VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

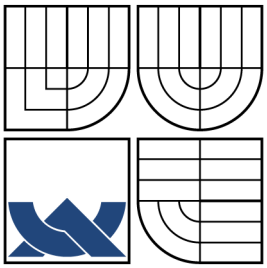
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

ŘÍZENÍ CHYTRÉ DOMÁCNOSTI S VYUŽITÍM PLATFORMY ALLJOYN

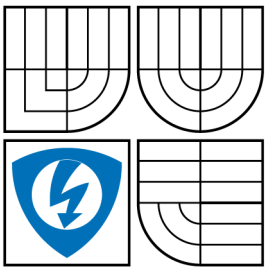
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE TOMÁŠ RESLER
AUTHOR

Brno 2015



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY
A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND
COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

ŘÍZENÍ CHYTRÉ DOMÁCNOSTI S VYUŽITÍM PLATFORMY ALLJOYN

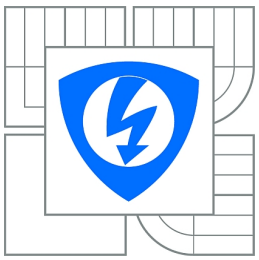
MANAGEMENT USING THE SMART HOME PLATFORM ALLJOYN

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — TOMÁŠ RESLER
AUTHOR

VEDOUCÍ PRÁCE — Ing. JIŘÍ HOŠEK, Ph.D.
SUPERVISOR

BRNO 2015



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Bakalářská práce

bakalářský studijní obor
Teleinformatika

Student:	Tomáš Resler	***ID:***	153610
Ročník:	3	***Akademický rok:***	2014/2015

NÁZEV TÉMATU:

Řízení chytré domácnosti s využitím platformy AllJoyn

POKYNY PRO VYPRACOVÁNÍ:

V rámci teoretické části bakalářské práce nastudujte komunikační platformu AllJoyn určenou pro integraci systémů domácí automatizace a inteligentních domácností v rámci jedné platformy. Praktická část práce zahrnuje kompilaci open-source frameworku AllJoyn od AllSeen Alliance pro vybraný IP směrovač s operačním systémem OpenWRT nebo embedded systém. Pro účely testování bude vyvinuta mobilní aplikace pro operační systém Android umožňující komunikaci s vybraným systémem chytré domácnosti.

DOPORUČENÁ LITERATURA:

[1] HERSENT, Olivier, David BOSWARTHICK a Omar ELLOUMI. The internet of things: applications to the smart grid and building automation. Hoboken: John Wiley, 2012, xxv, 344 s. ISBN 1119994357.
[2] GOODWIN, Steven. Smart home automation with linux and raspberry pi. Berkeley: Apress. ISBN 978-1430258872.

Termín zadání:	9.2.2015	***Termín odevzdání:***	2.6.2015

Vedoucí práce: Ing. Jiří Hošek, Ph.D.
Konzultanti bakalářské práce:

doc. Ing. Jiří Mišurec, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

ABSTRAKT

Bakalářská práce je věnována systému AllJoyn, který se snaží standardizovat rozhraní pro komunikaci v rámci chytré domácnosti. V teoretické části jsou popsány základní principy a struktura frameworku Alljoyn včetně nejnovějšího návrhu modulu Lighting framework pro ovládání světel a modulu Gateway Agent pro vzdálený přístup k AllJoyn zařízením. V praktické části je nejprve řešeno zprovoznění Alljoyn frameworku na hardwarovém směrovači se systémem OpenWRT, a to včetně světelného řídícího ovladače a modulu pro vzdálený přístup. Dále je naprogramován a zprovozněn konektor pro AllJoyn Gateway modul umožňující vzdálené ovládání světel přes službu Twitter. Činnost tohoto směrovače je otestována pomocí softwarového simulátoru Alljoyn Luminaire a programu pro ovládání světel LSF vyvinutého pomocí Lighting SDK pro mobilní zařízení se systémem Android.

KLÍČOVÁ SLOVA

AllJoyn, AllSeen aliance, OpenWRT, Světla, Internet věcí, D-Bus, Gateway, Agent, Twitter, Muzzley

ABSTRACT

This bachelor work concerns with system Alljoyn which tries to standardize the interface for communication in the frame of intelligent house. The basic principles and the structure of the framework Alljoyn including the recent proposal of the modules Lighting framework and Gateway Agent have been described in the theoretical oriented part of the work. The practical oriented part stars with the description of the getting the Alljoyn framework including the Lighting controller and the Gateway Agent to work on the hardware router provided with operating system OpenWRT. The following parts present own testing connector for the Alljoyn Gateway which have been written to allow the remote control of the lights via cloud service Twitter. The functionality of the router have been tested with software simulator of the Alljoyn lights Luminaire and an own ligths driving program developed utilizing the experimental Lighting SDK for mobile devices with operating system Android.

KEYWORDS

AllJoyn, AllSeen Alliance, OpenWRT, Lighting, Internet of Things, D-Bus, Gateway, Agent, Twitter, Muzzley

RESLER, Tomáš *Řízení chytré domácnosti s využitím platformy AllJoyn*: bakalářská práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2015. 59 s. Vedoucí práce byl Ing. Jiří Hošek, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Řízení chytré domácnosti s využitím platformy AllJoyn“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

(podpis autora)

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu bakalářské práce panu Ing. Jiřímu Hoškovi, Ph.D. a panu Ing. Pavlu Maškovi za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Brno ....

(podpis autora)

Faculty of Electrical Engineering
and Communication
Brno University of Technology
Purkynova 118, CZ-61200 Brno
Czech Republic

http://www.six.feec.vutbr.cz

PODĚKOVÁNÍ

Výzkum popsaný v této bakalářské práci byl realizován v laboratořích podpořených z projektu SIX; registrační číslo CZ.1.05/2.1.00/03.0072, operační program Výzkum a vývoj pro inovace.

Brno ....

(podpis autora)

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

OBSAH

Úvod **12**

1 Internet věcí **13**
1.1 Platformy, software 13
1.1.1 Home Gateway Initiative 13

2 AllSeen Aliance **15**
2.1 Struktura AllSeen 15
2.1.1 Pracovní skupina analytiky a telemetrie (Tellient) 15
2.1.2 Skupina vývoje základních služeb (Qualcomm) 15
2.1.3 Kompilační a certifikační skupina (Qualcomm) 16
2.1.4 Skupina pro ovládání světelných zdrojů (LIFX) 16
2.1.5 Skupina pro vývoj jádra (Qualcomm) 17
2.1.6 Skupina pro daty řízená API (Technicolor) 17
2.1.7 Skupina vývojových nástrojů (Qualcomm) 17
2.1.8 Skupina vývoje Gateway Agenta (Affinegy) 17
2.1.9 Pracovní skupina chytré domácnosti 17

3 AllJoyn **18**
3.1 Charakteristika AllJoyn platformy 18
3.1.1 Fyzická vrstva 18
3.1.2 Základní framework 19
3.1.3 Tenký klient 19
3.1.4 Servisní framework 19
3.1.5 Aplikační vrstva 19
3.2 Sdílená sběrnice - AllJoyn daemon 19
3.3 Podpora OS, architektur a programovacích jazyků 21
3.3.1 Architektury 21
3.3.2 Operační systémy 21
3.3.3 Programovací jazyky 22
3.4 AllJoyn SDK 22
3.5 Bezpečnost a šifrování 23

4 Framework pro ovládnání světelných zdrojů **25**
4.1 Architektura frameworku pro ovládání světelných zdrojů 25
4.2 AllJoyn SDK pro ovládání světelných zdrojů pro OS Android 26

5 AllJoyn Gateway Agent **27**
5.1 Charakteristika AllJoyn Gateway Agenta . . . 27
5.2 Konektor . . . 28
5.3 Řídící aplikace . . . 28
5.3.1 Managment Application . . . 29
5.3.2 Controller Application . . . 29
5.4 OpenVPN . . . 30

6 OpenWRT **31**
6.1 Stažení OpenWRT . . . 31
6.2 Feeds . . . 31
6.2.1 AllJoyn základní framework . . . 31
6.2.2 Servisní framework pro světla . . . 32
6.2.3 AllJoyn Gateway Agent . . . 32
6.3 Kompilace . . . 32
6.3.1 Doinstalace potřebných balíčků . . . 32
6.3.2 Přípava a tvorba obrazu . . . 33
6.4 Tvorba vlastního balíčku . . . 34
6.4.1 Cross-kompilace vlastního balíčku . . . 35
6.4.2 Instalace balíčku . . . 35
6.5 Flash systém . . . 35
6.5.1 Webové rozhraní LuCI . . . 36
6.5.2 Konzolový příkaz SCP . . . 36
6.6 Úpravy a konfigurace OpenWRT . . . 37
6.6.1 Rozšíření paměti extroot . . . 37
6.6.2 Swap oddíl . . . 38
6.6.3 Přístup a zabezpečení routeru . . . 38

7 AllJoyn aplikace **40**
7.1 Aplikace pro simulaci a práci se světly . . . 40
7.1.1 Luminaire . . . 40
7.1.2 LFS Sample App . . . 40
7.1.3 Tvorba vlastních Android aplikací . . . 42
7.2 Aplikace pro vzdálený přístup pomocí AllJoyn Gateway Agenta . . . 44
7.2.1 AllJoyn Sample Connector App . . . 44
7.2.2 AllJoyn Sample Gateway Controller . . . 44
7.2.3 Muzzley řešení pro vzdálený přístup . . . 44
7.2.4 Vlastní konektor . . . 45
7.3 Ukázkový příklad a jeho běh . . . 48

8 Závěr **50**

Literatura **51**

Seznam symbolů, veličin a zkratek **53**

Seznam příloh **55**

A Přílohy **56**

A.1 Rozchození systému z přeložených balíčků 56

A.2 Překlad vlastních obrazů a balíčků 56

A.3 Překlad muzzley konektoru 58

SEZNAM OBRÁZKŮ

2.1 Struktura AllSeen Alliance 16
3.1 Schéma systému AllJoyn 18
3.2 AllJoyn se sdílenou sběrnicí 20
3.3 Používání sběrnice AllJoyn 21
3.4 AllJoyn SDK 24
4.1 Světelný framework 25
5.1 Gateway Agent 27
5.2 Příklad AllJoyn Gateway Agenta 29
6.1 Kompilace 34
6.2 Webové rohraní Luci 36
6.3 Nastavení bezdrátového routeru 39
7.1 Luminaire 41
7.2 Ukázková kontrolní aplikace 41
7.3 Inistalace vývojového Android prostředí do Eclipse 43
7.4 Nalinkování AllJoynových knihoven 43
7.5 Muzzley 45
7.6 Mobilní aplikace Luminaire 49
7.7 Mobilní aplikace Twitter 49

SEZNAM TABULEK

3.1 Podporované jazyky základního rámce 22
3.2 Podporované jazyky aplikačního rámce 23

ÚVOD

Platforma AllJoyn se snaží o novou normu pro nově se vyvíjející IoT (Internet of Things) v chytré domácnosti a kancelářích. Pro úspěšné dosažení tohoto cíle je třeba nejen široké podpory jak jednotlivých systémových platforem a procesorových architektur, pro které je AllJoyn vyvíjen, ale též podpora firem daného odvětví, na nichž závisí, zda-li budou tento systém používat místo toho, aby každá z nich vyvíjela svoji vlastní platformu. Díky velkému množství firem, které se do AllSeen Alliance připojily, je velká naděje, že se podaří vizi obecně přijatého a používaného rámce naplnit. Iniciátorem aliance je společnost Qualcomm, avšak dnes mezi jejími členy najdeme řadu dalších významných společností, jako například společnosti LG, Pannasonic, Sony, TP-Link, Sharp, Cisco, HTC, Microsoft a mnohé další.

V rámci teoretické části byla popsána komunikační platforma AllJoyn, určená pro integraci systémů domácí automatizace a inteligentních domácností pod jednu platformu. V praktické části pak byla provedena kompilace upraveného frameworku AllJoyn pro světelné zdroje a Gateway Agenta od AllSeen Alliance na router TP-Link TL-WDR4300. Nad touto platformou byla pomocí SDK vyvinuta aplikace pro ovládání světel, které byly simulovány na mobilním zařízení pomocí volně dostupné aplikace pro operační systém Android. Dále pak byl přepsán a rozšířen ukázkový konektor, který implementuje možnost vzdáleného přístupu k ovládání světel pomocí služby Twitter.

V rámci bakalářské práce se bylo také nutné seznámit s OpenWRT systémem. OpenWRT je linuxový operační systém určený pro vestavěná zařízení (např. IP směrovače), který má velkou komunitní podporu. Pro tuto platformu byl posléze přeložen a upraven AllJoyn framework pro světelné zdroje a pro Gateway Agenta, a to ze zdrojových kódů dostupných ke stažení na stránkách AllSeen aliance. To vše je možné díky otevřenosti celé platformy OpenWRT i AllJoyn frameworku.

1 INTERNET VĚCÍ

IoT Internet of Things je založený na myšlence vzájemné komunikace různých elektrických zařízení a tím vytváření nových možností jejich využití. Tato zařízení mohou spolu komunikovat, sdílet informace nebo provádět různé úkony.

Jelikož žádná firma nemůže dosáhnout takové velikosti, aby její produkty zahrnovaly veškeré oblasti každodenního života, je potřeba univerzální platformy, která zajistí možnosti bezproblémové integrace a spolupráce jednotlivých zařízení a systémů různých výrobců [1].

1.1 Platformy, software

Platforem, pro které je vyvíjený IoT, je celá řada. Velké množství firem zabývající se určitým okruhem IoT si vytvořilo svůj vlastní software řešící jejich dílčí potřeby. Následně ale vzniká problém komunikace mezi těmito zařízeními navzájem. Proto nově vznikají společenství firem snažící se vytvořit univerzální otevřený systém, který bude pokrývat všechna odvětví a navzájem je propojovat. Jedním tímto společenstvím je i AllSeen Aliance [1] vyvíjející systém AllJoyn, který je podrobně rozebrán a řešen v této bakalářské práci, není však jediným. Dalším takovým společenstvím třeba Home Gateway Initiative [2].

1.1.1 Home Gateway Initiative

Home Gateway Initiative je nezisková organizace řešící specifikaci a standartizaci domácích bran (home gateways). Založily ji převážně firmy působící jako telekomunikační operátoři a to konkrétně Belgacom, BT (British Telecommunications), Deutsche Telekom, France Telecom, KPN (Koninklijke PTT Nederland), Teliasonera, Nippon Telegraph and Telephone (NTT), Telefonica a Telecom Italia. Postupně se přidali i další partneří a to například i takoví, kteří zároveň spolupracují na vývoji AllJoyn. Příkladem jsou firmy Cisco, Technicolor, LG a Qualcomm. Snahou je vytvořit jednotné univerzální API (Application Programming Interface) pro vývojáře vyvýjejcí aplikace na kontrolu a řízení domácích aplikací. Toto API by mělo být nezávislé na HAN technologii [3] a mělo by tak pomoci vytvářet vývojářům aplikace bez nutnosti znát fungování technologií ZigBee [4], Z-Wave [5], wireless, m-bus [6] a dalších.

