VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INTELIGENTNÍCH SYSTÉMŮ
DEPARTMENT OF INTELLIGENT SYSTEMS

SMART HOME REALIZOVANÝ PROSTŘEDKY ECLIPSE IOT

SMART HOME IMPLEMENTATION BY MEANS OF ECLIPSE IOT

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **KRISTIÁN HEŘMAN**
AUTHOR

VEDOUCÍ PRÁCE **Doc. Ing. VLADIMÍR JANOUŠEK, Ph.D.**
SUPERVISOR

BRNO 2022

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav inteligentních systémů (UITS) Akademický rok 2021/2022

Zadání bakalářské práce

Student: **Heřman Kristián**
Program: Informační technologie
Název: **Smart Home realizovaný prostředky Eclipse IoT**
Smart Home Implementation by Means of Eclipse IoT
Kategorie: Vestavěné systémy

Zadání:

1. Prostudujte problematiku distribuovaných řídicích systémů a IoT. Zaměřte se na aplikace v oblasti Smart Home.
2. Seznamte se s existujícími projekty v rámci Eclipse IoT, určenými jak pro edge, tak cloudová řešení. Pozornost věnujte projektu Eclipse Kura a také možnosti využití systému Apache Kafka.
3. Navrhněte demonstrační řidicí aplikaci pro Smart Home s využitím vybraných prostředků z bodu 2. Pro porovnání navrhněte tutéž aplikaci na bázi Node-RED a Home Assistant.
4. Aplikaci realizujte s využitím vybraných prostředků tak, aby byla snadno konfigurovatelná a instalovatelná z vhodného uložiště. Realizujte i alternativní aplikaci navrženou v bodu 3. Ověřte realizované varianty systému v reálném provozu nebo v simulovaném prostředí.
5. Porovnejte navržené varianty realizace a vyhodnoťte dosažené výsledky.

Literatura:

- Dle pokynů vedoucího.

Pro udělení zápočtu za první semestr je požadováno:

- První 2 body zadání a část návrhu.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Janoušek Vladimír, doc. Ing., Ph.D.**
Vedoucí ústavu: Hanáček Petr, doc. Dr. Ing.
Datum zadání: 1. listopadu 2021
Datum odevzdání: 11. května 2022
Datum schválení: 3. listopadu 2021

Abstrakt

Cílem této práce je návrh řídicí aplikace pro Smart Home s využitím prostředků Eclipse IoT, její sestavení a srovnání s referenční řídicí aplikací. Referenční aplikace je realizována kombinací aplikací Node-RED a Home Assistant. Referenční aplikace však navrhované řešení překoná jak v praktické použitelnosti, tak uživatelské přívětivosti.

Abstract

The aim of this work is to design Smart Home control application by means of Eclipse IoT, its assembly and comparison with the standard control application. Reference application is realized by a combination of Node-Red and Home Assistant applications. However, the reference application will overcome the proposed solution both in practical usability and user-friendliness.

Klíčová slova

chytrá domácnost, řídicí aplikace, domácí automatizace, vestavěné systémy, Eclipse IoT, Node-RED, Home Assistant, Turris Omnia

Keywords

Smart Home, control application, home automation, embedded systems, Eclipse IoT, Node-RED, Home Assistant, Turris Omnia

Citace

HEŘMAN, Kristián. *Smart Home realizovaný prostředky Eclipse IoT*. Brno, 2022. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Doc. Ing. Vladimír Janoušek, Ph.D.

Smart Home realizovaný prostředky Eclipse IoT

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Doc. Ing. Vladimíra Janouška, Ph.D. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Kristián Heřman
11. května 2022

Poděkování

Děkuji mému vedoucímu práce za cenné rady během konzultací. Dále děkuji své rodině za podporu během celého studia.

