiky se jedná kompletně o naši práci, přičemž jsme kladli důraz jak na design, tak na použitelnost aplikace HomeVoice. Jednotlivé grafické prvky (převážně ikony) jsme si buď připravili sami v programu Graphic[1], nebo je převzali z volně dostupných databází[2]. Zatímco kód (XML) realizující grafiku je čistě naší prací, logiku na pozadí (např. naplnění seznamu zařízení) jsme převzali z ukázkové aplikace Z-Way-Android.



Obrázek 12.1: Domovská obrazovka aplikace HomeVoice zobrazuje interaktivní plán domácnosti. Po kliknutí na vybranou místnost se zobrazí seznam zařízení, která se v této místnosti nachází (viz obr. níže). V dolní části obrazovky se nachází pole pro hlasové ovládání, kde ikona mikrofonu spouští rozpoznání řeči a v poli vedle se následně zobrazuje přepis zachycené řeči. Za povšimnutí stojí také ikona v levém horním rohu, která zobrazuje stav přípojení k řídicí jednotce (rozlišuje stavy: nepřipojeno, připojeno lokálně, připojeno přes internet)

[1] Graphic (dříve označovaný jako iDraw) je editor vektorové grafiky pro Mac OS X.
[2] Pro open-source projekty například zde: https://icons8.com



Obrázek 12.2: Po kliknutí na požadovanou místnost se objeví seznamem všech zařízení v této místnosti. Zde se dozvíme aktuální hodnotu/stav jednotlivých zařízení a zařízení typu aktuátor můžeme ovládat bez použití hlasu. Součástí této obrazovky je také sada filtrů (ikony vlevo), které po kliknutí zobrazí pouze zařízení dané kategorie.

12.2 Správa chyb pomocí Firebase

Hlavně v počátečních fázích vývoje přijde vhod zaznamenávání chyb a pádů navrhované mobilní aplikace. Díky tomu můžeme aplikaci testovat i bez připojení k počítači, aniž bychom se museli bát, že v případě pádu aplikace přijdeme o podrobné informace o chybě. V případě komerčního nasazení aplikace bychom tímto způsobem mohli získávat informace o pádech aplikace používané koncovými uživateli. Nejedná se o jednoduchou úlohu, neboť záznam chyb musí spolehlivě pracovat i tehdy, pokud aplikace není funkční. Naštěstí existují hotová řešení, která plní přesně tento úkol.

Z existujících řešení použijeme **Firebase Crash Reporting**, neboť služba Firebase je již součástí našeho projektu a zajišťuje zasílání dat prostřednictvím PUSH notifikací. Nemusíme tedy znovu doplňovat konfigurační soubor **google-services.json**, ale postačí přidat knihovnu `com.google.firebase:firebase-crash:9.4.0` pomocí nástroje Gradle. Tím je tato služby připravená a při dalším pádu aplikace je informace o pádu zaslána na server služby Firebase.

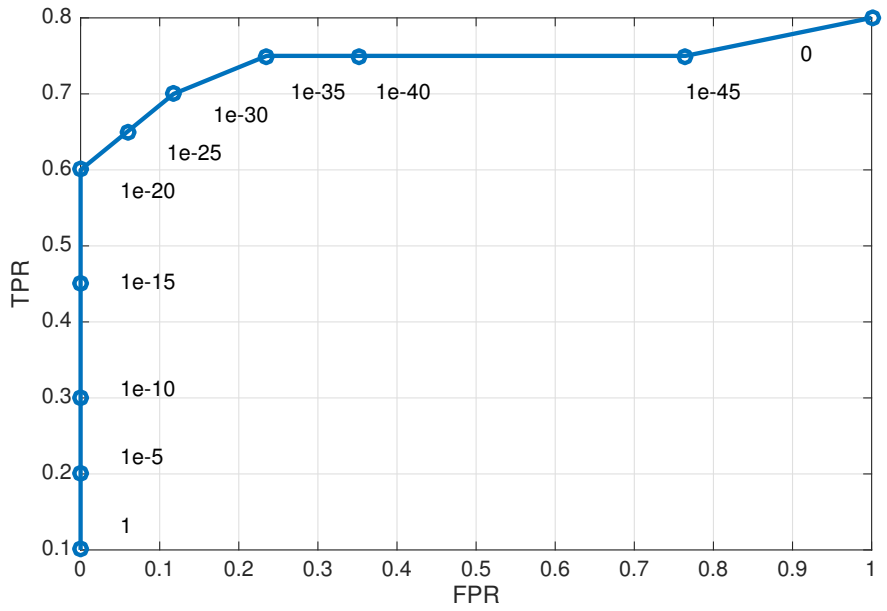
Zachycené pády aplikace HomeVoice jsou na serveru služby Firebase automaticky tříděny do souvisejících tříd, kde všechny případy v dané třídě mají stejnou příčinu. Každý den je navíc informace o chybách zaslána na e-mail uvedený v uživatelském profilu ve službě Firebase. Díky tomu je aplikace v podstatě neustále testována a v případě jakéhokoli problému jsme ihned na tento problém upozorněni včetně všech potřebných informací.

12.3 Nastavení detekce klíčového slova

Chování knihovny **PocketSphinx** při detekci klíčového slova je možné ovlivnit parametrem `KeywordThreshold`. Čím vyšší je hodnota tohoto parametru, tím méně případů falešné detekce klíčového slova zaznamenáme. Zároveň se ale zvýší pravděpodobnost nedetekování klíčového slova. Našim úkolem je tedy nalézt optimální hodnotu parametru `KeywordThreshold`. Pro řešení tohoto úkolu použijeme tzv. ROC analýzu.