[1] `https://allseenalliance.org/about/members`
[2] `http://www.homegatewayinitiative.org/`
[3] `http://www.hantechnology.com.sg/`
[4] `http://www.zigbee.org/`
[5] `http://www.z-wave.com/`
[6] `http://www.m-bus.com/`

Vývoj by měl být možný díky multiplatformě bez nutnosti zásahu do systému. Popis jednotlivých zařízení zde probíhá pomocí XML (Extensible Markup Language).

2 ALLSEEN ALIANCE

AllSeen aliance je neziskové sdružení, které se zabývá myšlenkou na propojení zařízení, systémů a služeb do IoT. Tohoto cíle chce dosáhnout pomocí univerzálního frameworku AllJoyn [1], který je vyvíjen ve velké technické komunitě pomocí ekosystému s otevřeným vývojovým prostředím. Aliance je pod hlavičkou Linux Foundation, jejímiž členy je několik set firem, jako např. Intel, IBM, Samsung, Qualcomm, HP, Cisco, Google, RedHat. Na vývoji systému Alljoyn se podílí více než 80 firem a každý den se jejich počet rozrůstá. Mezi nejvýznamnějšími lze zmínit například Qualcomm, LG, Sony, Microsoft, TP-Link, Panasonic, Silicon Image, Electrolux, Haier a TechnicolorBosch, Cisco, D-Link, HTC a Lenovo [1].

2.1 Struktura AllSeen

Aliance je v současné době rozdělena do devíti pracovních skupin. Jednotlivé skupiny mají za úkol návrh a vývoj určité části frameworku. Projektoví vývojáři každé skupiny si volí svého předsedu, který je zastupuje v technickém řídícím výboru. Nad tím vším ještě dohlíží správní rada viz obrázek 2.1. Většina ze skupin je iniciována vždy jednou velkou firmou, která následně koordinuje touto část vývoje a práci skupiny (uvedeno v závorce).

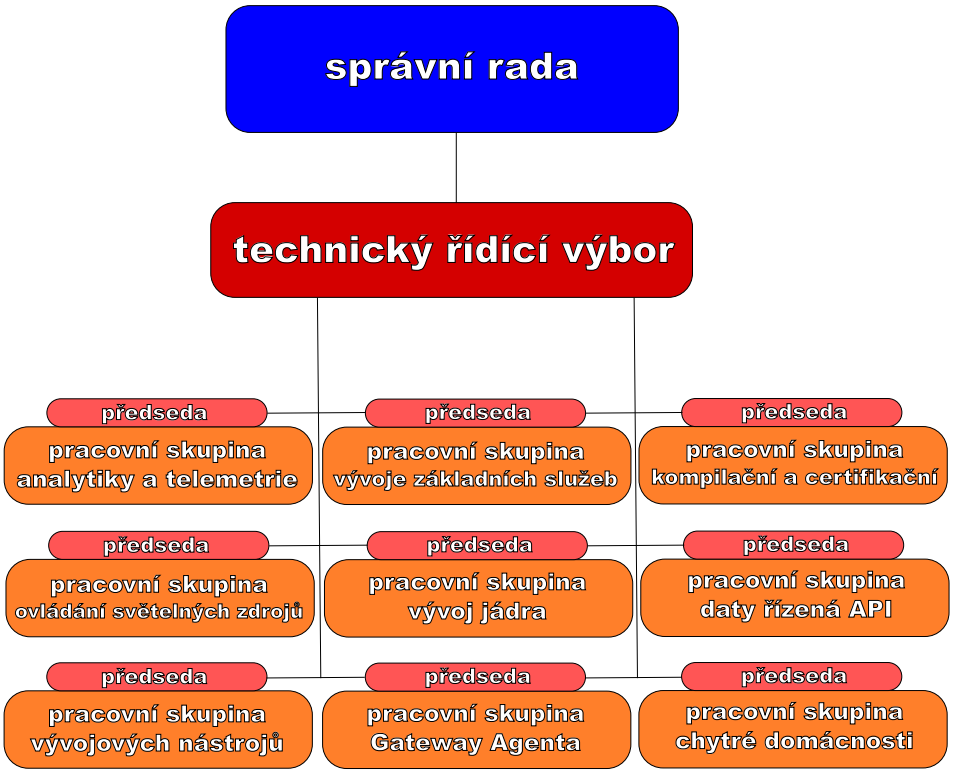
2.1.1 Pracovní skupina analytiky a telemetrie (Tellient)

Analytická skupina pracuje na vytvoření rozhraní a jeho open source referenční implementace umožňující výrobcům zjišťovat stav AllJoyn zařízení a systémů a shromažďovat statistická a jiná definovaná data o jejich činnosti. Tyto údaje jsou pomocí bezpečného protokolu přenášeny do analytického serveru [1].

2.1.2 Skupina vývoje základních služeb (Qualcomm)

Skupina základních služeb vyvíjí základní servisní rozhraní a funkce jako jsou notifikace, řídící panel, systém pro konfiguraci či systém pro umisťování zařízení do sítě. Tyto služby rozšiřují vlastní jádro frameworku AllJoyn o základní služby, které jsou využívány AllJoyn aplikacemi a zjednodušují jejich návrh a implementaci. Služby jsou součástí základu frameworku a další skupiny staví na těchto metodách svoje rozšíření [1].

[1] `https://allseenalliance.org/developers/download`



Obr. 2.1: Struktura AllSeen Alliance

2.1.3 Kompilační a certifikační skupina (Qualcomm)

Tato skupina připravuje směrnice, návody, software a nástroje pro certifikaci zařízení a softwaru potvrzující jejich kompatibilitu s platformou AllJoyn. Skupina vydává dokumentaci pro testování a specifikuje požadavky na Alljoyn zařízení [1].

2.1.4 Skupina pro ovládání světelných zdrojů (LIFX)

Hlavním úkolem této skupiny je vývoj sjednoceného rozhraní pro ovládání světel, a to jak pro světla samotná, tak pro aplikace a zařízení pro jejich ovládání. Při implementaci softwaru samotných žárovek (lamp) se počítá s využitím tenkého klienta. Rozhraní dokáže zapnout a vypnout žárovku, měnit barvu, odstín, saturaci, teplotu apod, a to jak pro jednotlivé žárovky, tak pro jejich skupiny. Tato kontrola má mnoho výhod pro uživatele, jako je například šetření energie, změna atmosféry prostředí podle nálady a komfort sjednoceného ovládání [1].

2.1.5 Skupina pro vývoj jádra (Qualcomm)

Skupina navrhuje a vyvíjí AllJoyn směrovač pro komunikaci zařízení mezi sebou v síti. Zabývá se jak problémy síťového propojení, tak věnuje velké úsilí zabezpečení spojení [1].

2.1.6 Skupina pro daty řízená API (Technicolor)

V této skupině je připravováno speciální API pro snadnou práci s datově řízenými událostmi. To se hodí například ve firmách při výrobě apod. API pracuje na vyšší úrovni abstrakce, než většina AllJoyn API. Od tohoto přívětivého rozhraní, v němž bude mít programátor snadnou práci při vývoji softwaru bez nutnosti bližšího poznání vlastního AllJoyn frameworku, si AllSeen Aliance slibuje rozšíření počtu vývojářů AllJoyn zařízení a celého AllSeen ekosystému [1].

2.1.7 Skupina vývojových nástrojů (Qualcomm)

Tato skupina vytváří softwarové nástroje, které budou usnadňovat návrh zařízení a vývoj sotware a pomáhat tak AllJoyn vývojářům a přispěvatelům [1].

2.1.8 Skupina vývoje Gateway Agenta (Affinegy)

Skupina zaměřující se na vývoj Gateway Agenta, umožňující standardní a bezpečnou metodu dálkového přístupu k AllJoyn zařízením a aplikacím z prostředí Internetu, tedy z prostředí mimo chráněnou domácí síť. Gateway umožňuje připojit domácí síť ke cloudovým službám, PAN (Personal Area Network) technologiím i k jiným sítím. Zároveň ale nadále zůstává možnost přímé komunikace s daným zařízením bez nutnosti komunikace přes cloud, pokud jsme s daným zařízením ve stejné síti [1].

2.1.9 Pracovní skupina chytré domácnosti

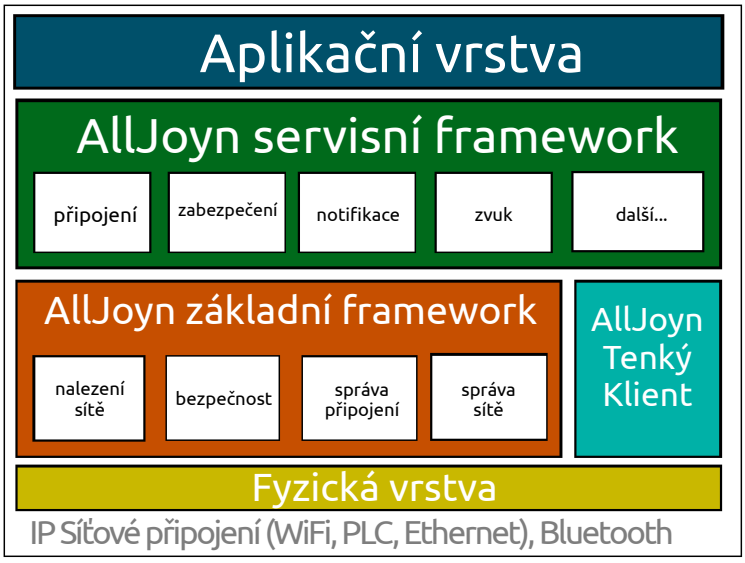
Vývojáři zde se zaměřují na centralizaci řízení domácnosti pomocí řídící základny, přičemž systém se skládá ze dvou součástí programového API, klientského API a serverového API. Zařízení s integrovaným serverovým API je funguje jako centrální řídící zařízení, které je pak v domácí síti řízeno zařízením s klientským API. API pro serverové zařízení je vyvíjeno tak, aby mohlo udržovat velký počet spojení s aplikacemi v domácnosti [1].

3 ALLJOYN

Jak již bylo v úvodu práce řečeno, systém AllJoyn si klade za cíl sjednotit různá proprietární řešení vznikající v oblasti Internetu věcí a tím v budoucnu zajistit interoperabilitu mezi řešeními různých výrobců HW (hardware) i SW (software). Důležitými aspekty celého návrhu je proto kvalitní, dobře specifikovaný a dostatečně obecný návrh rozhraní, který je nezávislý na konkrétním HW i SW prostředí či vývojovém jazyku a dále dostupná referenční implementace systému s otevřenou licencí [3].

3.1 Charakteristika AllJoyn platformy

Framework AllJoyn je specifikací a implementací distribuované sběrnice. Systém používá formátu zpráv D-Bus (Desktop Bus), který je dobře etablovaný na více systémech. D-Bus je sběrnicový systém zpráv, který umožňuje jednoduchým způsobem aplikacím spolu komunikovat [9]. AllJoyn projekt se tímto systémem inspiruje a dále ho rozšiřuje o možnost síťové komunikace při komunikaci aplikací v různých zařízeních. Základní rozdělení AllJoyn platformy lze vidět na obrázku 3.1.



Obr. 3.1: Schéma systému AllJoyn

3.1.1 Fyzická vrstva

Fyzická vrstva se stará o samotný transport dat pomocí různých komunikačních standardů, jako například Bluetooth, Ethernet nebo Wi-Fi. AllJoyn se ve svém

návrhu snaží být nezávislý na konkrétním současném či případném budoucím transportním mediu a protokolu. Proto je navržený modulárně tak, že o vlastní komunikaci s fyzickou přenosovou vrstvou se stará konkrétní modul, čímž dává do budoucna snadnou cestu k rozšiřování systému pomocí napsání a vložení nového modulu [2].

3.1.2 Základní framework

Základní framework se především stará o zpřístupnění AllJoyn sběrnice aplikacím. Funguje jako brána mezi aplikacemi na různých zařízeních. Mezi služby poskytované základním frameworkem patří např. správa sítě, oznamování zařízení v síti a prohledávání sítě, správa spojení v síti a bezpečnost připojení [2].

3.1.3 Tenký klient

Tenký klient je speciálně upravený základní framework pro zařízení, která mají málo paměti ROM (Read Only Memory) a RAM (Random Access Memory), a měly by tak problém provozovat kompletní základní framework. Jedná se především o koncová zařízení jako jsou programovatelná světla, lednička, termostat a další [2].

3.1.4 Servisní framework

Servisní framework je vrstva poskytující vývojářům prostředky pro jednodušší tvorbu aplikací. Využívá základní rámec a přidává k němu další moduly a API (Application Programming Interface), které je možno využívat při psaní uživatelských aplikací. Vytváří tak pro vývojáře bohatší API s připravenými funkcemi například pro připojení, notifikace, zabezpečení, zvuk a další [2].

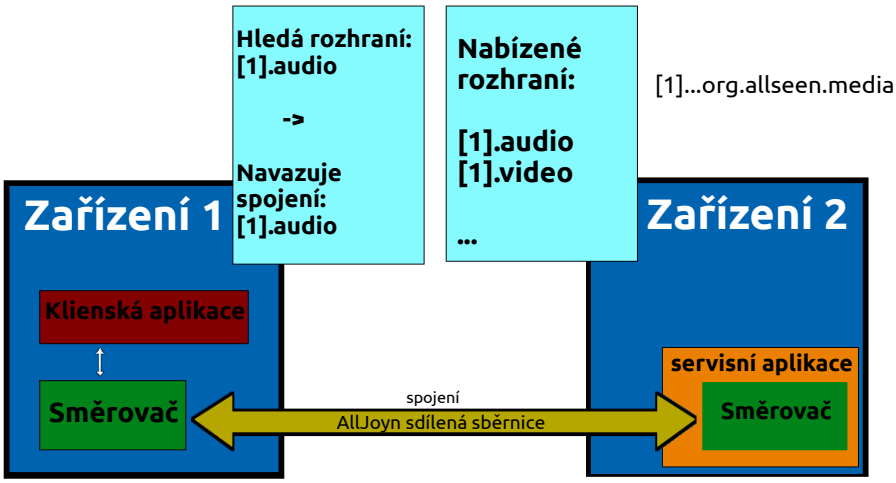
3.1.5 Aplikační vrstva

Aplikační vrstva představuje již vlastní programy pracující nad platformou AllJoyn, které umožní uživateli snadno pracovat se vzdálenými zařízeními podle programátorem určených schopností dané aplikace. Aplikace mohou být psány v různých programovacích jazycích a mohou být určeny pro různé operační systémy a typy zařízení. Podrobněji viz kapitola 3.3 [2].