Obsah

1 Úvod **3**

2 IoT a chytré domácnosti **4**
2.1 DCS, IoT, Smart Home 4
2.1.1 Distribuované řídicí systémy (DCS) 4
2.1.2 M2M – Machine to machine 4
2.1.3 Internet věcí (IoT) 4
2.1.4 Smart Home 5
2.2 Jednotky chytré domácnosti 5
2.2.1 Gateway 5
2.2.2 Cloud 5
2.2.3 Cloud/edge počítání 6
2.3 Existující software realizace 6
2.3.1 Google Assistant / Google Nest 6
2.3.2 Alexa Smart Home 6
2.3.3 Node-RED 6
2.3.4 Home Assistant 7
2.4 Aspekty vlastního řešení 7
2.4.1 Přenos dat 7
2.4.2 Nároky na hardware 8
2.4.3 Užívané normy a komunikační protokoly 8
2.5 Vlastní software realizace 9
2.5.1 Eclipse 4DIAC 9
2.5.2 Eclipse Kura 10
2.5.3 Eclipse Hono 11
2.5.4 Shrnutí 11
2.6 Hardware realizace 11
2.6.1 Senzory, aktuátory, termostaty 11
2.6.2 Centrální řídicí jednotky 12

3 Technologie a komponenty pro implementaci **15**
3.1 Specifikace požadavků na aplikaci 15
3.2 Centrální řídicí jednotka 15
3.2.1 Zprovoznění Turris Omnia 15
3.2.2 Kontejnerová řešení 16
3.2.3 Statické DHCP zápůjčky a mapování na jména 17
3.3 Užívané aktory a senzory 18
3.3.1 Aktory 18

3.3.2 Senzory . . . 19
3.3.3 Cenový přehled komponent . . . 21

4 Řídicí aplikace sestavená z Eclipse IoT 22
4.1 Testované aplikace . . . 22
4.1.1 Eclipse Kura . . . 22
4.1.2 Eclipse Hono a Apache Kafka . . . 23
4.1.3 Eclipse Kapua . . . 23
4.1.4 Eclipse Streamsheets . . . 23
4.2 Architektura výsledného řešení . . . 25

5 Řídicí aplikace sestavená z Node-RED a Home Assistant 26
5.1 Výběr aplikací a konfigurace . . . 26
5.2 Instalace Node-RED . . . 26
5.3 Konfigurace komponent . . . 27
5.3.1 Žárovka Yeelight . . . 28
5.3.2 WiFi relé Shelly . . . 29
5.3.3 2 kanálové USB relé . . . 29
5.4 Tvorba nástěnek . . . 30
5.5 Instalace Home Assistant . . . 30
5.6 Konfigurace komponent . . . 30
5.6.1 Žárovka Yeelight . . . 31
5.6.2 WiFi relé Shelly . . . 31
5.6.3 2 kanálové USB relé . . . 31
5.6.4 Digitální teploměr a PIR čidlo . . . 31
5.6.5 Bezpečnostní kamera . . . 31
5.7 Tvorba nástěnek . . . 32
5.8 Propojení Node-RED a Home Assistant . . . 32
5.8.1 Vytváření automatizací . . . 32

6 Srovnání 34
6.1 Vzájemné srovnání aplikací . . . 34

7 Závěr 35

Literatura 36

Kapitola 1

Úvod

Existující řešení pro tvorbu chytrých domácností jsou velmi drahá a nebo příliš náročná pro uživatele. Náročnost u finančně výhodnějších variant je pak zejména časová – uživatel musí sám hlídat kompatibilitu jednotlivých zařízení, jejich uživatelská rozhraní pro konfiguraci jsou navíc často odlišná, tedy pro běžného zájemce o instalaci chytré domácnosti nevyužitelná. Tato práce by měla všem těmto uživatelům poskytnout bezplatně jednoduše instalovatelný a konfigurovatelný balík, s pomocí kterého každý zvládne zprovoznit chytrou domácnost do několika dní. Implementace celého řešení Smart Home by zabrala velké množství času, proto jsou v této práci využity knihovny ze skupiny *Eclipse IoT*, které jsou vyvíjené a distribuované jako otevřený software.