ROC (z anglického výrazu Receiver Operating Characteristics) označuje analytickou metodu, která se používá pro ohodnocení výsledků binárního klasifikátoru (pozitivní, negativní) v závislosti na hodnotách vybraného parametru. Výstupem této metody je graf závislosti TPR[3] na FPR[4] pro různé varianty vybraného parametru. V našem případě budeme vynášet míru správně detekovaného klíčového slova v závislosti na míře falešné detekce klíčového slova pro různé hodnoty parametru `KeywordThreshold`.

Abychom mohli ROC křivku sestavit, potřebujeme nejprve příslušná data. Pro několik hodnot parametru `KeywordThreshold` otestujeme detekci klíčového slova na reálném zařízení NVIDIA SHIELD K1. Připravíme si dvě nahrávky, kde první obsahuje celkem 20 variant vyslovení klíčového slova (Ok Smart Home) bez rušivých zvuků na pozadí. Druhá nahrávka naopak obsahuje okolo 30 minut rozhovorů v angličtině na téma Smart Home. Pro vybrané hodnoty parametru `KeywordThreshold` jsme postupně přehráli obě připravené nahrávky a zaznamenali počet správně a chybně detekovaných klíčových slov. Z těchto dat jsme následně vypočítali TPR a FPR. Závislost TPR na FPR je zobrazena jako ROC křivka v grafu níže.



Obrázek 12.3: ROC analýza detekce klíčového slova pro parametr `KeywordThreshold`.

[3]TPR (True Positive Rate) označuje míru správně klasifikovaných pozitivních případů.
[4]FPR (False Positive Rate) představuje míru chybně klasifikovaných pozitivních případů.

Na základě ROC analýzy jsme zvolili následující hodnotu parametru `KeywordThreshold`:

```
KeywordThreshold = 1e-30
```

Pro toto nastavení by mělo být klíčové slovo detekováno správně v 70% případů. Míra falešné detekce klíčového slova přitom představuje pouze 11% v porovnání s maximální dosaženou mírou falešné detekce.

12.4 Testování služby Wit.ai

Nemá příliš velký smysl testovat knihovnu PocketSphinx, či službu pro rozpoznání řeči od Googlu, neboť tyto nástroje již byly podrobně otestovány (například v [50]). Co však nebylo a ani nemohlo být otestováno je extrahování záměru a entit pomocí služby Wit.ai. V tomto případě jsou totiž dosažené výsledky silně závislé na konkrétním natrénování této služby. Rozhodli jsme se tedy provést vlastní testování služby Wit.ai.

Pro testování jsme si připravili sadu testovacích vět, které pokrývají všechny natrénované oblasti, tj. ovládání světel (`SET_LIGHT`), monitorování světel (`GET_LIGHT`), monitorování teploty (`GET_TEMP`), monitorování vlhkosti (`GET_HUM`), ukončení dialogu (`CANCEL`), ukončení aplikace (`EXIT_APP`) a získání informací o aplikaci (`ABOUT`). Celkem jsme připravili 46 různých vět (viz příloha E), které se více, či méně liší od trénovacích vět. Následně jsme tyto věty zaslali do služby Wit.ai a zkoumali navrácené hodnoty záměru a entit. Dosažené výsledky jsou zobrazeny v tabulce níže.

Kategorie	Funkční	Chybějící záměr	Chybný záměr	Chybějící entita	Chybná entita	Přebývající entita
SET_LIGHT	3/7	1/7	0/7	1/7	3/7	2/7
GET_LIGHT	1/7	3/7	2/7	4/7	3/7	4/7
GET_TEMP	6/8	1/8	0/8	2/8	0/8	8/8
GET_HUM	5/6	0/6	0/6	1/6	0/6	6/6
CANCEL	1/5	3/5	1/5	0/5	0/5	4/5
EXIT_APP	5/7	2/7	0/7	0/7	0/7	7/7
ABOUT	6/6	0/6	0/6	0/6	0/6	6/6

Tabulka 12.1: Výsledky testování služby Wit.ai.

V této tabulce jsme uváděli počet chybně rozpoznaných záměrů, chybějících záměrů (navrácená struktura neobsahuje záměr), chybějících entit, chybně určených hodnot entit a míru entit, které byly rozpoznány nad rámec očekávání. Dále jsme uvedli míru extrahovaných záměru a entit, které sice nemuseli být zcela správně určené (např. přebývá některá z entit), ale v rámci aplikace HomeVoice dojde k jejich správnému zpracování. Extrakce dopadla dle očekávání (byla funkční) v 58% případů, což je dle nás velmi dobrý výsledek. Testovací věty můžeme navíc použít pro další trénování služby Wit.ai a tím dále zlepšit rozpoznání záměru a entit.

Jak je patrné z uvedené tabulky, prakticky ve všech případech přebývala některá entita (nejčastěji `on_off`). Dle našeho pozorování k tomuto docházelo s přibývajícím počtem trénovacích vět. Soudíme tedy, že se jedná o důsledek přeučení systému. Naštěstí se nejedná o závažný problém, neboť s přebývajícími entitami si poradí námi navržený mechanismus v rámci zpracování záměru a entit v aplikaci HomeVoice.

13. Závěr

Cílem této diplomové práce bylo navrhnout a implementovat hlasové ovládání vybrané inteligentní domácnosti prostřednictvím mobilní aplikace (pro OS Android). Tato úloha v sobě zahrnovala volbu bezdrátové komunikační technologie, přípravu inteligentní domácnosti, propojení mobilního zařízení s inteligetní domácností a realizaci hlasového ovládání. Při vývoji mobilní aplikace HomeVoice jsme se inspirovali ukázkovou aplikací Z-Way-Android. Z této aplikace jsme převzali rozhraní pro komunikaci s řídicí jednotkou a základ pro grafické uživatelské rozhraní.