3.2 Sdílená sběrnice - AllJoyn daemon

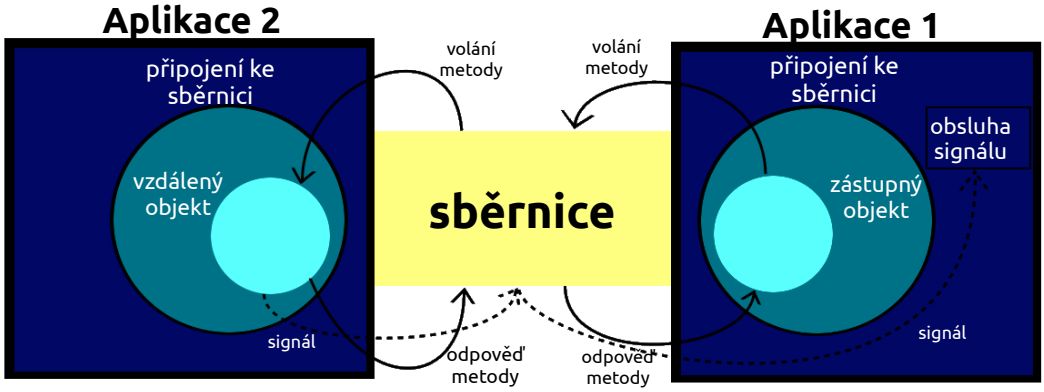
Framework AllJoyn pracuje na rozšířeném principu sdílené sběrnice. Ta zařizuje rychlou a jednoduchou cestu pro řídící systémové zprávy. Jak lze vidět na obrázku 3.2 sběrnice AllJoyn je jakási „dálnice“, pomocí níž tyto řídící zprávy proudí [3].

Fungování systému bude pro názornost demonstrováno na příkladu. Jsou zde umístěna dvě zařízení připojená do stejné sítě. Zařízení č.2 nabízí pomocí servisní aplikace rozhraní audio, video a další. Zařízení č.1 naopak pomocí servisní aplikace a zpráv hledá rozhraní audio. Nastane tedy shoda požadavků a aplikace následně navážou spojení pomocí všesměrové adresy. Mezi aplikacemi na zařízeních se následně vytvoří sdílená sběrnice pro využití audio rozhraní. Takto může komunikovat i více klientských aplikací s jednou servisní aplikací zároveň a naopak.



Obr. 3.2: AllJoyn se sdílenou sběrnicí

Na dalším obrázku 3.3 je po navázání spojení ukázána realizace vzdálené komunikace. Aplikace volající (č.1) si vytvoří vlastního zástupce objektu, se kterým chce komunikovat v aplikaci č.2. Př.: Komunikace pak už vypadá následovně. Je-li volána metoda, tak provede nejdříve změnu na zástupném („proxy“) objektu vytvořeném ve volající aplikaci. Změna je následně přenesena pomocí zprávy přes sdílenou sběrnici a aplikuje se na původní objekt v druhé aplikaci. Tato druhá aplikace následně pošle pomocí sběrnice odpověď, která způsobí vykonání změny opět v zástupném objektu první aplikace. Se signálem je to obdobné, až na to, že jde jen o jednocestnou událost. Jestliže nastane situace, která vyvolá signál v aplikaci č.2, pošle se do sběrnice zpráva odpovídající tomuto signálu, která je přijata a následně zpracována aplikací č.1, kde je na jejím základě vygenerován příslušný signál a vyvolána jeho obsluha v příslušné metodě („signal handler“).



Obr. 3.3: Používání sběrnice AllJoyn

3.3 Podpora OS, architektur a programovacích jazyků

AllSeen Alliance se snaží vytvořit standard pro chytrou domácnost, proto je zapotřebí široká podpora všech architektur a operačních systémů a s tím také související podpora různých programovacích jazyků pro psaní vlastních aplikací [3].

3.3.1 Architektury

Framework podporuje veškeré dnes běžně používané architektury. Jak tedy procesory x86 a x86_64, které dnes využívá řada moderních serverů a desktopů, tak architekturu ARM (Acorn RISC Machine) [2], která je dnes běžná především u tabletů a moderních mobilních telefonů. Dále je podporována architektura MIPS (Microprocessor without Interlocked Pipeline Stages), která je využívaná u vestavěných („embedded“) zařízení [3]. Celá platforma je ovšem navržena tak, aby závislost na konkrétní architektuře byla minimalizována a rozšíření platformy na další případně budoucí architektury bylo snadné [3].

3.3.2 Operační systémy

Referenční implementace frameworku je multiplatformní a lze ji použít na všech dnes běžně používaných operačních systémech Linux, OS X i Windows. Framework tak lze nainstalovat jak na desktopy a servery s těmito operačními systémy, tak samozřejmě i na směrovače založené na OpenWRT, mobilní telefony se systémem Android a iOS, mini-počítače Raspberry Pi, systémy RTOS (Real-Time Operating System) jako například ThreadX a podobně [2]. Opět je kladen důraz na lokalizaci systémově závislého kódu tak, aby případné rozšíření na další OS bylo co nejjednodušší.

3.3.3 Programovací jazyky

Podpora programovacích jazyků pro využití AllJoyn platformy se odvíjí od použitého systému, procesorové architektury a subsystému AllJoyn (základní nebo servisní framework).

Vlastní základní framework je napsán v jazyce C++, který spojuje možnost objektového programování s širokou podporou v různých prostředích. Tenký klient je pak napsán v jazyce C, který je vhodný pro programování zabudovaných zařízení, pro které je to často jediný dostupný vyšší programovací jazyk [3]. Podpora dalších programovacích jazyků je realizována pomocí obalovacích vrstev („wrapper“), které zpřístupňují C++ API v těchto jazycích. V současné době je tak dostupné API rozhraní pro jazyky Java, C#, Objective C a JavaScript [2].

Názvy rozhraní jsou tvořeny obrácenou doménovou metodou, jako známe například z jazyku Java. Příkladem může být jeden ze základních rozhraní `org.alljoyn.Bus`, který stanovuje některé základní vlastnosti realizace sdílené sběrnice na sběrnici. Název rozhraní je tedy řetězec znaků v relativně volné formě. To může způsobit problém, kdy například `org.alljoyn.fridge.chat` může mít úplně jiné vlastnosti, signály a metody, než například `org.alljoyn.mobile.chat` [3]. Lze však předpokládat, že dvě rozhraní `org.alljoyn.sample.chat.window.color` a `org.alljoyn.sample.chat.window.size` budou sloužit nadřazeně metodě `window`.

Rozdělení podporovaných jazyků lze nalézt v tabulce 3.1 pro základní framework a v tabulce 3.2 pro servisní framework.

Základní framework							
Operační systémy	C	C++	C#	Java	JavaScript	Objective C	WinJS
Linux	⊕	⊕	⊖	⊕	⊕	⊖	⊖
Windows	⊕	⊕	⊕	⊕	⊖	⊖	⊖
Windows 8	⊕	⊕	⊕	⊕	⊖	⊖	⊕
OS/X	⊖	⊕	⊖	⊖	⊖	⊕	⊖
Android OS	⊕	⊕	⊕	⊕	⊖	⊖	⊖
iOS	⊖	⊖	⊖	⊖	⊖	⊕	⊖
OpenWRT	⊖	⊕	⊖	⊖	⊖	⊖	⊖

Tab. 3.1: Podporované jazyky základního rámce

3.4 AllJoyn SDK

AllJoyn SDK (Software Development Kit) funguje na principu objektového programování. Každá aplikace může pomocí AllJoyn API vytvořit jeden nebo více objektů,

Servisní framework						
Operační systémy	C	C++	C#	Java	JavaScript	Objective C
Linux	⊕	⊕	⊖	⊕	⊖	⊖
Windows	⊖	⊖	⊖	⊖	⊖	⊖
OS/X	⊖	⊖	⊖	⊖	⊖	⊖
Android OS	⊖	⊖	⊖	⊕	⊖	⊖
iOS	⊖	⊖	⊖	⊖	⊖	⊕
OpenWRT	⊖	⊕	⊖	⊖	⊖	⊖

Tab. 3.2: Podporované jazyky aplikačního rámce

z nichž každý využívá jedno nebo více rozhraní [8]. Ten pak může být v síti nalezen a může s ním být navázáno spojení. Rozhraní každého objektu obsahuje prvky tří základních kategorií [8]:

- Metody - klasická interakce objektů.
- Signály - asynchroní upozornění na událost.
- Vlastnosti - nastavení a získání datových členů.

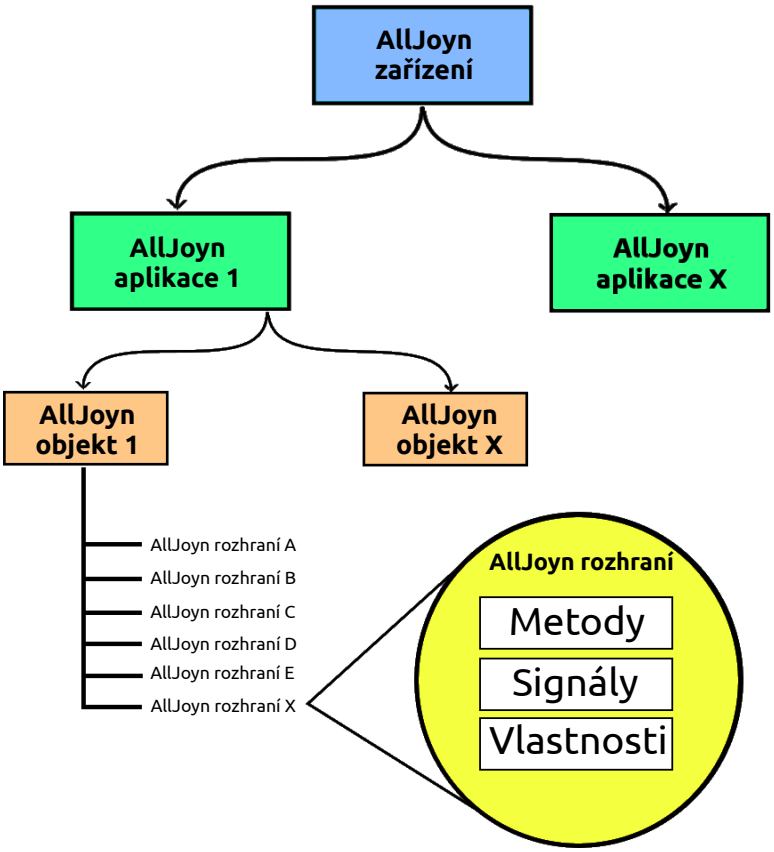
Názorně lze vidět na obrázku 3.4.

Při vlastní komunikaci systém vytváří a využívá tzv. zástupné objekty, jejichž prostřednictvím komunikace probíhá. Způsob jejich použití byl podrobněji popsán v kapitole 3.2.

3.5 Bezpečnost a šifrování

Na bezpečnost celého systému je při návrhu frameworku kladen velký důraz. Zabezpečení přístupu a komunikace vždy probíhá mezi aplikacemi, tedy nikoliv mezi zařízeními. Párování zařízení tak není nutné, pokud to nevyžaduje přímo ono zařízení nebo komunikační protokol (například Bluetooth) [3]. Je dovolené vytvořit různá zabezpečení i pro jednotlivá rozhraní aplikace. Zabezpečení umožňuje jak ověřování uživatele a přidělování práv, tak šifrování vzdálené komunikace. Přidělovaná práva mohou být rozdělena na zápis, spuštění a čtení. Zabezpečení, při němž jsou všechny zprávy zašifrovány, je použito jak při volání metod a zasílání jejích odpovědí, tak při posílání signálů. V AllJoyn platformě je též připravena úschovna klíčů („Key Store“)[2], kterou aplikace mohou použít, avšak její využití není povinné a lze k ukládání využít i jiných prostředků dostupných v daném systému. Pro ověřování identity jsou dostupné čtyři zabezpečovací metody[8]:

- PIN kód - použití klasického hesla.



Obr. 3.4: AllJoyn SDK

- Zjednodušený PIN kód - především pro aplikace postavené nad tenkým frameworkem, které nemají možnost dlouhých hesel.
- Přihlášení pomocí jména a hesla.
- Certifikace - použití ověření pravosti pomocí RSA (Rivest, Shamir, Adleman cryptology) klíčů.

4 FRAMEWORK PRO OVLÁDNÁNÍ SVĚTELNÝCH ZDROJŮ

V listopadu 2014 uvolnila AllSeen Alliance rozšíření AllJoyn platformy o rámec pro ovládání světel. Jde o doplnění frameworku o část API, která je speciálně určena k ovládání světel a obsahuje tak specializované objekty a metody pro tento účel [6]. Tento nový framework se snaží sjednotit doposud roztříštěné přístupy k ovládání světel napříč výrobci a operačními systémy [6]. V rámci zdrojových kódů jsou přiloženy i dvě ukázkové aplikace vyvinuté firmou Qualcomm viz kap. 7.1.

4.1 Architektura frameworku pro ovládání světelných zdrojů

Framework se skládá ze tří částí znázorněných na obrázku 4.1. Pravý blok znázorňuje ovladač žárovky. Jedná se o tu část systému, která zajišťuje příjem a zpracování došlých instrukcí a na jejich základě pak mění vlastnosti fyzických světel. Kromě toho zajišťuje ostatní potřebné podpůrné činnosti. Prostřední blok znázorňuje část implementující řídící ovladač. Tato část zajišťuje především nalezení žárovek a komunikaci s nimi. V ukázkové aplikaci je tento ovladač integrován v aplikaci Luminaire. Jedním z cílů této práce bylo integrovat tuto řídící část do bezdrátového routeru postaveném na OpenWRT s platformou AllJoyn. Poslední část je pak aplikace, pomocí které lze provádět ovládání světel a zjišťování jejich stavu. K tomuto účelu lze použít jak volně dostupnou vzorovou aplikaci, která je součástí frameworku nebo si lze naprogramovat aplikaci vlastní. V tom případě je dostupné SKD AllJoyn[1] (kap. 7.1.3) s framworkem na ovládání světelných zdrojů obsahujícím potřebné objekty a metody. Vyjít při psaní je možno z tutoriálu, který je dostupný na stránkách projektu[2].