K výběru tématu autora vedlo více důvodů, mezi zásadní patří již zmíněná cena a dostupnost řešení chytrých domácností pro zájemce, kteří chtějí jednoduché a levné řešení. Dalším neopomenutelným důvodem pro výběr tohoto tématu je také uplatnitelnost výstupů této práce v reálném rodinném domě, v kterém je plánováno chytrou domácnost využívat.

Cílem práce je prozkoumat možnosti knihoven Eclipse IoT a z vybraných projektů následně sestavit demonstrační řídicí aplikaci pro chytrou domácnost. Pro porovnání bude navrhnuta tatáž aplikace na bázi projektů Node-RED a Home Assistant.

Součástí práce bude i kompletace výsledné aplikace spolu s postupem instalace a konfigurace. Aplikace je sestavena z knihoven projektu Eclipse IoT.

Tato práce je rozdělena do několika kapitol. Následující kapitola 2 se zabývá obecným pohledem na řídicí systémy, dále Internetem věcí a Smart Home, včetně existujících řešení a souvisejících podrobností. V kapitole 3 jsou diskutovány technologie a komponenty využívané pro tvorbu chytrých domácností. Kapitola 4 popisuje postup práce na aplikaci s využitím projektů Eclipse IoT, kapitola 5 se poté věnuje referenčnímu řešení s využitím softwaru Node-RED a Home Assistant. V kapitole 6 jsou pak obě řešení porovnána a vyhodnocena.

Kapitola 2

IoT a chytré domácnosti

Chytrá domácnost jako ekosystém obsahující nespočet technologických prvků od termostatů, osvětlení, elektroinstalace až po bezpečnostní kamery nebo zámky dveří vychází z konceptu řídicích systémů.

2.1 DCS, IoT, Smart Home

2.1.1 Distribuované řídicí systémy (DCS)

Jedna z možností implementace tohoto je i typ distribuovaného řídicího systému (*DCS – distributed control system*), kde každá součástka je zároveň i řídicí jednotkou s pokročilejší logikou. Tyto jednotky si mezi s sebou dle zadaného komunikačního standardu vyměňují informace a žádná z nich tak není jedním centrálním prvkem. Koncept však lze rozvinout i na systém s centrální řídicí jednotkou, která všechny tyto informace shromažďuje, pak mluvíme o centrálním řídicím systému.

Mezi nesporné výhody distribuovaných systému patří jejich vyšší výkonnost, možnost výstavby složitějších systémů a snadná rozšiřitelnost. Proto jsou na tomto konceptu projektovány i chytré domácnosti, které principiálně vycházejí z řídicích systémů pro velké průmyslové firmy.

2.1.2 M2M – Machine to machine

M2M, neboli *Machine to machine* je princip přímé komunikace mezi dvěma zařízeními, typicky mezi čidly, ale také mezi jinými prvky řídicích systémů. Zásadní myšlenkou je princip, při které si tato zařízení zadaným standardem předávají data přímo mezi sebou, a to automatizovaně, bez pomoci člověka. Tuto technologii můžeme také považovat za předchůdce, ale také nezbytnou součást Internetu věcí (*IoT* – Internet of Things).

2.1.3 Internet věcí (IoT)

Se zvyšující se dostupností chytrých zařízení, která lze připojit do internetové sítě začala vzrůstat také poptávka po jejich vzájemném propojení a komunikaci. I proto, že technologie M2M tomuto požadavku nedostačovala, vznikla technologie IoT. Více zařízení lze připojit do jednoho centrálního místa, které je nazýváno *cloud*, tyto pak mohou zasílat data z reálného času a současně si je také přes tuto jednotku předávat. Narozdíl od *Machine to machine* principu je *Internet of Things* lépe škálovatelný, s tím souvisí také zvýšení výkonnosti celého systému. IoT je tak možné využívat u malých projektů jako jsou chytré domácnosti,

ale také u velkých senzorických sítí ve výrobních provozech. Na základě těchto technologií a již zmíněné dostupnosti chytrých prvků vznikl nový obor zabývající se projektováním chytrých domácností či komplexních řídicích systémů v továrnách, návrhem a výrobou zařízení, software a podobně.