První část práce obsahuje teoretický rozbor problematiky inteligentních domácností a stručné dělení inteligentních zařízení. Zároveň se zde nachází popis několika existujících technologií (od protokolů po kompletní ekosystémy), na kterých je dnes možné vystavět intelignentní domácnost. Z uvedených technologií jsme vybrali ekosystém Z-Wave, který nás přesvědčil svým zpracováním (pokrývá vše od fyzické až po aplikační vrstvu) a množstvím vzájemně kompatibilních zařízení (1700 zařízení od více než 450 výrobců).

Dále jsme se zaměřili na volbu řídicí jednotky, která představuje nezbytnou součást inteligentní domácnosti založené na technologii Z-Wave. Při výběru řídicí jednotky hrál důležitou roli použitý řídicí software. Na základě průzkumu jsme zvolili software Z-Way. Jedná se sice o komerční řešení, ale pro majitele kompatibilního hardwaru je k dispozici zdarma a nabízí široké možnosti programování. Z hlediska hardwaru jsme pro stavbu řídicí jednotky použili modul RaZberry (podporován softwarem Z-Way) a mikropočítač Raspberry Pi 3 s operačním systémem Raspbian. Z výše zmíněných komponent jsme připravili řídicí jednotku. Nevynechali jsme podrobný popis instalace OS Raspbian a řídicího softwaru Z-Way. Pro úplnost jsme uvedli také možnosti programování vybrané řídicí jednotky, resp. softwaru.

Následně jsme se podívali na možnosti komunikace mezi mobilním zařízením (aplikací HomeVoice) a řídicí jednotkou s REST API. Ačkoli implementace připojení k řídicí jednotce z lokální sítě i z internetu byla součástí ukázkové aplikace Z-Way-Android, nepodařilo se nám ji uvést do funkčního stavu. Pro použití v aplikaci HomeVoice jsme museli přepsat zastaralé části a opravit proces autorizace. Z ukázkové aplikace jsme převzali také implementaci skenování zařízení na lokální síti, která je součástí námi představeného mechanismu pro automatické připojení k řídicí jednotce. Díky tomu jsme z pohledu uživatele výrazně zjednodušili proces připojování k řídicí jednotce (bez nutnosti složité konfigurace aplikace HomeVoice).

Doposud byla veškerá komunikace iniciována mobilní aplikací HomeVoice, což je sice vhodné pro ovládání inteligentní domácnosti, ale neumožňuje to rychlý a efektivní přenos informací o událostech v inteligentní domácnosti v době jejich vzniku. Hlavní přínos této práce vidíme v návrhu komunikace ve směru z řídicí jednotky do mobilního zařízení. Doposud k tomuto účelu sloužily SMS, či e-maily. Námi navržené řešení využívá pro přenos dat PUSH notifikací vázaných na konkrétní aplikaci v OS Android. Nabízí se tak automatické zpracování těchto dat v rámci příslušné aplikace (v našem případě HomeVoice) a okamžité upozornění uživatele například na probíhající požár.

Protože oficiální možnost ovládání Z-Wave inteligentní domácnosti prostřednictvím existujícího hlasového rozhraní v OS Android neexistuje, navrhli jsme vlastní řešení pokrývající vše od detekce klíčového slova (náhrada za Ok Google) až po syntézu řeči. Zde vidíme další přínos této diplomové práce, neboť jsme představili kompletní architekturu pro hlasové ovládání inteligentních domácností z OS Android. S použitím existujících knihoven a služeb se nám podařilo realizovat hlasové ovládání, které rozumí přirozenému jazyku a dokáže vést s uživatelem krátké dialogy podobně jako aplikace Google Now. Podstatnou část při realizaci hlasového ovládání představuje natrénování služby Wit.ai, která zajišťuje porozumnění přirozenému jazyku (extrahování záměru a entit). Za tímto účelem jsme vytvořili sadu trénovacích vět, záměrů a entit. Tyto věty jsme následně použili pro natrénování služby Wit.ai. Dále jsme představili mechanismus pro zpracování záměru a entit, který v kontextu konkrétní inteligentní domácnosti dokáže vést s uživatelem dialog a zajistit vykonání požadované akce.

V další části diplomové práce jsme se zaměřili na testování navrženého řešení. Abychom mohli testovat aplikaci HomeVoice na reálném hardwaru, vybrali jsme několik Z-Wave zařízení, které budou tvořit testovací intelignentní domácnost. Vzhledem k rozsahu této diplomové práce nebylo časově reálné provádět systematické testování dílčích částí aplikace HomeVoice (unit testy) ani integrační testy. Většina chyb tak byla zachycena při průběžném praktickém testování aplikace. Pro tento účel jsme se rozhodli použít službu Firebase Crash Reporting, která automaticky zachytává pády aplikace HomeVoice a odesílá informace o nich na web.

Co se týká testování hlasového ovládání z pohledu uživatele, většina použitých knihoven a služeb již byla důkladně otestována a jejich výkony jsou dobře známy. Za otestování ovšem stojí detekce klíčového slova, neboť nám to pomůže nalézt vhodnou konfiguraci knihovny PocketSphinx pro zvolenou aktivační frázi. Protože detekce klíčového slova představuje obecně binární klasifikátor, rozhodli jsme se použít ROC analýzu. Připravili jsme tedy vhodné audio nahrávky, zaznamenali počet správně a chybně detekovaných klíčových slov pro různé hodnoty vybraného parametru a následně sestavili ROC křivku. Díky tomu se nám podařilo nastavit detekci klíčového slova tak, že správně rozpozná aktivační frázi v 70% případů.