Obr. 4.1: Světelný framework

[1] http://goo.gl/gyJaf2
[2] http://goo.gl/Hk874N

4.2 AllJoyn SDK pro ovládání světelných zdrojů pro OS Android

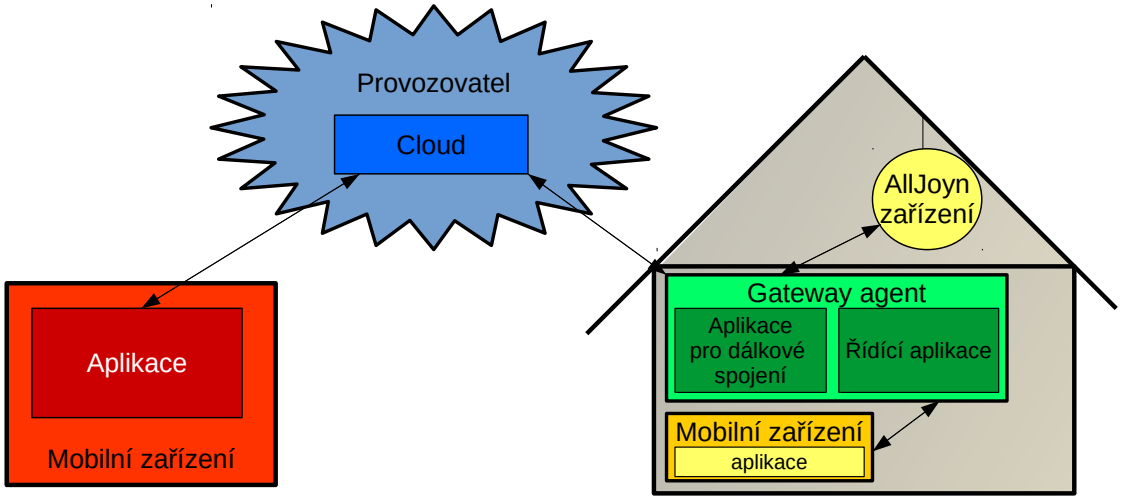
Pro vytvoření AllJoyn aplikace na ovládání světel připravila AllSeen aliance beta verzi SDK, která je dostupná ke stažení na adrese: `http://goo.gl/80mUE1`.

Balíček obsahuje jak potřebné knihovny, tak též LFSTutoriál[3] na tvorbu aplikací využívajících toto SDK pro ovládání světel. Podrobná instalace je rozebrána v kapitole 7.1.3.

[3] `http://goo.gl/Hk874N`

5 ALLJOYN GATEWAY AGENT

Gateway Agent poskytuje vzdálený přístup AllJoyn k zařízením a funkcím platformy AllJoyn z prostředí Internetu tj. mimo domácí síť. Jde především o funkce kontroly a notifikací, což následně umožňuje i vzdálenou kontrolu a správu domácnosti. Předpokládá se, že domácí síť je od vnějšího Internetu chráněna firewallem. Proto spojení zajišťuje gateway agent který se pomocí konektoru připojuje z vnitřní sítě k serveru či cloud systému viditelnému z Internetu. Tento způsob zajišťuje nejen funkčnost systému bez nutnosti úpravy firewallu nebo portů, ale i zvyšuje bezpečnost celého řešení. Agenta je možné nainstalovat například na jakémkoliv WiFi směrovači založeném na Linuxu (např. OpenWRT) či na automatických rozbočovačích [4]. Zjednodušeně si systém opět lze přiblížit na názorném schématu 5.1.



Obr. 5.1: Gateway Agent

5.1 Charakteristika AllJoyn Gateway Agenta

V pravé části obrázku 5.1 je schématicky znázorněna řízená domácnost. Pro zjednodušení uvažujme zařízení typu světel (případ této práce) komunikující přes WiFi směrovač s řídící aplikací a ovládané mobilním zařízením s k tomu určenou aplikací. Aby bylo možné se k systému vzdáleně připojit, je potřeba přidat na WiFi směrovač framework pro kontrolu a řízení dálkových spojení (Gateway managment aplikace) a v něm vytvořit konektor, který bude propojovat domácnost k vnějšímu serveru či cloudu. Pomocí kontrolní aplikace, která se připojí ke Gateway managment aplikaci lze, u správně napsaného konektoru, kontrolovat běh, změny a popřípadě měnit práva pro připojení ke sdílené sběrnici. Vzdálený přístup k mobilním zařízením s AllJoyn je řízený pomocí konfiguračního souboru uloženého na směrovači [4].

V levé části obrázku 5.1 je mobilní zařízení, které není přímo připojené k domácí síti, ale je připojeno k Internetu. Toto zařízení se může pomocí příslušné aplikace připojit k vybranému cloudu a prostřednictvím tohoto cloudu a spojení s Gateway Agentem může získat informace o domácích světlech a popřípadě provádět povolené změny.

Konkrétní příklad funkčnosti AllJoyn frameworku řešeným v této práci je znázorněn obr. 5.2. Především je tu směrovač TP-Link TL-WDR 4300 na kterém běží systém OpenWRT a na němž je nainstalován AllJoyn framework s rozšířením pro ovládání světel a pro vzdálené připojení. Konkrétně je celý systém realizován pomocí automaticky spouštěných programů AllJoyn daemon (základní AllJoyn sdílená sběrnice), Lighting controller service (správa pro kontrolu, vyhledávání a řízení světel) a AllJoyn Gateway Manament App (aplikace pro správu a řízení konektorů (kap. 5.3.1)). Na směrovači je dále nainstalován a spuštěn napsaný konektor, který je napojen na AllJoyn sběrnici a na Twitter. Ten pomocí obsluhy signálu („signal handler“) připojenému ke sběrnici kontroluje změny stavů žárovek. V případě, že se stav změní, vytvoří na Twitteru nový status v podobě speciální zprávy definovaného tvaru. Mimo to každých 15 vteřin stáhne poslední uživatelův tweet a pokud se jedná o tweet obsahující AllJoyn Lighting Comand (AllJoynLC), pošle příkaz ke změně stavu žárovky.

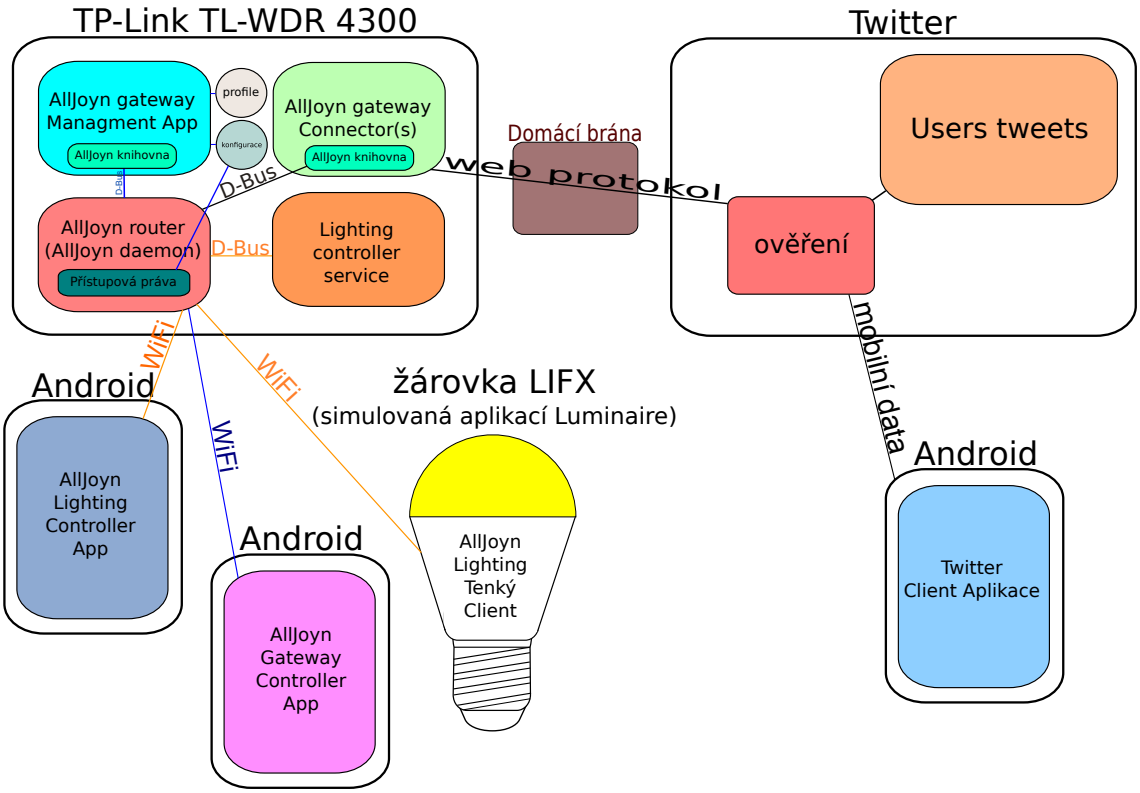
5.2 Konektor

Konektor zajišťuje vlastní spojení vnitřní AllJoyn sítě se serverem či cloudem dostupným z Internetu. Může to být jakákoliv aplikace, která využije AllJoyn knihoven k přístupu do sdílené sběrnice k předávání informací nebo příkazů. Takovýto jednoduchý konektor může být postaven na libovolném protokolu a není nutně potřeba využívat řešení Gateway Agent z AllJoyn frameworku. Takto je například v současné době řešen i konektor od firmy Muzzley (viz kap. 7.2.3).

Výhodou konektoru napsaného podle AllJoyn normy využívajícího Gateway Agenta je možnost řízení běhu, změn práv nebo automatického startování. Na to musí být v aplikaci definovány metody `shutdown`, `mergedAclUpdated` a `receiveGetMergedAclAsync` [7].

5.3 Řídící aplikace

Dálkové ovládání je řízeno pomocí konfiguračního souboru na směrovači. Tento konfigurační soubor je založen na definovaných uživatelských profilech. Kontrolní aplikace se tak spojí s řídící aplikací a pomocí uživatelských profilů v konfiguračním souboru



Obr. 5.2: Příklad AllJoyn Gateway Agenta

poskytuje přístup k jednotlivým zařízením, službám a k jejich ovládání. Profily jsou vytvářeny pomocí UNIX user ID [5].

5.3.1 Managment Application

Gateway Managment App je řídící aplikace běžící na směrovači, která zajišťujíce správu přístupových práv, profilů a řídí běh AllJoyn konektorů.

5.3.2 Controller Application

Controller Application je klientská část řídící aplikace. Spojuje se s Managment Application, a tak může uživatel řídit práva například pomocí aplikace v mobilním telefonu (kap. 7.2.2) nebo v PC.

5.4 OpenVPN

Pro srovnání je uveden ještě jiný projekt a to OpenVPN. Ten rovněž slouží k simulaci blízkého spojení zařízení napříč celosvětovým Internetem bez potřeby konfigurace vnějšího NAT (Network address translation). OpenVPN používá protokol SSL/TLS (Secure Sockets Layer/Transport Layer Security). Platforma je multiplatformní a to pro systémy Linux, Solaris, FreeBSD, Mac OS X, Windows a další. K ověřování se používá sdílený klíč nebo SSL certifikát. OpenVPN používá standardně protokol UDP (User Datagram Protocol) s portem 1194. Je však možno použít i protokol TCP (Transmission Control Protocol). Celý démon běží v uživatelském režimu a komunikuje pomocí TUN (Network TUNnel), který běží na 3. vrstě modelu ISO/OSI, anebo TAP (Network Test Access Point) režimu, který běží na druhé vrstvě a tak dokáže přenášet jakýkoliv typ dat.

6 OPENWRT

Jedním z cílů práce bylo nainstalovat na směrovač TP-LINK TL-WDR4300 distribuci OpenWrt rozšířenou o základní framework AllJoyn, včetně nového experimentálního modulu pro ovládání světel a frameworku pro AllJoyn Gateway Agenta s vlastním konektorem k Twitteru. Po instalaci je též potřeba zajistit, aby se tato služba automaticky spouštěla při každém startu směrovače.

6.1 Stažení OpenWRT

Nejlepším zdrojem informací a materiálů o OpenWRT jsou přímo stránky projektu [1]. Nejnovější podporovaná verze pro náš směrovač TP-Link TL-WDR4300 je Barrier Breaker 14.07. V konzoli je tedy potřeba vytvořit složku `barrier_breaker` a stáhnout z SVN repozitáře aktuální stabilní verzi pomocí příkazu:
`svn co svn://svn.openwrt.org/openwrt/branches/barrier_breaker`

6.2 Feeds

Pokud existuje připravená konfigurace tzv. feed , jedná se o nejjednodušší možnost, jak přidat balíčky třetích stran do OpenWRT. Přidání feed do distribuce OpenWRT pro překlad a instalaci se provádí přidáním řádku do souboru feeds.conf v hlavním adresáři staženého adresářového stromu barrier breaker, a to ve formě:
`src-git <návez> <url>;<prefix>`
Následující příkazy pak provedou stažení makefile souboru a balíček je tím připraven k překladu a integraci do výsledného obrazu systému:
`./scripts/feeds update -a`
`./scripts/feeds install -a`

6.2.1 AllJoyn základní framework

Při instalaci standardního AllJoyn frameworku lze postupovat dle následujících instrukcí. V konzoli se přesuneme do složky barrier_breaker (např. pomocí Mignight commanderu `mc` nebo příkazu `cd` pro přechod v adresářové struktuře). Obsah soubor feeds.conf.default zkopírujeme do souboru feeds.conf [11] a do tohoto souboru doplníme na první řádek odkaz na stažení instalačního balíčku AllJoyn (celé na jeden řádek):
`src-git alljoyn`
`https://git.allseenalliance.org/gerrit/core/openwrt_feed;`

[1] `http://wiki.openwrt.org/toh/tp-link/tl-wdr4300`

`barrier_breaker`

6.2.2 Servisní framework pro světla

V tomto případě lze místo instalace standardního AllJoyn frameworku použít upravený framework AllJoyn obsahující i modul pro ovládání světel. 13. března 2015 na neoficiálních stránkách začal vytvářet jeden z vývojářů [2] překlad AllJoynLSF pro OpenWRT Barrier Breaker pomocí feeds. Změnu provedeme výměnou specifikace za středníkem git-adresy místo obecného `barrier_breaker` za `feature/lighting`:

```
src-git alljoyn
https://git.allseenalliance.org/gerrit/core/openwrt_feed;
feature/lighting
```

Po stažení makefilů je třeba v této rané verzi udělat několik změn. Makefile soubory najdeme v podsložce `feeds/alljoyn/`. V prvé řadě je potřeba upravit verzi a dále opravit některé chyby, na které znemožňují kompilaci.