2.1.4 Smart Home

Pojem Smart Home vznikl již v 60. letech 20. století [32], když začaly vznikat první počítače, určené k základní automatizaci, nebo získávání dat z čidel přítomných v domácnosti a odkazuje právě k něčemu, co bychom mohli nazvat automatizace v domácnosti (*home automation*). Jako příklad užití automatizací v domácnosti můžeme zmínit regulaci jasu světel podle denní doby, reakce termostatů nejen na změny teploty, ale současně na přítomnost osob v domě na základě dat z pohybových čidel. Autoři článku *Major requirements for building Smart Homes in Smart Cities based on Internet of Things technologies*[33] z roku 2016 uvádí 7 nejdůležitějších požadavků na chytrou domácnost:

1. Různorodost zařízení (heterogenita)
2. Autokonfigurace
3. Rozšiřitelnost
4. Kontextovost
5. Použitelnost
6. Bezpečnost a soukromí
7. Inteligence

2.2 Jednotky chytré domácnosti

2.2.1 Gateway

Gateway je prvek, který je v chytré domácnosti přímo spojen s jednotlivými senzory, aktuátory a termostaty. Nachází se přímo v objektu chytré domácnosti právě proto, že je fyzicky propojen s nižšími prvky. Shromažďuje všechna data z čidel a odesílá je do cloudové části, v které jsou typicky zpracována (u edge řešení 2.2.3 je lze zpracovat již na gateway). Úkolem gateway je ale také všechny prvky ovládat na základě příkazů, přicházejících z cloudu. Gateway tak můžeme nazývat jako vstupně-výstupní bránu chytré domácnosti a považovat ji za její nezbytnou součást.

2.2.2 Cloud

Cloud (mohli bychom nazývat jako vzdálený internetový server), je v základní struktuře chytré domácnosti umístěn v serverové části. To znamená, že se fyzicky nemusí nacházet v řízeném objektu a je tak často realizován jako virtuální privátní server, který lze zakoupit u běžných internetových poskytovatelů. Do cloudu jsou zasílány data z gateway a to buď v původní podobě jak je brána přijala, nebo již v nějakém zpracovaném stavu.

2.2.3 Cloud/edge počítání

Cloud a edge počítání (neboli *cloud computing* a *edge computing*) jsou dva přístupy k zpracování dat v závislosti na vzdálenosti od cloudu. *Cloud computing* zpracovává informace až v cloudu. Je třeba ale vyhodnotit, která data můžeme zpracovat až v cloudu a která je třeba zpracovat již dříve po cestě, protože jsou zásadní pro další běh a není čas je odesílat k zpracování do cloudu. Druhým důvodem, proč se využívá *edge computing* je také předzpracování velkých objemů dat příchozích na gateway tak, abychom těmito daty nezatěžovali server, ale poslali mu data až po dokončeném předzpracování na vstupní bráně.

2.3 Existující software realizace

Jak bylo zmíněno – v posledních letech vzniká mnoho řešení chytrých domácností. Některá z nich budou detailněji představena a srovnána níže. Řešení umožňují tvorbu ekosystémů jednoho výrobce – s tím ale souvisí problémy rozšiřitelnosti. Uživatel je nucen si po výběru jednoho výrobce kupovat všechny další prvky od toho stejného. To je nevýhodné jak finančně (tyto prvky ze značkových ekosystémů jsou velmi drahé), tak limitují uživatele dostupností kompatibilních zařízení. Výhodou je naopak jednoduchost sestavení domácnosti. Pro cílového uživatele chytré domácnosti, která bude navržena a implementována v této bakalářské práci jsou ale tato řešení nevyhovující.