Další část hlasového ovládání, kterou je třeba otestovat, představuje služba Wit.ai pro extrahování záměru a entit. Její chování je totiž silně závislé na natrénování této služby. Abychom tak mohli učinit, připravili jsme si sadu testovacích vět a očekávaných záměrů a entit. Tyto věty jsme následně předkládaly službe Wit.ai a zaznamenávali si odchylky od očekávání. Na základě tohoto zkoumání jsme zjistili, že si služba Wit.ai poradí s téměř 60% vět, které dříve neznala, tj. nebyly použity pro natrénování služby.

Závěrem je tedy možno říci, že se nám podařilo navrhnout dostatečně robustní a univerzální řešení, které umožňuje ovládat Z-Wave inteligentní domácnost z mobilního zařízení prostřednictvím hlasu. Jedná se o volně dostupnou alternativu k inteligentním asistentům Amazon Echo a Google Now, která navíc uživateli poskytne grafické uživatelské rozhraní na displeji mobilního zařízení. Na rozdíl od uvedených asistentů uživateli postačí standardní telefon s OS Android. Od ovládání inteligentní domácnosti ho pak dělí už jenom instalace aplikaci HomeVoice a vyslovení fráze "Ok Smart Home".

Zdroje

[1] *Here's how the Internet of Things will explode by 2020*. Business Insider [on-line], 08/2016. Dostupné z http://www.businessinsider.com/iot-ecosystem-internet-of-things-forecasts-and-business-opportunities-2016-2 [citováno 14.05.2017]

[2] Schiefer, Michael. *Smart Home Definition and Security Threats*. Ninth International Conference on IT Security Incident Management & IT Forensics, Magdeburg, 2015. ISBN 978-1-4799-9902-6. Dostupné z http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7195812&isnumber=7195793 [citováno 27.12.2016]

[3] Harper, Richard. *Inside the Smart Home*. Springer-Verlag London Limited, 2003. ISBN 1-85233-688-9.

[4] *Definition of smart home in english* . Oxford Dictionaries [on-line]. Dostupné z https://en.oxforddictionaries.com/definition/smart_home [citováno 27.12.2016]

[5] *What is a "Smart Home"?*. SmartHomeUSA [on-line]. Dostupné z http://www.smarthomeusa.com/smarthome/ [citováno 27.12.2016]

[6] *Playbulb Smart Bulb*. Playbulb [on-line]. Dostupné z http://www.playbulb.com/en/playbulb-smart.html [citováno 28.12.2016]

[7] *Smart Device*. Techopedia [on-line]. Dostupné z https://www.techopedia.com/definition/31463/smart-device [citováno 28.12.2016]

[8] Kyas, Othmar. *How To Smart Home*. Key Concept Press, 2013. ISBN 978-3-944980-00-3. [citováno 29.12.2016]

[9] Ripka, Pavel, Ďaďo, Stanislav, Kreidl, Marcel, Novák, Jiří. *Senzory a převodníky*. Vydavatelství ČVUT, 2005. ISBN 80-01-03123-3. [citováno 29.12.2016]

[10] Holub, Jan, Novák, Jiří. *Přednášky předmětu A3B38DSY*. Stránky předmětu A3B38DSY, 2014. [citováno 31.12.2016]

[11] Kuzlt, M., Pipattanasomporr, M., Rahman, S.. *Review of communication technologies for smart homes/building applications*. 2015 IEEE Innovative Smart Grid Technologies - Asia (ISGT ASIA), 2015. Dostupné z http://80.ieeexplore.ieee.org.dialog.cvut.cz/xpls/icp.jsp?arnumber=7437036 [citováno 31.12.2016]

[12] *Recommendation ITU-T G.9960*. Telecommunication Standardization Sector of ITU, 07/2015. Dostupné z https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-G.9960-201507-I!!PDF-E&type=items [citováno 7.1.2017]

[13] Rahman, M., Hong, Ch. S., Lee, S.. *Medium access control for power line communications: an overview of the IEEE 1901 and ITU-T G.hn standards*. IEEE Communications Magazine (vol. 49), 06/2011. Dostupné z http://80.ieeexplore.ieee.org.dialog.cvut.cz/document/5784004/ [citováno 7.1.2017]

[14] Loewy, Chen. *Make your smart home better connected with a Wi-Fi mesh network*. Texas Instruments E2E Community, 08/2016. https://e2e.ti.com/blogs_/b/connecting_wirelessly/archive/2016/08/03/make-your-smart-home-better-connected-with-a-wi-fi-mesh-network [citováno 7.1.2017]

[15] Jorád Córdova, C.E.P., Asare-Bediako, B, Vanalme, G.M.A., Kling, W.L.. *Overview and Comparison of Leading Communication Standard Technologies for Smart Home Area Networks Enabling Energy Management Systems*. 46th International Universities' Power Engineering Conference, 09/2011. http://80.ieeexplore.ieee.org.dialog.cvut.cz/stamp/stamp.jsp?tp=&arnumber=6125611 [citováno 7.1.2017]

[16] Gislason, Drew. *ZigBee Wireless Networking*. Newnes, 08/2008. ISBN 978-0750685979.