6.2.3 AllJoyn Gateway Agent

Stejně jako u servisního frameworku pro světla existuje komunitou připravený feed i pro překlad a instalaci Gateway Agenta do rozšířeného AllJoyn frameworku. Konfigurace opět spočívá v přidání definičního řádku do souboru feeds.conf a od předchozího případu se opět liší jen postfixem:

```
src-git alljoyn
https://git.allseenalliance.org/gerrit/core/openwrt_feed;
gateway/gwagent
```

I zde je potřeba pro úspěšnou kompilaci provést několik úprav v makefile souborech.

6.3 Kompilace

V následujících odstavcích bude popsáno, jak přeložit systém pro příslušnou architekturu z připravených zdrojových kódů, a jak jej nahrát do bezdrátového WiFi směrovače.

6.3.1 Doinstalace potřebných balíčků

Křížovou kompilaci provedeme na počítači se systémem Linux. Je nutné pro kompilaci použít UN*X (Unix-like system) systém a vzhledem k tomu, že překlad vyžaduje

[2] `https://build.allseenalliance.org/lighting/view/All/job/Lighting_Service_Framework_OpenWRT_BB_v14_12/`

diskový systém rozlišující velikost písmen, je též třeba překlad provádět na oddílu disku formátovaném systémem, který toto podporuje, tedy např. ext4 (fourth extended filesystem), XFS ('X' File System) a nelze použít diskový oddíl formátovaný např. NTFS (New Technology File System) či VFAT (Virtual File Allocation Table). V našem případě byla kompilace provedena na Ubuntu 14.04 LTS x86_64 a disku formátovaném na ext4. Nejprve je nutné zkontrolovat a případně doinstalovat potřebné vývojové balíčky:

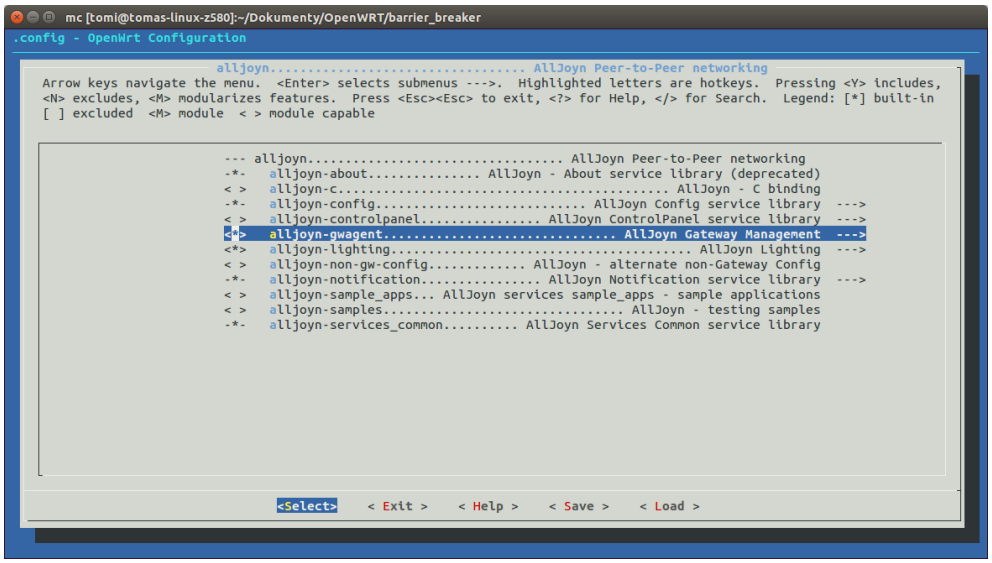
bash	bc	bcc	binutils
bin86	b43-fwcutter	build-essential	bzip2
ccache	fastjar	flex	gawk
gcc	gcj	genisoimage	gettext
git	git-core	g++	intltool
javac	jikespg	libboost1.53-dev	libgtk2.0-dev
libncurses5-dev	libssl-dev	libxml-parser-perl	libusb-dev
make	mercurial	subversion	patch
perl-modules	python2.6-dev	rsync	ruby
sdcc	unzip	util-linux	uudecode
wget	xsltproc	zip	zlib1g-dev

To lze provést v systému Ubuntu a dalších systémech založených na distribuci Debian například příkazem v konzoli:
`sudo apt-get install` „a názvy příslušných balíčků“ nebo pomocí některého grafického správce balíčků (např Synaptic).

6.3.2 Přípava a tvorba obrazu

Pomocí příkazu `make menuconfig` provedeného v hlavním adresáři `barrier_breaker`) lze spustit nástroj, pomocí kterého lze provést konfiguraci procesu kompilace. Při instalaci standardního AllJoyn framworku je třeba označit v sekci „Networking“ položku „alljoyn“. Označením pomocí mezerníku lze zajistit, aby se rámec přeložil a zahrnul do tvořeného obrazu. V případě Lighting frameworku nebo Gateway agenta je nutné rozkliknout podrobnější seznam v němž se objeví jednotlivé dílčí volby, a to včetně alljoyn-gwagent nebo alljoyn-lighting. Ty rovněž označíme mezerníkem a zahrneme tak do překladu a instalace (viz obrázek 6.1).

Po uložení upravené konfigurace a opuštění konfiguračního menu již lze provést kompilaci obrazu pomocí příkazu:
`make`
Pro případ, že potřebujeme podrobnější výpis pro hledání chyb a řešení problémů,



Obr. 6.1: Kompilace

použijeme vhodný parametr příkazu make:

```
make V=s
```

6.4 Tvorba vlastního balíčku

Pro účely práce je potřeba vytvořit balíček AllJoyn frameworku, který bude obsahovat jak rozšíření o modul řízení světel, tak gateway agent framework, a to z důvodu možnosti pozdějšího překladu gateway konektoru s napojením na knihovny frameworku řízení světel. Tento balíček je v současném stavu projektu třeba vytvořit zvlášť, přičemž lze rámcově postupovat podle návodu pro tvorbu jednoduchého vlastního balíčku dostupného na stránkách OpenWRT [3]. Nejdříve je nutno stáhnout potřebné zdrojové kódy ve stejné verzi a zabalit je do tar archivu se správnou adresářovou strukturou. Pro tento případ to znamená vytvořit tar s následujícím obsahem:

* core
  * alljoyn
    (https://git.allseenalliance.org/cgit/core/alljoyn.git/)
  * service_framework
    (https://git.allseenalliance.org/cgit/lighting/service_framework.git/)
* gateway

[3] `http://wiki.openwrt.org/doc/devel/packages`

        * gwagent
          (https://git.allseenalliance.org/cgit/gateway/gwagent.git/)
    * services
        * base
          (https://git.allseenalliance.org/cgit/services/base.git/)

Tento archiv je třeba pojmenovat `alljoyn-gwagent-<verze>-src` a přesunout do podsložky `dl`. Dále je nutno pomocí feeds stáhnout základní framework AllJoyn a k vzniklým makefilům v podsložce package/feeds/alljoyn přidat makefile alljoyn-gwagent [4].

6.4.1 Cross-kompilace vlastního balíčku

Kompilaci balíčku lze provést buď zahrnutím do tvorby systémového obrazu dříve popsaným postupem pomocí `make menuconfig` anebo samostatně pomocí příkazů:
`make package/<cesta k makefilu>/compile`
`make package/<cesta k makefilu>/install`.

6.4.2 Instalace balíčku

V případě dobře napsaného programu a úspěšně provedené kompilace se vytvoří v podadresáři `/bin/ar71xx/packages` balíček `<nazev>_ar71xx.ipk`. Tento balíček pak lze pomocí `scp` přenést do směrovače a po přihlášení na směrovači nainstalovat příkazem:
`ipkg install <název>.ipk`

6.5 Flash systém

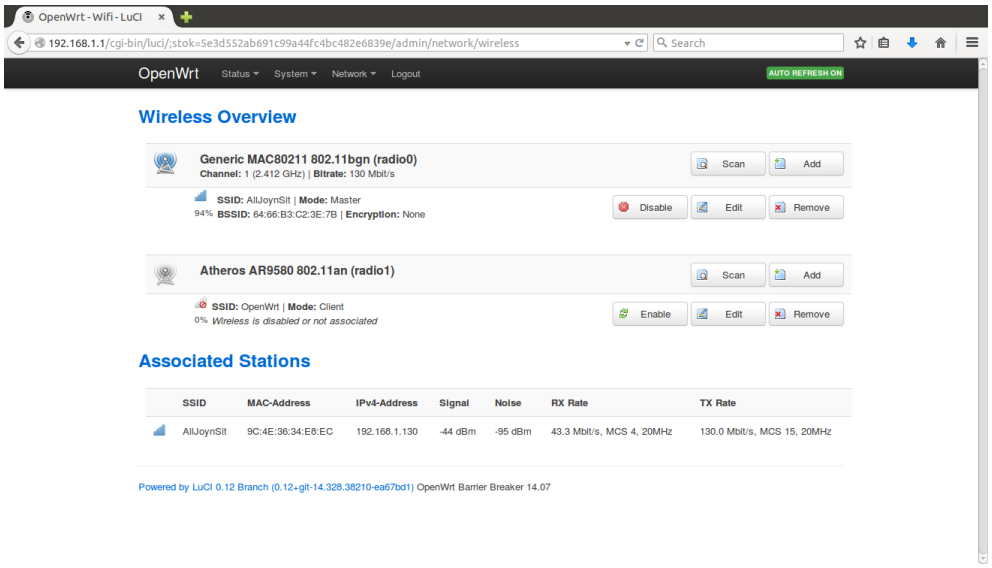
Po úspěšné kompilaci vzniknou ve složce `/barrier_breaker/bin/ar71xx/` překompilované verze pro různé typy a verze směrovačů. Pro účely této práce byl dostupný směrovač TP-Link WDR4300. Tomu odpovídá přeložený obraz `openwrt-ar71xx-generic-tl-wdr4300-v1-squashfs-sysupgrade.bin`. Velikost obrazu je přibližně 7.5Mb. Počítač, na němž je prováděna kompilace, je nutné připojit ke směrovači pomocí Ethernet kabelu a navázat spojení. Směrovač nesmí být zároveň připojený k Internetu (je tedy vhodné zrušit ostatní internetová spojení) a je třeba nastavit ručně IP adresu počítače na vhodnou adresu. Pokud je směrovač v původním nastavení, jeho IP adresa je `192.168.1.1` a maska `255.255.255.0`. Nastavíme tedy adresu počítače na jinou adresu tohoto adresního bloku.

[4] `https://github.com/allseenalliance/core-openwrt_feed/blob/master/alljoyn-gwagent/Makefile`

Nahrání obrazu do směrovače je možné provést více způsoby. V následujícím textu budou popsány dva z nich.

6.5.1 Webové rozhraní LuCI

Prvním způsobem je využití webového rozhraní Luci 6.2). Ve webovém prohlížeči



Obr. 6.2: Webové rohraní Luci

se lze připojit pomocí administrátorského účtu k adrese směrovače `http://192.168.1.1/`. Je možné se přesunout do `System -> Backup / Flash Firmware`, tam vybrat náš upravený firmware a spustit nahrávání. Po úspěšném dokončení procesu se směrovač odpojí a restartuje.

6.5.2 Konzolový příkaz SCP

Druhá popisovaná možnost využívá příkazových řádků v konzole. Nejprve pomocí příkazu `scp` lze zkopírovat obraz do směrovače. To je možné provést po přesunutí do složky s obrazem pomocí příkazu:
`scp /openwrt...factory-sysupgrade.bin root@192.168.1.1:/tmp`
přičemž bude vyžadováno zadání hesla uživatele root. Po úspěšném zkopírování obrazu se lze přihlásit pomocí příkazu ssh ke směrovači:
`ssh root@192.168.1.1`.
V konzole směrovače pak lze provést přepsání systému následujícím příkazem :
`mtd -r write /tmp/openwrt...factory.bin firmware`.
Opět následuje odpojení konsole a restart systému směrovače.

6.6 Úpravy a konfigurace OpenWRT

V případě vývoje a úpravy fungování směrovače může být zapotřebý některých změn nastavení směrovače. Je možné se setkat s potřebou o rozšíření vnitřní paměti směrovače, dále potřebu vytvořit swap oddíl v důsledku nedostatečné velikosti RAM, nebo změnu přístupů.

6.6.1 Rozšíření paměti extroot

Rozšíření vnitřní paměti není vždy nutnou záležitostí. TP-Link WDR4300 obsahuje poměrně velké množství volné paměti (128 MB paměti RAM). Pro vývoj však ani ta nemusí být dostatečná. Proto je možné celou paměť rozšířit např. o USB disk, který vložíme do USB směrovače. USB disk by měl být čistý a naformátovaný unixovým formátem ext4.

V systému OpenWRT nesmí chybět podporta usb a ex4. Pokud není, je nutné vytvořit nový obraz OpenWRT, ve kterém při konfiguraci pomocí make menuconfig je nutné zahrnout instalaci `kmod-usb-storage` (podpora USB) a `kmod-fs-ext4` (podpora EXT4). Příprava a tvorba obrazu je řešena v kapitole 6.3.2.

Směrovač je potřeba mít pro stáhnutí dalších balíčků připojen k Internetu. Pokud ovšem je na bráně adresa sítě schodná s tou nastavenou pro náš směrovač (192.168.1.1), tak dojde ke kolizy adres. Je tedy v takovém případě potřeba vnitřní adresu směrovače změnit na např. 192.168.2.1, viz kap. 6.6.3.

Dále je nutné doinstalovat balíček block-mount. To lze provést po přihlášení k směrovači příkazem ssh a pomocí příkazů „opkg update“ a „opkg install <balíček>“. Do připojeného usb-disku přehrajeme systémové soubory příkazem:
`tar -C /overlay -cvf - . | tar -C /mnt/sda1 -xf -`
a nakonfigurujeme připojení pomocí:
`block detect > /etc/config/fstab`
Doladění konfigurace souboru fstab lze provést v editoru vi:
`vi /etc/config/fstab`
Konkrétně upravením následující položky:
`option target '/overlay'`, `option delay_root '0'` a `option auto_mount '1'`.
Nakonec už stačí jen nakonfigurovat automatické připojení:
`/etc/init.d/fstab enable`,
poté připojit block a restartovat směrovač:
`block mount`
`reboot`

6.6.2 Swap oddíl

Může se také stát, že je potřeba vyřešit problém s nedostatečnou pamětí RAM, což lze pomocí konfigurace odkládacího oddílu swap, přičemž existuje více možností, jak tento oddíl zprovoznit. Nejprve je potřeba nainstalovat nezbytné balíčky:

```
ipkg install swap-utils
```

Následně je třeba vytvořit konfigurační UCI soubor (Unified Configuration Interface) v adresáři /etc/config/swap, který obsahuje:

```
config swap
option path '/tmp'
option filename 'swapfile'
option size '2000'
```

V /etc/init.d/swap je třeba vytvořit skript, který zajistí vytvoření swap oddílu při restartu OpenWRT. Skript je přiložen na CD. Následně je třeba změnit práva a oddíl nastartovat:

```
chmod +x /etc/init.d/swap
/etc/init.d/swap enable
/etc/init.d/swap start
```

Pokud je třeba změnit swap oddíl na místo na připojeném flash disku, tak to lze provést příkazy:

```
uci set swap.cfg1.path=/mnt/usb
uci set swap.cfg1.size=512000
uci commit swap
/etc/init.d/swap restart
```

6.6.3 Přístup a zabezpečení routeru

Jsou ukázány dva způsoby konfigurace routeru, a to jak nastavení webovým rozhraním Luci, tak pomocí příkazů v konzoli.