[17] *EnOcean Technology*. EnOcean [on-line]. Dostupné z https://www.enocean.com/en/technology/ [citováno 20.2.2017]

[18] *Understanding Z-Wave Networks, Nodes & Devices*. Vesternet [on-line]. Dostupné z http://www.vesternet.com/resources/technology-indepth/understanding-z-wave-networks [citováno 26.3.2017]

[19] *IoT battle of the titans: Bluetooth 5 vs. Wi-Fi HaLow*. IoT Agenda [on-line]. Dostupné z http://www.vesternet.com/resources/technology-indepth/understanding-z-wave-networks [citováno 1.4.2017]

[20] *What is ZigBee?*. ZigBee [on-line]. Dostupné z http://www.zigbee.org/what-is-zigbee/ [citováno 1.4.2017]

[21] *About Z-Wave*. Z-Wave [on-line]. Dostupné z http://www.z-wave.com/about [citováno 1.4.2017]

[22] *VeraEdge*. Vera [on-line]. Dostupné z http://getvera.com/controllers/veraedge/ [citováno 1.4.2017]

[23] *Luup Intro*. micasaverde.com [on-line]. Dostupné z http://wiki.micasaverde.com/index.php/Luup_Intro [citováno 1.4.2017]

[24] *Javascript API for UI7*. micasaverde.com [on-line]. Dostupné z http://wiki.micasaverde.com/index.php/JavaScript_API [citováno 1.4.2017]

[25] *UPnP Technical basics: UPnP Device Architecture (UDA)*. UPnP [on-line]. Dostupné z http://upnp.org/resources/documents/UPnP_UDA_tutorial_July2014.pdf [citováno 2.4.2017]

[26] *About openHAB*. openHAB [on-line]. Dostupné z http://docs.openhab.org/introduction.html [citováno 2.4.2017]

[27] *What is Freedomotic?*. freedomotic [on-line]. Dostupné z http://freedomotic-user-manual.readthedocs.io/en/latest/the-project/what-is-freedomotic.html [citováno 2.4.2017]

[28] *Users tutorial*. freedomotic [on-line]. Dostupné z http://www.freedomotic.com/content/users-tutorial [citováno 2.4.2017]

[29] *OpenRemote Documentation*. OpenRemote Github [on-line]. Dostupné z https://github.com/openremote/Documentation/wiki [citováno 2.4.2017]

[30] *Z-Way the Gold-Standard Z-Wave Controller*. Z-Wave>Me [on-line]. Dostupné z https://www.z-wave.me/index.php?id=22 [citováno 2.4.2017]

[31] *Z-Way Developers Documentation*. Z-Wave>Me [on-line]. Dostupné z https://razberry.z-wave.me/docs/zway.pdf [citováno 3.4.2017]

[32] *UZB - the world-smallest Z-Wave USB Stick*. Z-Wave>Me [on-line]. Dostupné z https://www.z-wave.me/index.php?id=28 [citováno 3.4.2017]

[33] *Home Automation with Raspberry Pi, Z-Wave devices and Domoticz*. IQ JAR [on-line]. Dostupné z http://iqjar.com/jar/home-automation-with-raspberry-pi-z-wave-devices-and-domoticz/ [citováno 3.4.2017]

[34] *GPIO*. Raspberry Pi [on-line]. Dostupné z https://www.raspberrypi.org/documentation/hardware/raspberrypi/gpio/README.md [citováno 3.4.2017]

[35] *Raspberry Pi 3 Model B*. Raspberry Pi [on-line]. Dostupné z https://www.raspberrypi.org/products/raspberry-pi-3-model-b/ [citováno 3.4.2017]

[36] *Welcome to Raspbian*. Raspbian [on-line]. Dostupné z https://www.raspbian.org [citováno 9.4.2017]

[37] *Formatting an SDXC card for use with NOOBS*. Raspberry Pi [on-line]. Dostupné z https://www.raspberrypi.org/documentation/installation/sdxc_formatting.md [citováno 9.4.2017]

[38] *The Secure Shell (SSH) Transport Layer Protocol*. IETF [on-line]. Dostupné z https://tools.ietf.org/html/rfc4253 [citováno 9.4.2017]

[39] *AEONLABS MULTISENZOR 6*. SmarterHOME [on-line]. https://smarterhome.sk/cs/bezdratove-senzory-a-detektory/aeonlabs-multisenzor-6-171.html [citováno 14.4.2017]

[40] *MultiSensor 6*. AEOTEC [on-line]. Dostupné z http://aeotec.com/z-wave-sensor [citováno 14.4.2017]

[41] *Popp CO detektor*. POPP [on-line]. Dostupné z http://www.popp.eu/wp-content/uploads/2016/08/Manual_CO-Detector_POPP_en.pdf [citováno 14.4.2017]

[42] *Wall Plug*. Fibaro [on-line]. Dostupné z https://www.fibaro.com/en/products/wall-plug/ [citováno 15.4.2017]

[43] *Radiator control with Comet Z-Wave*. Eurotronic [on-line]. Dostupné z https://www.eurotronic.org/en/products/comet-z-wave.html [citováno 15.4.2017]

[44] *Z-Wave LED Bulb*. AEOTEC [on-line]. Dostupné z http://aeotec.com/z-wave-led-lightbulb [citováno 15.4.2017]

[45] *The RaZberry Solution*. RAZBERRY [on-line]. Dostupné z https://razberry.z-wave.me/index.php?id=2 [citováno 15.4.2017]

[46] *Z-Way-Android*. GitHub [on-line]. Dostupné z https://github.com/Z-Wave-Me/Z-Way-Android [citováno 24.4.2017]

[47] *Firebase Cloud Messaging*. Firebase [on-line]. Dostupné z https://firebase.google.com/docs/cloud-messaging/ [citováno 1.5.2017]

[48] Allen, Grant. *Android 4 Průvodce programováním mobilních aplikací*. Computer Press Brno, 2013. ISBN 978-80-251-3782-6 [citováno 5.5.2017]

[49] *PocketSphinx 5prealpha*. GitHub [on-line]. Dostupné z https://github.com/cmusphinx/pocketsphinx [citováno 6.5.2017]

[50] Hjulström, Rickard. *Evaluation of a speech recognition system*. UMEÅ Universitet [on-line]. Dostupné z http://www.diva-portal.org/smash/get/diva2:852179/FULLTEXT01.pdf [citováno 7.5.2017]

[51] Koppen, Thijs. *Api.ai vs Wit.ai (or is it Google vs Facebook?)*. The Marketing Technologist [on-line]. Dostupné z https://www.themarketingtechnologist.co/api-ai-vs-wit-ai/ [citováno 8.5.2017]

Seznam symbolů a zkratek

symbol/zkratka	popis
IoT	Internet věcí
IP	Internet protokol
OS	Operační systém
SSH	Secure Shell
API	Application Programming Interface
HTTP	Hypertext Transfer Protocol
JSON	JavaScript Object Notation
SMS	Short Message Service
CO	Oxid uhelnatý
HVAC	Heating, ventilation and air conditioning
NAT	Datové úložiště na síti
ADC	Analogově digitální převodník
UV	Ultrafialové
IR	Infračervené
DVD	Digital Versatile Disc
HiFi	High fidelity
ITU	Mezinárodní telekomunikační unie
PLC	PowerLine Communication
WiFi	Wireless fidelity
ISM	Industrial, scientific and medical
MAC	Media Access Control
CSMA/CA	Carier Sense Multiple Access with Collision Avoidance
AP	Přístupový bod
P2P	Klient-klient
WPAN	Wireless Personal Area Network
USB	Universální sériová sběrnice
LE	Low Energy
FFD	Full Function Device
RFD	Reduced Function Device
USD	Americký dolar
RFID	Radio Frequency Identification
ID	Identifikátor
ROM	Paměť pouze pro čtení
LF	Nízká frekvence
HF	Vysoká frekvence
VHF	Velmi vysoká frekvence
MF	Mikrovlny
REST	Representational State Transfer

symbol/zkratka	popis
URI	Jednotný identifikátor zdroje
URL	Jednotná adresa zdroje
UPnP	Universal Plug and Play
SOAP	Simple Object Access Protocol
XML	Extensible Markup Language
GUI	Grafické uživatelské rozhraní
RAM	Paměť s náhodným přístupem
HDMI	High-Definition Multi-Media Interface
GPIO	General Purpose Input/Output
I2C	Inter-Integrated Circuit
SPI	Sériové periferní rozhraní
UART	Asynchronní sériové rozhraní
EEPROM	Elektronicky vymazatelná paměť pouze pro čtení
CSI	Camera Serial Interface
DSI	Display Serial Interface
SD	Storage Device
FAT	File Allocation Table
NOOBS	New Out of the Box Software
ZIP	Souborový formát
TCP	Transmission Control Protocol
UI	Uživatelské rozhraní
vDev	Virtual Device
zDev	Z-Wave Device
JS	JavaScript
DHCP	Dynamic Host Configuration Protocol
CIDR	Classless Inter-Domain Routing
IPv4	Internet protokol verze 4
IPv6	Internet protokol verze 6
NAT	Překlad síťových adres
SDK	Software Development Kit
FIFO	First in, first out
SSL	Secure Sockets Layers
TLS	Transport Layer Security
DynDNS	Dynamic Domain Name System
ISP	Poskytovatel internetu
FCM	Firebase Cloud Messaging
XMPP	Extensible Messaging and Presence Protocol
RH	Relativní vlhkost
PIR	Pasivní infračervený senzor
ppm	Parts per million
LED	Light-Emitting Diode
ROC	Receiver Operating Characteristic
TPR	True Positive Rate
FPR	False Positive Rate

Seznam obrázků

A. Parametry zařízení (vDev API)

Hlavním rysem rozhraní vDev API jsou tzv. Virtuální zařízení. V rámci řídicího softwaru Z-Way představuje Virutální zařízení objekt, který disponuje určitými vlastnostmi a nabízí funkce pro manipulaci s nimi. Níže uvedená tabulka obsahuje seznam všech dostupných vlastností (včetně datových typů), jejich popis a informaci, zda je možné jednotlivé vlastnosti měnit.

Parametr	Typ	Fixní	Popis
id	string	ano	jedinečný identifikátor
deviceType	string	ano	typ virtuálního zařízení (např. switchBinary, thermostat, sensorMultilevel)
updateTime	integer	ano	čas poslední aktualizace (počet sekund od roku 1970)
metrics	struktura	-	parametry zařízení
metrics.probeTitle	string	ne	název funkce zařízení (např. Teploměr)
metrics.scaleTitle	string	ne	jednotka (např. %, °C)
metrics.level	string integer	ne	stav/hodnota zařízení (např. 21.4, on/off, ALARM)
metrics.icon	string	ne	ikona zařízení (URI, typ)
metrics.title	string	ne	název fyzického zařízení (např. Teploměr v kuchyni)
creationTime	integer	ne	čas vytvoření zařízení
creatorId	integer	ne	identifikátor instance, která toto zařízení vytvořila (typicky ZWave)
hasHistory	boolean	ne	virtuální zařízení má historii
permanently_hidden	boolean	ne	zařízení je trvale skryté (neaktivní)
probeType	string	ano	podrobnější třída zařízení (např. temperature)
visibility	boolean	ne	zařízení je skryté v uživatelském rozhraní
tags	pole	ne	seznam TAGů přiřazených virtuálnímu zařízení

Tabulka A.1: Seznam parametrů virtuálních zařízení ve vDev API. Více informací je možné nalézt zde: http://docs.zwayhomeautomation.apiary.io/#reference/devices/virtual-device.