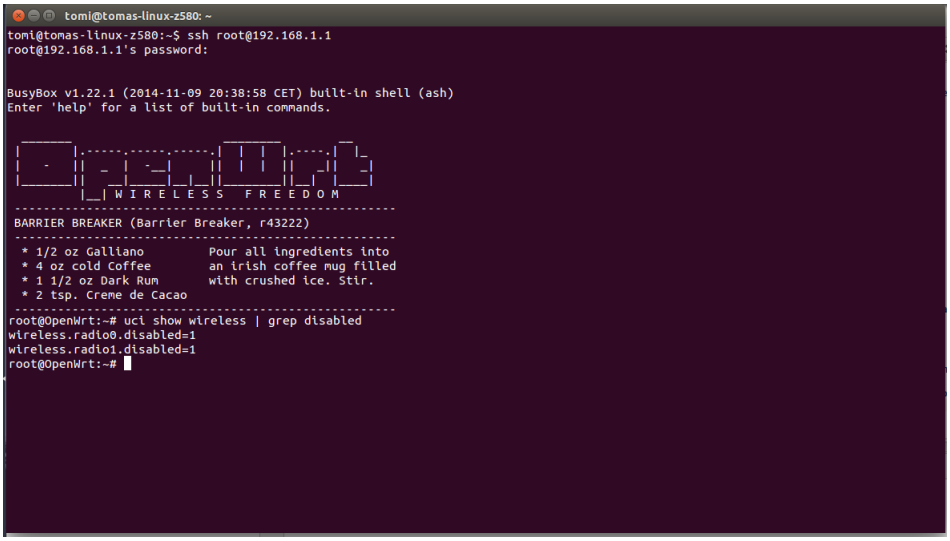
Luci

Po již dříve popsaném přihlášení k webovému rozhraní Luci je třeba kliknout v menu na `Network->Wifi` a zde tlačítkem `Edit` nastavit požadované vlastnosti. Ve spodní části nastavení je pak třeba uložit pomocí `Save & Apply`. Následně už jen tlačítkem `Enable` u položky `Wireless network is` můžeme WiFi aktivovat.

Konzole

Prvně je třeba se připojit pomocí ssh administrátorským účtem na naš směrovač. To provedeme jako v předchzím odstavci v konzoli příkazem:
`ssh root@192.168.1.1`
Po vyplnění hesla jsme připojeni. WLAN (Wireless Local Area Network) je primárně po přehrání nové verze firmwaru vypnutá. Stav bezdrátové sítě lze zjistit příkazem:
`uci show wireless | grep disabled`
Ten zobrazí stav jednotlivých WiFi rozhraní. V případě našeho směrovače uživatel obdrží dvě položky ve stavu „disabled“, viz obrázek 6.3. Pro zapnutí je potřeba tyto položky přepnout do stavu aktivního stavu, což provedeme příkazem:
`uci set wireless.radio[x].disabled=0`,
kde „x“ nahradíme číslem příslušného rozhraní 0 nebo 1.

Dále je potřeba nastavit jméno sítě, způsob zabezpečení a heslo. K tomu slouží



Obr. 6.3: Nastavení bezdrátového routeru

příkazy:
`uci set wireless.@wifi-iface[x].ssid=AllJoynSit`
`uci set wireless.@wifi-iface[x].encryption=psk2`
`uci set wireless.@wifi-iface[x].key=„AllJoyn"`
`uci commit wireless`
`wifi`.

7 ALLJOYN APLIKACE

Vývojáři AllSeen Aliance bylo vytvořeno několik ukázkových aplikací pro fungování jednotlivých částí frameworku. Jde nejčastěji o aplikace napsané v jazyce C++, dále pak jsou to aplikace pro OS Android a iOS. Jsou dostupné některé testovací aplikace i pro další podporované programovací jazyky. V následujících odstavcích popíšeme některé ukázkové aplikace použité při řešení této bakalářské práce.

7.1 Aplikace pro simulaci a práci se světly

Nejkomplexnější ukázková aplikace pro simulaci a ovládání světel se jmenuje Luminaire a pochází od firmy Qualcomm. Je napsaná pro operační systém Android verze 4.0 výše a pro systém iOS. Tato aplikace není ovšem jediná. Další příklady napsané v C++ lze najít v adresáři rozšířeného frameworku. To se bude například hodit pro tvorbu vlastního konektoru. Dále je dostupný ke stažení AllJoynLSF tutoriál, který popisuje tvorbu vlastních aplikací pro systém Android.

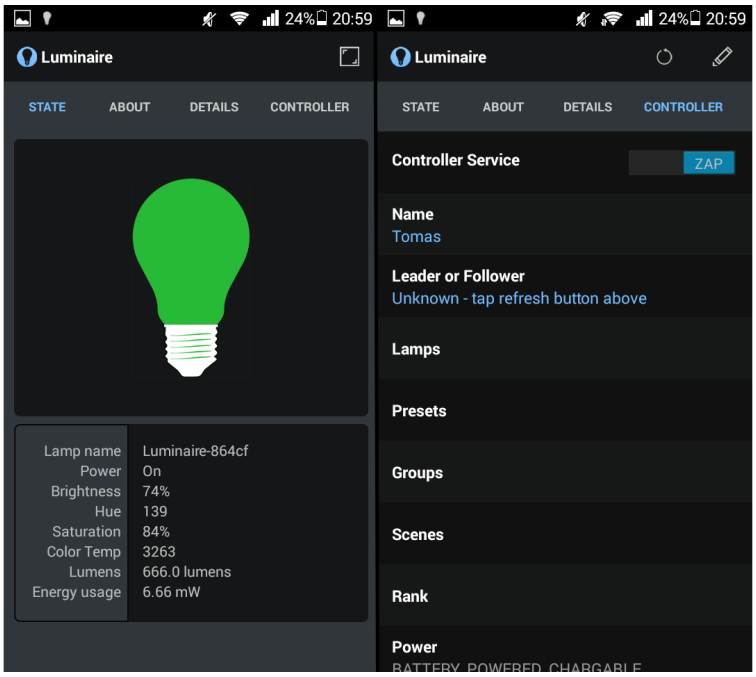
7.1.1 Luminaire

Firma Qualcomm vytvořila ukázkovou aplikaci Luminaire [1] (obrázek 7.1), která je dostupná pro systémy Android a iOS a která simuluje inteligentní světla podle AllJoyn normy na obrazovce mobilního telefonu [6]. Tato aplikace v sobě též zahrnuje volitelnou možnost spuštění Lighting Controller Service. Lze na okraj poznamenat, že fyzická světla implementující framework Alljoyn již lze též zakoupit, a to například v obchodě LIFX [2].

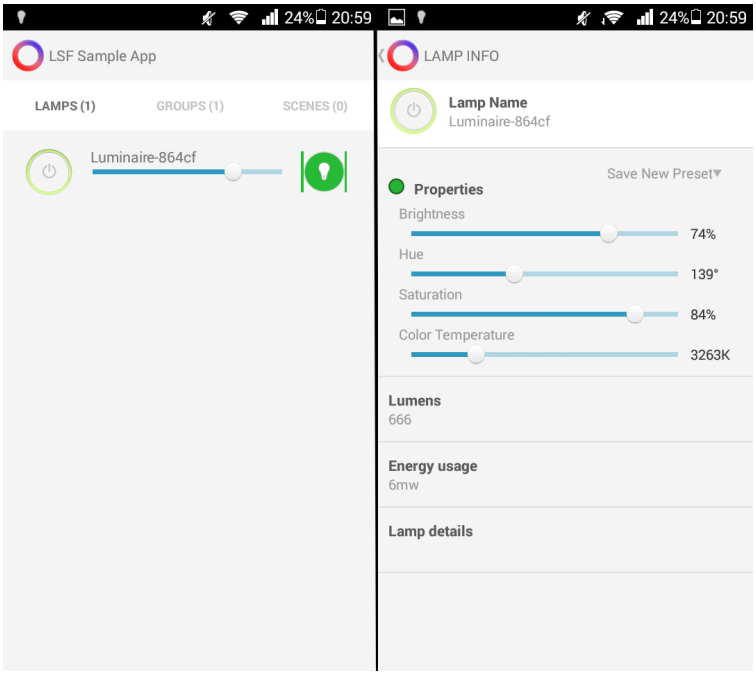
7.1.2 LFS Sample App

Aplikace („LSF Sample App“ (obrázek 7.2)), kterou lze najít na citovaných stránkách buď v podobě zdrojových kódů [3] anebo v podobě instalačního balíčku pro Android [4], nám představuje ukázkovou řídící aplikaci. Při současném spuštění obou aplikací, a jejich připojení do stejné sítě WiFi, aplikace zobrazí nalezenou žárovku, kterou pak lze pomocí řídícího programu ovládat. Je možné například nastavovat intenzitu, barvu, saturaci nebo teplotu, lze též vytvářet různé scény, skupiny žárovek a efekty.

[1] `https://play.google.com/store/apps/details?id=com.qualcomm.luminaire`
[2] `http://store.lifx.co/`
[3] `http://goo.gl/LCC4oY`
[4] `http://goo.gl/Y5VewC`



Obr. 7.1: Luminaire



Obr. 7.2: Ukázková kontrolní aplikace

7.1.3 Tvorba vlastních Android aplikací

Programování aplikací pro Android s využitím Alljoyn frameworku předpokládá, že je nakonfigurované standardní prostředí pro programování aplikací pro systém Android. Dále bude popsán postup konfigurace s využitím vývojového IDE prostředí Eclipse. To je ke stažení na stránkách projektu: `http://www.eclipse.org/downloads/`.

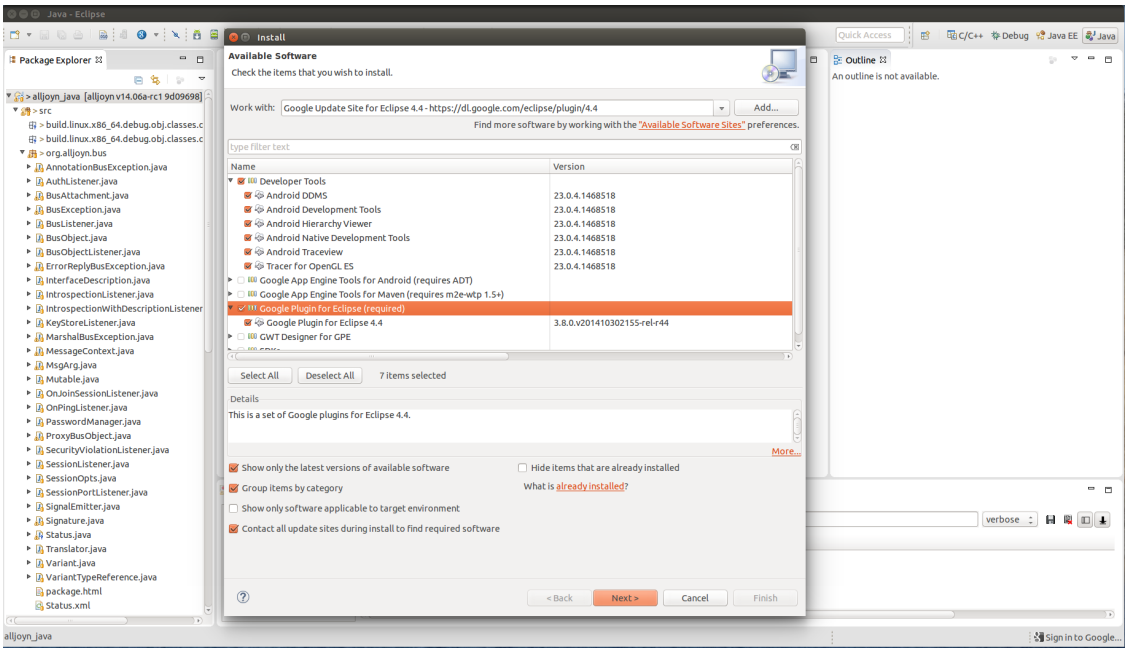
Před samotnou instalací Eclipse je třeba zkontrolovat verzi prostředí java. V terminále se to provede příkazem: `java -version`. Pokud je Java v pořádku, pomocí příkazu `cd` se přesuneme do složky se staženým tar archivem Eclipse a ten rozbalíme příkazem `tar xvzf <jmeno archivu>`. Tato akce vytvoří složku, kde již bude spouštěč programu.

Do prostředí Eclipse je po nainstalování a spuštění nejdříve potřeba stáhnout a nalinkovat ADT (Android Development Tools). V liště hlavního menu programu je třeba vybrat `Help->Install new Software`. Otevře se nabídka, kde v pravém horním rohu je možné tlačítkem `Add` přidat nový software, ten je třeba nazvat např. ADT Plugin a do url adresy vložit: `https://dl-ssl.google.com/android/eclipse/`.