B. Příkazy zařízení (vDev API)

Typické použití Virtuálních zařízení představuje jejich namapování na funkce fyzických (skutečných) zařízení. V případě aktuátoru je potřeba tyto funkce ovládat. Ve vDev API, resp. jeho JSON verzi za tímto účelem existuje následující URL:

`/ZAutomation/api/v1/devices/<id>/command/<příkaz>`

Níže uvedená tabulka obsahuje seznam podporovaných příkazů (včetně použitých parametrů) a také popis funkce jednotlivých příkazů.

Příkaz	Parametry	Typ	Popis
on	-	-	zapnutí zařízení/funkce
off	-	-	vypnutí zařízení/funkce
increase	-	-	navýšení hodnoty o 10
decrease	-	-	snížení hodnoty o 10
open	-	-	odemknutí dveřního zámku, otevření náhledu IP kamery
close	-	-	zamknutí dveřního zámku, zavření náhledu IP kamery
setMode	mode	string	vybírá kofiguraci zařízení
exact	level	string	nastaví konkrétní hodnotu, např. teplota na termostatu
exact	red green blue	integer integer integer	nastaví barvu zařízení
zoomIn	-	-	přiblížení obrazu IP kamery
zoomOut	-	-	oddálení obrazu IP kamery
left	-	-	posun obrazu doleva
right	-	-	posun obrazu doprava
up	-	-	posun obrazu nahoru
down	-	-	posun obrazu dolů

Tabulka B.1: Seznam příkazů pro ovládání virtuálních zařízení. Více informací je možné nalézt zde: http://docs.zwayhomeautomation.apiary.io/#reference/devices/virtual-device.

C. Záměry a entity v HomeVoice

Základním prvkem námi použité metody pro porozumění přirozenému jazyku jsou tzv. entity. Jednotlivé entity jsou automaticky extrahovány z textu pomocí služby Wit.ai, kterou jsme dříve za tímto účelem natrénovali. Entita `intent` a entity začínající předponou `wit/` jsou vybrány z existujících entit ve službě Wit.ai, ostatní jsme vytvořili výhradně pro použití v aplikaci HomeVoice.

Entita	Metoda	Hodnoty	Popis
intent	trait	viz níže	záměr
wit/on_off	-	on, off	binární stav on/off
wit/number	-	six, zero, 1, 3.14,...	číslo
wit/location	-	Prague, bedroom,...	název lokace
yes_no	trait	yes, no	potvrzení/zamítnutí
query	trait	value, count, list	druh dotazu

Tabulka C.1: Seznam entit použitých ve službě Wit.ai.

Entita `intent` vyjadřuje úmysl uživatele. Zde je potřeba nadefinovat jednotlivé hodnoty podle konkrétní domény (ovládání vybraných funkcí inteligentní domácnosti). V našem případě jsme použili seznam záměrů uvedený níže.

Záměr	Popis
SET_LIGHT	ovládání binárních a víceúrovňových typů světel
GET_LIGHT	dotaz na světla (hodnota, počet, názvy)
GET_TEMP	dotaz na teplotní senzory
GET_HUM	dotaz na senzory vlhkosti
CANCEL	zrušení dialogu
EXIT_APP	ukončení aplikace
ABOUT	informace o aplikaci

Tabulka C.2: Seznam použitých záměrů.

D. Trénovací set pro Wit.ai

D.1 Ovládání světel

Věta	záměr	entita (hodnota)
Turn on the light in bedroom.	SET_LIGHT	on_off (on) location (bedroom)
Turn the light on in bedroom.	SET_LIGHT	on_off (on) location (bedroom)
Turn the bedroom light on.	SET_LIGHT	on_off (on) location (bedroom)
Turn on the lights everywhere.	SET_LIGHT	on_off (on) location (everywhere)
Dim the light in bedroom to 10%.	SET_LIGHT	number (10) location (bedroom)
Set the light level 50% in bedroom.	SET_LIGHT	number (50) location (bedroom)
Brighten the bedroom light to 80%.	SET_LIGHT	number (80) location (bedroom)

Tabulka D.1: Trénovací sada pro ovládání světel.

D.2 Monitorování světel

Věta	záměr	entita (hodnota)
Is there any light in bedroom?	GET_LIGHT	query (count) location (bedroom)
How many lights are in bedroom?	GET_LIGHT	query (count) location (bedroom)
Which lights are in bedroom?	GET_LIGHT	query (list) location (bedroom)
List of lights in bedroom?	GET_LIGHT	query (list) location (bedroom)
Name all lights in bedroom?	GET_LIGHT	query (list) location (bedroom)
Is the light turned on in bedroom?	GET_LIGHT	query (value) location (bedroom)
What is the bedroom light level?	GET_LIGHT	query (value) location (bedroom)

Tabulka D.2: Trénovací sada pro monitorování světel.

D.3 Monitorování teploty

Věta	záměr	entita (hodnota)
What is the temperature in bedroom?	GET_TEMP	location (bedroom)
Tell me the bedroom temperature.	GET_TEMP	location (bedroom)
Get the temperature in bedroom.	GET_TEMP	location (bedroom)
Temperature in bedroom?	GET_TEMP	location (bedroom)
How cold is it in bedroom?	GET_TEMP	location (bedroom)
How hot is it in bedroom?	GET_TEMP	location (bedroom)
Is it cold in bedroom?	GET_TEMP	location (bedroom)
Is it hot in bedroom?	GET_TEMP	location (bedroom)

Tabulka D.3: Trénovací sada pro monitorování teploty.

D.4 Monitorování vlhkosti

Věta	záměr	entita (hodnota)
What is the humidity in bedroom?	GET_HUM	location (bedroom)
Tell me the bedroom humidity.	GET_HUM	location (bedroom)
Get the humidity in bedroom.	GET_HUM	location (bedroom)
Humidity in bedroom?	GET_HUM	location (bedroom)
How wet is it in bedroom?	GET_HUM	location (bedroom)
How dry is it in bedroom?	GET_HUM	location (bedroom)

Tabulka D.4: Trénovací sada pro monitorování vlhkosti.

D.5 Ukončení dialogu

Věta	záměr	entita (hodnota)
Cancel!	CANCEL	-
Stop!	CANCEL	-
Be quiet!	CANCEL	-
Stop talking!	CANCEL	-
Nothing!	CANCEL	-

Tabulka D.5: Trénovací sada pro ukončení dialogu.

D.6 Ukončení aplikace HomeVoice

Věta	záměr	entita (hodnota)
Goodbye.	EXIT_APP	-
See you later.	EXIT_APP	-
Close everything.	EXIT_APP	-
Bye.	EXIT_APP	-
Have a nice time.	EXIT_APP	-
Close yourself.	EXIT_APP	-
Go away.	EXIT_APP	-

Tabulka D.6: Trénovací sada pro ukončení aplikace HomeVoice.

D.7 Informace o aplikaci HomeVoice

Věta	záměr	entita (hodnota)
What can you do?	ABOUT	-
How can you help me?	ABOUT	-
What are you good for?	ABOUT	-
Who are you?	ABOUT	-
Tell me something about you?	ABOUT	-
What can you do?	ABOUT	-

Tabulka D.7: Trénovací sada pro získání informací o aplikaci HomeVoice.

E. Testovací set pro Wit.ai

E.1 Ovládání světel

Věta	záměr	entita (hodnota)
Light off in kitchen.	SET_LIGHT	on_off (off) location (kitchen)
Could you turn the kitchen light on?	SET_LIGHT	on_off (on) location (kitchen)
Cut off the restroom light, please.	SET_LIGHT	on_off (off) location (restroom)
Switch the lights off everywhere.	SET_LIGHT	on_off (off) location (everywhere)
Turn the kitchen lights out.	SET_LIGHT	on_off (off) location (kitchen)
Set the bathroom light 20%.	SET_LIGHT	number (20) location (bathroom)
Dim the attic to 10%.	SET_LIGHT	number (80) location (bedroom)

Tabulka E.1: Testovací sada pro ovládání světel.

E.2 Monitorování světel

Věta	záměr	entita (hodnota)
Any light in kitchen?	GET_LIGHT	query (count) location (kitchen)
How many bedroom lights are there?	GET_LIGHT	query (count) location (bedroom)
What are names of lights in kitchen?	GET_LIGHT	query (list) location (kitchen)
Tell me the list of restroom lights?	GET_LIGHT	query (list) location (restroom)
List all the lights in kitchen?	GET_LIGHT	query (list) location (kitchen)
Are the lights off in bathroom?	GET_LIGHT	query (value) location (bathroom)
What is the level of kitchen light?	GET_LIGHT	query (value) location (kitchen)

Tabulka E.2: Testovací sada pro monitorování světel.

E.3 Monitorování teploty

Věta	záměr	entita (hodnota)
Kitchen temperature?	GET_TEMP	location (kitchen)
I wanna know bedroom temperature.	GET_TEMP	location (bedroom)
Get the restroom temperature.	GET_TEMP	location (restroom)
Temperature in bathroom?	GET_TEMP	location (bathroom)
Measure temperature in bedroom?	GET_TEMP	location (bedroom)
How hot is the bedroom now?	GET_TEMP	location (bedroom)
Measure the bathroom temperature?	GET_TEMP	location (bathroom)
Is the kitchen cold right now?	GET_TEMP	location (kitchen)

Tabulka E.3: Testovací sada pro monitorování teploty.

E.4 Monitorování vlhkosti

Věta	záměr	entita (hodnota)
Bathroom humidity?	GET_HUM	location (bathroom)
I wanna know bedroom humidity.	GET_HUM	location (bedroom)
Get the restroom humidity.	GET_HUM	location (restroom)
Humidity in kitchen?	GET_HUM	location (kitchen)
How wet is the bedroom now?	GET_HUM	location (bedroom)
Measure the bathroom humidity?	GET_HUM	location (bathroom)

Tabulka E.4: Testovací sada pro monitorování vlhkosti.

E.5 Ukončení dialogu

Věta	záměr	entita (hodnota)
Stop it!	CANCEL	-
Do not talk anymore!	CANCEL	-
Could you be quiet now?	CANCEL	-
Quiet, please!	CANCEL	-
I didn't say anything!	CANCEL	-

Tabulka E.5: Testovací sada pro ukončení dialogu.

E.6 Ukončení aplikace HomeVoice

Věta	záměr	entita (hodnota)
Bye Bye.	EXIT_APP	-
That's all for now.	EXIT_APP	-
Close App.	EXIT_APP	-
Quit.	EXIT_APP	-
Kill yourself.	EXIT_APP	-
Close HomeVoice.	EXIT_APP	-
Exit App.	EXIT_APP	-

Tabulka E.6: Testovací sada pro ukončení aplikace HomeVoice.

E.7 Informace o aplikaci HomeVoice

Věta	záměr	entita (hodnota)
What are your skills?	ABOUT	-
What can you help me with?	ABOUT	-
What are you good at?	ABOUT	-
What are you?	ABOUT	-
Tell me about you?	ABOUT	-
What is your purpose?	ABOUT	-

Tabulka E.7: Testovací sada pro získání informací o aplikaci HomeVoice.