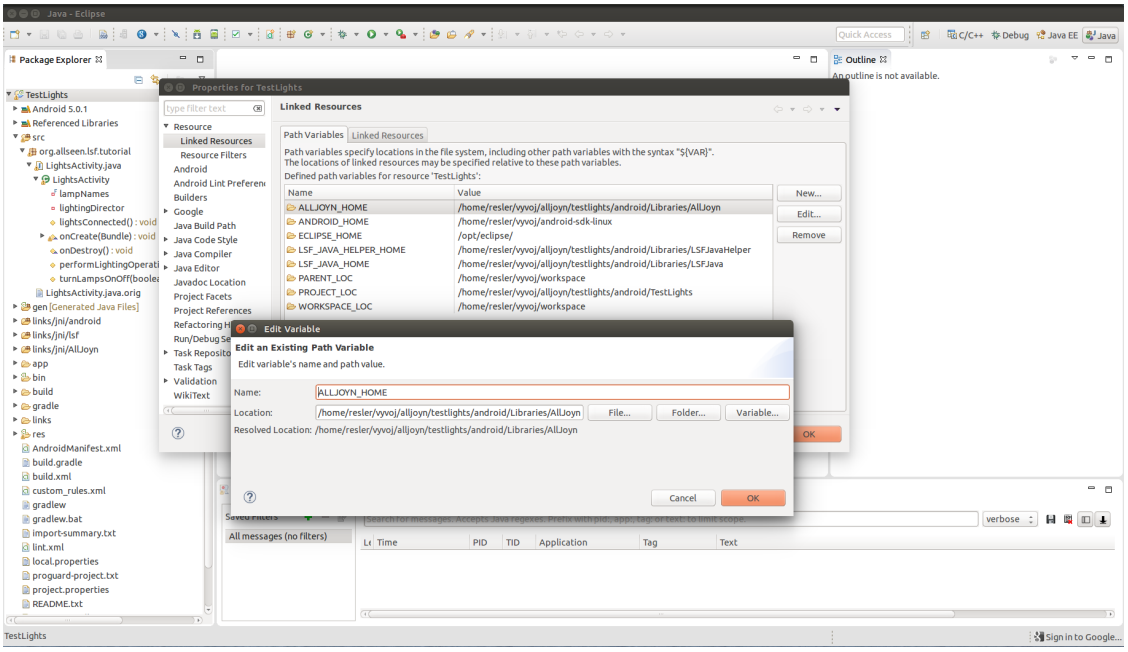
Tlačítkem OK se zobrazí seznam balíčků k instalaci (viz obr. 7.3). Po odsouhlasení podmínek proběhne stažení a instalace potřebných modulů a restart prostředí Eclipse. Konfigurace dále pokračuje stažením Android SDK. Pod hlavní lištou je třeba využít nové ikony Android SDK manager, po jejímž stisknutí je lze vybrat instalaci Android SDK Tools, Android 5.0 (API 21) a Android Support Library. Tlačítkem v levém dolním rohu instalaci spustíme [10].

Nyní již lze přistoupit k načtení LSFTutoriálu [5]. V Package Exploreru se pravým tlačítkem myši zobrazí dialog `Import...` a v něm je třeba vybrat `General->Existing Projects into Workspace`. V dalším okně v položce Select root directory je třeba vybrat cestu k rozbalenému LFS Tutoriálu. Tím se projekt LFSTutoriálu přidá do Eclipse. V projektu je dále zapotřebí nastavit cesty k Lighting SDK. To se provede v nastavení projektu. Vybráním `Resource->Linked Resources` a v záložce `Path Variables` je třeba opravit nebo přidat cesty pro proměnné `ALLJOYN_HOME`, `LFS_JAVA_HOME` a `LSF_JAVA_HELPER_HOME` (viz obr. 7.4), které je nutno nastavit do příslušných podadresářů adresáře, v němž je rozbalen Lighting SDK pro Android.

[5] `http://goo.gl/80mUE1`



Obr. 7.3: Inistalace vývojového Android prostředí do Eclipse



Obr. 7.4: Nalinkování AllJoynových knihoven

7.2 Aplikace pro vzdálený přístup pomocí AllJoyn Gateway Agenta

Ukázkové aplikace pro chod AllJoyn Gateway Agenta jsou napsané především v C++ tak, aby je bylo možné přeložit na desktopových systémech a na směrovačích. Dále pak je zahrnuta ovládací aplikace pro OS Android.

7.2.1 AllJoyn Sample Connector App

Ve frameworku je obsažena i ukázková aplikace pro spojení vnitřní sítě s uživatelským účtem na Twitteru. Konektor je napsaný v C++ a ke spojení s Twitterem používá bash skripty. Tyto skripty používají RestApi Twitteru a přihlášení pomocí generovaných hash kódů. Konektor má za úkol sledovat stav sdílené sběrnice ve vnitřní síti a v případě zaregistrování AllJoyn notifikace tuto notifikaci pošle na uživatelskou zeď jako tweet. Jde tedy jen o ukázkový jednostraný přenos.

7.2.2 AllJoyn Sample Gateway Controller

Ukázková kontrolní aplikace, která se spojuje s Gateway managment aplikací a může tak vypínat, zapínat konektory nebo jim dávat práva je napsaná pro systém Android. Dostupná je na stránkách projektu [6].

7.2.3 Muzzley řešení pro vzdálený přístup

Muzzley je společnost zabývající se IoT a sídlící v San Francisku a Lisabonu. Byla založena roku 2012 s cílem vytvořit jednotnou aplikaci pro ovládání všech bežně známých a dostupných chytrých zařízení. Od roku 2014 je rovněž členem AllSeen aliance. Jako první cíl s AllJoyn platformou bylo rozhodnuto spojit a vytvořit vzdálené ovládání pro světla podle této normy. První verze ovšem nevychází z AllJoyn Agenta, avšak podle informací vyplývajících z osobní komunikace s pracovníky firmy má firma v plánu toto změnit v blízké budoucnosti a a řešení plně postavit na Alljoyn frameworku. Současná ovládací aplikace Muzzley je dostupná pro systémy Android, iOS a Windows Phone. Na stránkách github [7] je návod k zprovoznění Muzzley konektoru na OpenWRT a pro spojení s cloud řešením. V této rané fázi je nutné mít developerský účet u firmy Muzzley k zapnutí a zobrazení kanálu AllJoyn Lighting v aplikaci.

[6] `https://wiki.allseenalliance.org/gateway/gatewayagent`
[7] `https://github.com/muzzley/alljoyn-muzzley-connector`

Obr. 7.5: Muzzley

7.2.4 Vlastní konektor

Jedním z cílů této práce bylo ukázat praktickou možnost ovládání světel z prostředí Internetu mimo vnitřní domácí síť. K tomu je potřeba napsat konektor, který bude zajišťovat komunikaci mezi sdílenou sběrnicí Alljoyn poskytující informace o stavech světel a přenášející příkazy k jejich ovládání a mezi vnějším serverem či cloudem dostupným z Internetu. Jako východisko pro případ této práce byl použit ukázkový konektor, který byl následně přepsán do potřebné podoby. Je třeba tak upravovat balíček a navazovat tak na práci popsanou v kap. 6.4 - kompletní zdrojové kódy konektoru jsou přiloženy na CD.

V hlavní sekci main je nutné vytvořit vlastní sběrnici „bus“, kterou pak lze připojit ke sdílené sběrnici „bus.Connect“.

```
BusAttachment bus("ConnectorApp", true);
QStatus status = bus.Start();
status = bus.Connect();
```

V další sekci je třeba definovat bash skripty pro stažení posledního a pro vytvoření nového tweetu. V ukázkovém konektoru je pak sekce, kde se startuje notifikační handler, který v tomto příkladě není potřeba. Proto je celá zakomentovaná. V tomto příkladu je potřeba vytvořit nový handler, který bude napojen k „Lighting Controller Service“ a bude hlídat informace o stavu žárovek.

```
ControllerClientCallbackHandler controllerClientCBHandler;
LampManagerCallbackHandler lampManagerCBHandler;
ControllerClient client(bus, controllerClientCBHandler);
LampManager lampManager(client, lampManagerCBHandler);
ControllerClientStatus cstatus = client.Start();
    while (!connectedToControllerService) {
        printf("cekam");
        sleep(1);
    }
cstatus = lampManager.GetAllLampIDs();
cstatus = CONTROLLER_CLIENT_OK;
```

Vlastní třídu „LampManagerCallbackHandler“ najdeme mimo hlavní třídu. Obsluhu událostí zde zajišťuje metoda „LampStateChangedCB“, která je vykonávana právě při změně stavu žárovky. Podmínka v metodě zajišťuje, aby případná změna stavu žárovky iniciované z vnější sítě nevyvolala zápis nového statusu. Protože využitá služba Twitter má omezenou velikost zprávy, je potřeba vhodně volit její formát. Navržený formát tedy vypadá následovně:

„`AllJoynL<prikaz> lampID=<id> onOff=<stav> hue=<stav> saturation=<stav> colorTemp=<stav> brightness=<stav>`“

- **AllJoynL**: Určuje jestli se jedná o stav světla (AllJoynLS) nebo command pro provední změny žárovky(AllJoynLC).
- **lampID**: Definuje ve formátu uint32_t přesné ID světla, kterého se status týká.
- **onOff**: Světlo může být ve stavu zapnutém (1) nebo vypnutém (0).
- **hue**: Přiděluje světlu barvu. Hodnoty nabývají hodnot (stupňů) od 0 do 360.
- **saturation**: Sytost barev udaná v procentech od 0 do 100.
- **colorTemp**: Určuje teplotu světla. Je použito převodu na Kelviny a může nabývat hodnot od 2700 do 5500.
- **brightness**: Na závěr základní hodnota a to jas. Bez této hodnoty se neprojeví žádná z předchozích hodnot. Je opět v procentech od 0 do 100

Po sestavení této zprávou se vytvoří příkaz pro spuštění skriptu pro odeslání zprávy na Twitter se zprávou jako parametrem.

`sh <cesta>/postTweet.sh \<zprava>\`

Tímto je vyřešeno posílání zprávy o změně stavu, tedy směr z vnitřní sítě ven. Znamená to, že pokud nastane nějaká změna nastavení světel, ihned se vytvoří nový tweet.

Zbývá vytvoření propojení v opačném směru, tedy programu pro načtení statusu

Twitteru s údaji o požadované změně stavu světel a jejich převodu do Alljoyn příkazů na změnu stavu žárovky. Zde byla nejprve přepsána hlavní (main) část programu, a to tak, že se každých 15 vteřin provádí cyklus testující existenci nového statusu patřičného formátu.

```
while (!exitManager.isExiting()) {
...
sleep(15);}
```

Dále by vytvořen skript getTweet.sh, pomocí kterého lze stáhnout poslední tweet a uložit jeho obsah do proměnné tweetText. Tento skript je vyvoláván uvnitř cyklu v metodě main.

```
char twScript[] = "getTweet.sh";
tweetGetScript = "/opt/alljoyn/apps/dummyapp1/bin/"+twScript;
qcc::String tweetText = execScript(tweetGetScript.c_str());
```

Text tweetu lze poté parsovat a rozdělit do vektoru podle jednotlivých proměnných. Je třeba zkontrolovat jestli se jedná o příkaz na změnu stavu žárovky (AllJoynLC). Pokud ne, program čeká opět 15 vteřin na opakování cyklu a provedení další kontroly statusu. Pokud se jedná o příkaz AllJoynLC, signalizační proměná „zmenalampy“ se nastaví na true. Tato proměnná zajistí, aby při následujících postupných změnách parametrů lampy nevyvolala každá změna vytvoření oznamovacího statusu na Twitteru. Je třeba, aby se vytvořil pouze jediný finální potvrzovací status po úspěšném provedení veškerých požadovaných změn. V následném cyklu se bude tedy procházet vektor paramentrů a každou část bude parsovat a získá tak jméno proměnné a její hodnotu, kterou případně vhodně přetypuje a převede. Následně se pak použijí metody AllJoyn Lighting API a ty vyvolají požadovanou změnu vlastností příslušného světla. Před poslední úpravou pak je třeba změnit signalizační proměnou „zmenalampy“ zpátky na false, tak aby s poslední změnou se též vytvořil a odeslal nový status na Twitter.

```
zmenalampy = true;
for (int i=2; i<tokSize; i++) {
  s1 = strtok(tok[i], "=");
  s2 = strtok(NULL, "=");
  ...
  if ( onOffstrcmp == 0) {
    onOff = atoi(s2);
    lampManager.TransitionLampStateOnOffField(lampId,onOff);
  } else if ( huestrcmp == 0 ) {
  ...
```

```
    zmenalampy = false;
    brightness = atoi(s2);
    lampManager.TransitionLampStateBrightnessField(lampId,
                                maxuint_32_t/100*brightness, 100);
    }
}
```

Kompilace vlastního konektoru

Pro kompilaci takového konektoru je výhodné vycházet z předpřipraveného balíčku viz kap. 6.4. To ovšem v tomto případě nestačí. Jelikož bylo použito funkce a knihovny rozšíření o světelný framework v sekci AllJoyn gateway je třeba tyto knihovny zpřístupnit pro proces kompilace a nalinkování. Překlad je řízen pomocí souborů SConstruct a SConscript. Proto tyto soubory je nutno upravit, a to jak přímo ve složce upraveného konektoru, tak v nadřazených složkách. Upravené soubory jsou na přiloženém CD včetně návodu v příloze.

7.3 Ukázkový příklad a jeho běh

V rámci bakalářské práce byl vypracován ukázkový příklad funkčnosti AllJoyn frameworku. Jeho kompletní schéma zapojení je ukázáno na obrázku 5.2. Pro spuštění je nejprve třeba nastartovat směrovač s AllJoyn daemonem, Lighting Servis Controllerem, Gateway managment aplikací a vlastním vytvořeným konektorem. Poté lze do WiFi sítě směrovače připojit mobilní telefon s aplikací Luminaire simulující chytrou žárovku a na jiném mobilním zařízení zapnout aplikaci pro ovládání světla. Pro ovládání simulovaného světla jsou připravené dvě možnosti.

První možností je změnit parametry žárovky v ovládací aplikaci. V takovém případě bude do AllJoyn sběrnice řízené AllJoyn daemonem poslán signál oznamující změnu parametrů. Na této sběrnici naslouchá jak připojená žárovka, která na základě odchycení tohoto signálu změní své parametry, tak konektor, který zprávu zpracuje a pomocí Twittrovského REST API vydá nový tweet ve formátu popsaném v kapitole 7.2.4.

Druhým způsobem je naspání tweetu v definovaném tvaru například pomocí webového rozhraní anebo oficiální mobilní aplikací viz obrázek 7.7. Konektorem na směrovači je každých 15 vteřin kontrolován poslední tweet. Když se jedná o AllJoyn Lighting Comand (AllJoynLC), pošle konektor do AllJoyn sběrnice příkaz pro změnu parametrů příslušného světla. Tím dojde ke změně parametrů dané žárovky viz

obrázek 7.6. Po poslední změně konektor tyto změny na žárovce zaregistruje a vydá na Twitter nový AllJoyn Lighting Status viz obr. 7.7.

Obr. 7.6: Mobilní aplikace Luminaire

Obr. 7.7: Mobilní aplikace Twitter

8 ZÁVĚR

Cílem bakalařské práce byla instalace a konfigurace nově vznikající platformy AllJoyn na směrovači se systémem OpenWRT, zprovoznění vybraného příkladu použití na domácím spotřebiči a ukázkové zprovoznění vzdáleného přístupu z vnější sítě. Framework AllJoyn, který je vytvářen AllSeen Alliancí pod hlavičkou Linux Foundation, se nachází ve stavu intenzivního vývoje a tvorby specifikací. Bakalářská práce se vedle základního frameworku Alljoyn soustředila na nový modul Lighting speciálně určený pro řízení inteligentních světelných zdrojů v domácnosti. V průběhu tvorby práce byla též uvolněna testovací verze modulu Alljoyn Gateway, rozšiřující systém o možnost bezpečného vzdáleného přístupu k zařízením v domácnosti z prostředí Internetu.

Problémy, které bylo nutno vyřešit, byla především kompilace upraveného frameworku Alljoyn a modulu pro řízení světelných zdrojů společně s modulem pro vzdálený přístup pro OpenWRT, dále sepsání a kompilace vlastního konektoru zajišťující propojení vnitřní sítě se službou Twitter. Nejprve bylo potřeba se seznámit a zorientovat v platformě AllJoyn a v systému OpenWRT. Po překladu a instalaci OpenWRT na zařízení TP-Link a jeho konfiguraci bylo nutné přeložit a na tomto zařízení zprovoznit systém AllJoyn s rozšířenými moduly pro správu světel a pro vzdálené připojení. Pro zprovoznění vzdáleného připojení pak bylo dále potřeba naprogramovat speciální konektor a ten přeložit pro OpenWRT. Pro účely této práce byla vybrána služba Twitter a byl vytvořen obousměrný konektor pro tuto službu umožňující jak zobrazovat změny stavu světel formou statusu na Twitteru, tak pomocí vytváření statusu Twitteru ve speciálním formátu měnit stav světel v domácí síti.

V bakalářské práci tak byly dosaženy všechny podstatné stanovené cíle. Případnému pokračovateli bych doporučil vytvoření konektoru postaveném na protokolu XMPP (Extensible Messaging and Presence Protocol), jehož ukázkový kód byl nyní nově vydán. Dále by bylo dobré vytvořit aplikaci pro vzdálené ovládání zařízení z mobilního telefonu prostřednictvím některé z těchto cloudových služeb.

LITERATURA

[1] ALLSEEN ALIANCE. *About AllSeen* [online]. 26.8.2014. 2014 [cit. 3. 11. 2014]. Dostupné z: <https://allseenalliance.org/about>.

[2] ALLSEEN ALIANCE. *AllJoyn System Description* [online]. 26.8.2014. 2014 [cit. 3. 11. 2014]. Dostupné z: <https://allseenalliance.org/sites/default/files/resources/allJoyn_system_description.pdf>.

[3] ALLSEEN ALIANCE. *Introduction to the AllJoyn Framework* [online]. 28.2.2014. 2014 [cit. 4. 11. 2014]. Dostupné z: <https://allseenalliance.org/sites/default/files/resources/intro_to_alljoyn_framework_1.pdf>.

[4] ALLSEEN ALIANCE. *Gateway Service Framework Definition 14.06* [online]. 14.11.2014. 2014 [cit. 28. 2. 2015]. Dostupné z: <https://wiki.allseenalliance.org/_media/compliance/alljoyn_gateway_service_framework_interface_definition_14.06.pdf>.

[5] ALLSEEN ALIANCE. *Gateway Service HLD - Update 2* [online]. 24.11.2014. 2014 [cit. 1. 4. 2015]. Dostupné z: <https://wiki.allseenalliance.org/_media/gateway/alljoyn-gateway-agent-hld_rev1-update2.docx>.

[6] ALLSEEN ALIANCE. *Lighting Service Framework* [online]. 10.11.2014. 2014 [cit. 5. 12. 2014]. Dostupné z: <https://wiki.allseenalliance.org/_media/tsc/lighting/connectedlighting_allseen_summit_nov_2014_v04.pdf>.

[7] ALLSEEN ALIANCE. *Gateway Connector Service API Guide* [online]. 3.2. 2015. 2014 [cit. 8. 5. 2015]. Dostupné z: <https://wiki.allseenalliance.org/_media/gateway/gateway_connector_service_api_guide_linux_2015_0203.pdf>.

[8] ALLSEEN ALIANCE. *Guide to Using the AllJoyn Events and Actions Feature* [online]. 30.1.2014. 2014 [cit. 5. 11. 2014]. Dostupné z: <https://allseenalliance.org/sites/default/files/resources/guide_to_using_alljoyn_events_actions_feature_0.pdf>.

[9] D-Bus - oficiální stránky *Bibliografické citace dokumentů podle ČSN ISO 690 a ČSN ISO 690-2* [online]. 20.1.2014 [cit. 22. 10. 2014]. Dostupné z URL: <http://www.freedesktop.org/wiki/Software/dbus/>.

[10] GOOGLE. *Installing the Eclipse ADT Plugin* [online]. [cit. 6. 12. 2014]. Dostupné z: <http://developer.android.com/sdk/installing/installing-adt.html>.

[11] OPENWRT. *Wirless Freedom* [online]. 30.1.2014. 2014 [cit. 5. 11. 2014]. Dostupné z: <http://wiki.openwrt.org/doc/howto/>.

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

ADT	Android Development Tools
ARM	Acorn RISC Machine
API	Application Programming Interface
BT	British Telecommunications
D-Bus	Desktop Bus
ext4	fourth extended filesystem
HW	hardware
ID	IDentificator number
IoT	Internet of Things
KPN	Koninklijke PTT Nederland
MIPS	Microprocessor without Interlocked Pipeline Stages
NAT	Network address translation
NTFS	New Technology File System
PAN	Personal Area Network
RAM	Random Access Memory
ROM	Read Only Memory
RTOS	Real-Time Operating System
RSA	Rivest, Shamir, Adleman cryptology
SSL	Secure Sockets Layer
TUN	Network TUNnel
TAP	Network Test Access Point
TCP	Transmission Control Protocol
TLS	Transport Layer Security
SDK	Software Development Kit

SW	software
UCI	Unified Configuration Interface
UDP	User Datagram Protocol
UN*X	Unix-like system
VFAT	Virtual File Allocation Table
WiFi	Wireless Fidelity
WLAN	Wireless Local Area Network
XFS	'X' File System
XML	Extensible Markup Language
XMPP	Extensible Messaging and Presence Protocol

SEZNAM PŘÍLOH

A Přílohy **56**

A.1 Rozchození systému z přeložených balíčků 56

A.2 Překlad vlastních obrazů a balíčků 56

A.3 Překlad muzzley konektoru 58

A PŘÍLOHY

A.1 Rozchození systému z přeložených balíčků

Zde je sepsán podrobný návod jak pomocí balíčků přeložených frameworkem AllJoyn ve verzi 14.12 sestavit celý ukázkový systém pro směrovač TP-Link WDR 4300. Všechny potřebný soubory najdeme na přiloženém DVD v adresáři „prelozeno“.

- Připojte se buď pomocí scp nebo přes webové rozhraní LUCI a nahrajte do směrovače obraz (popsané v kapitole 6.5):
  `openwrt-ar71xx-generic-tl-wdr4300-v1-squashfs-sysupgrade.bin`.
- Do složky: /opt/alljoyn/app-manager/ na směrovači nahrajte archív:
  `dummyapp1.tar.gz` a `installPackage.sh`.
- Příkazem „sh installPackage.sh“ v konzoli přihlášené pomocí ssh spusťte instalaci konektoru.
- Vytvořte na svém Twitter účtu vlastní aplikaci. To provedete na stránce:
  `apps.twitter.com`.
  Ve vlastní aplikaci v záložce „Keys and Access Tokens“ vygenerujte „Consumer Key and Secret“.
- Instalace konektoru se provede v adresáři /opt/alljoyn/apps/dummyapp1. Přesunte se tedy do složky konektoru a v podřazené složce bin/ naleznete skripty:
  `postTweet.sh` a `getTweet.sh`.
  V těchto souborech vyplňte vygenerovanými kódy položky:
  `oauth_consumer_key`,`oauth_consumer_secret`,`oauth_token`,`oauth_secret`.
- Ve skriptu `getTweet.sh` navíc definujte proměnou `screen_name` svým Twittrovským uživatelským jménem.
- Do složky: /usr/bin/ na směrovači přehrajeme z DVD:
  `lighting_controller_service`.
- Do složky: /etc/init.d/ na směrovači nahrajte obsah složky init.d z DVD.
- Nutné je vytvořit v pořadníku symbolický link pro spuštění. Přesunte se tedy do složky: /etc/rc.d/ a vytvořte link příkazem:
  `ln -s ../init.d/lighting_controller_service S69lighting_con...`
- Nyní již stačí směrovač restartovat a v případě připojeného směrovače k Internetu bude vše fungovat.

A.2 Překlad vlastních obrazů a balíčků

Pro překlad vlastního OpenWRT obrazu (například pro jiný typ směrovače) a vlastních AllJoyn balíčků a programů (například pro novější verzi frameworku) je zde

přiložen podrobný návod. Aktuální verze pro tvorbu jsou přiloženy na DVD ve složce „k_tvorbe“.

- Do počítače s požadavky probranými v kapitole 6.3.1 nahrajte adresář „barrier_breaker“ z DVD anebo v konzoli stáhněte nejnovější verzi příkazem:
  `svn co svn://svn.openwrt.org/openwrt/branches/barrier_breaker`
- V této složce změňte soubor „feeds.conf.default“ na „feeds.conf“ a v něm doplňte řádek pro tvorbu základního AllJoyn frameworku:
  `git alljoyn`
  `https://git.allseenalliance.org/gerrit/core/openwrt_feed;`
  `barrier_breaker`
- Použite příkazy pro update a instalaci feeds:
  `./scripts/feeds update -a`
  `./scripts/feeds install -a`
- Obsah složky /feeds z DVD nakopírujte do složky:
  `barrier_breaker/package/feeds/alljoyn`
- Do podsložky „dl“ nakopírujte archiv:
  `alljoyn-gwagent-14.12.00-src.tar.gz`
  Archiv je složen z částí popsaných v kapitole 6.4. Obsahuje námi přepsaný ukázkový konektor. Dále pak obsahuje upravené „SConstructy a SConscripty“ pro překlad s nalinkovanými knihovnami rozšířeného světlového frameworku. V případě překladu s novou AllJoyn verzí, je potřeba tento balíček vytvořit nový.
- Pomocí `make menuconfig` vyberte instalaci našich balíčků a základního AllJoyn frameworku.
- Příkazem `make V=s` zpusťte kompilaci s výpisem.
- Po úspěšné kompilaci vám vznikne v podsložce /bin/ar71xx obraz systému. Vyberte náš a nahrajeme viz kapitola 6.5.
- Dále v adresáři:
  `/build_dir/target-mi.../alljoyn-gw.../gateway/gwagent/`
  `build/openwrt/openwrt/release/dist/gatewayConnector`
  najdete složku tar obsahující přeložený konektor s knihovnami a skripty. Složku přejmenujte na „dummyapp1“ za tarujte.
- Instalaci konektoru proveďte stejně jako v předchozím návodě.
- Nově přeložený `lighting_controller_service` najdeme v adresáři:
  `/build_dir/target-mi.../alljoyn-gw.../core/service_framework/`
  `build/linux/standard_core_library/lighting_con..._service/bin`
- Dále už pokračujete stejně jako v předchozím návodu.

A.3 Překlad muzzley konektoru

Návod pro překlad a instalaci konektoru od firmy Muzzley pracující s AllJoyn Lighting.

- Vytvořte složku s názvem „alljoyn-muzzley“
- Do této složky stáhněte do příslušných podadresářů AllJoyn framework. Jednotlivé části stáhneme ze stránek:
  `https://git.allseenalliance.org/gerrit/#/admin/projects/`

```
core/
    gwagent/
    service_framework/
    ajtcl/
    alljoyn/
base_tcl/
base/
services/
    base_tcl/
    base/
```

- v adresáři /alljoyn-muzzley/core/service_framework/ přepište soubor SConscript souborem staženým ze stránek:
  `https://github.com/muzzley/alljoyn-muzzley-connector`
- Do podsložky /standard_core_library stáhneme ze stejných stránek komplet zdrojový kódy alljoyn_muzzley_connector.
- Jinam pak stáhněte zdrojové kódy OpenWRT například pomocí příkazu:
  `git clone git://git.openwrt.org/14.07/openwrt.git`
- Z DVD anebo ze stránek OpenWRT přepište soubor „config.ar71xx_generic“.
- Přepíšte feeds.config.default na feeds.config a doplňte o řádek:
  `src-git alljoyn`
  `https://git.allseenalliance.org/gerrit/core/openwrt_feed;`
  `barrier_breaker`
- Spusťte skripty pro update a instalaci feeds:
  `./scripts/feeds update -a`
  `./scripts/feeds install -a -p alljoyn`
  `./scripts/feeds install libgupnp`
  `./scripts/feeds install libgssdp`
  `./scripts/feeds install -a -p luci`
- Pomocí `make menuconfig` vyberte a mezerníke označte balíčky k nainstalování do obrazu OpenWRT.

```
Networking ---->
    < > alljoyn ---->
        < > alljoyn-about
        < > alljoyn-c
        < > alljoyn-config
            < > alljoyn-config-samples
        < > alljoyn-controlpanel
            < > alljoyn-controlpanel-samples
        < > alljoyn-notification
            < > alljoyn-notification-samples
        < > alljoyn-onboarding
            < > alljoyn-onboarding-samples
        < > alljoyn-sample_apps
        < > alljoyn-samples
        < > alljoyn-service_common
LuCI ---->
    < > Collections ---->
        < > luci
    < > Themes ---->
        < > luci-themes-openwrt
Libraries ---->
    < > libxml2
    < > libgupnp
    < > libgssdp
```

- Příkazem `make` případně `make V=s` nastartujete kompilaci.
- Do vámi vytvořené nové složky alljoyn-muzzley-lib stáhněte pomocí git clonu: `git clone https://github.com/muzzley/muzzley-client-cpp.git` Zde nakopírujte skript `kompilace.sh` pro cross-kompilaci. V něm upravte cesty označené „TODO“ a spusťte.
- Vytvořené knihovny nakopírujte na směrovač do složky /usr/lib. Knihovny a includy také nakopírujte do složky OpenWRT do toolchain_base.
- Do vámi dříve vytvořené složky alljoyn-muzzley zkopírujte soubor `crosskopilace.sh`, znovu upravte cesty a spusťte.
- Přeložený muzzley konektor vznikne ve složce: /core/service_framework/standard_core_library/alljoyn_muzzley_connector. Ten nakopírujte na směrovač do složky /usr/bin a spusťte.