2.3.1 Google Assistant / Google Nest

Google Assistant je jeden z hlasových asistentů mezi dalšími jako jsou Alexa od společnosti Amazazon, Siri od společnosti Apple či Cortana od společnosti Microsoft. Od roku 2016, kdy byl Google Assistant uveden na trh se rozšířil mezi obrovskou masu lidí (v roce 2020 jej používalo již přes 500 milionů uživatelů [21]). Současně je také považován za jeden z nejpokročilejších hlasových asistentů. Google Assistant spadá pod chytrý reproduktor Google Nest (dříve Google Home). Oběma technologiemi se dá ovládat přes 10 000 různých zařízení.

2.3.2 Alexa Smart Home

Alexa je chytrý reproduktor vyvíjený společností Amazon, který má stejné cíle. Hlasovým ovládáním řídit jakékoli zařízení v okolí. I proto je velmi těžké tyto dva produkty objektivně srovnávat. Faktem ale zůstává, že Google Assistant od společnosti Google disponuje obrovskou výhodou, kterou jsou bezesporu data z mobilních telefonů s operačním systémem Android, do kterých Google dodává spoustu svých aplikací. Mnohem jednodušeji tak získává a zpracovává data o pohybu či například denním programu uživatele. Alexa však nabízí takzvané uživatelské rutiny – tedy že ráno rozsvítí světla, zapne hudbu a přečte denní harmonogram.

2.3.3 Node-RED

Projekt Node-RED je vývojový nástroj pro propojování hardwarových součástek, jejich aplikačních rozhraní a online služeb pomocí intuitivní webové aplikace v internetovém prohlížeči. [22] Projekt poskytuje širokou nabídku součástek (uzlů), které představují jednotlivé akce v toku dat. Jako příklady takových elementárních součástek si lze představit například stisk tlačítka nebo spuštění zvuku. [4] Uživatel postupně propojuje jednotlivé uzly

a tím skládá svůj vlastní program. Projekt lze zprovoznit jak na základních jednodeskových počítačích Raspberry Pi, tak v cloudovém prostoru.

2.3.4 Home Assistant

Open-source projekt Home Assistant je software zaměření na sestavení a následné řízení chytrých domácích zařízení. Je navržen jako centrální jednotka pro řídicí systém. Z tohoto centrálního bodu (webové či mobilní aplikace) umožňuje nastavit komunikaci mezi jednotlivými zařízeními, vytvořit si z dlaždic přehlednou nástěnku a na ni si umístit pouze pro uživatele důležitá zařízení.

Je možné také nastavovat automatizace na základě stavu zařízení nebo velikosti jejich veličin. [20] V některých věcech se Home Assistant svou funkčností překrývá s projektem Node-RED, jinak je ale každá aplikace zaměřena trochu jiným stylem. Home Assistant vítězí v míře rozšiřitelnosti – pro mnoho zařízení není třeba instalovat nové balíčky (jako by to bylo u uzlů Node-RED). Ten naopak zase vyhrává v možnosti psát komplexnější a rozsáhlejší podmínky automatizace díky své podpoře logických operátorů nebo if/else.

Dobrých výsledků lze tak docílit kombinací těchto dvou řešení – Home Assistant využít jako centrální řídicí jednotku, do které vedou všechny chytré prvky, Node-RED zase obstará logiku všech automatizovaných operací. [23]

2.4 Aspekty vlastního řešení

Nejen u vlastního řešení chytré domácnosti je třeba dopředu vyřešit několik aspektů, které by v pozdější fázi instalace mohly přinést velké problémy. V podstatě jde o kritéria či základní pilíře, které je nutné stanovit pro pohodlnou následující práci. U existujících realizací chytrých domácností nám toto stanovuje specifikace daného řešení, u vlastního musíme toto specifikovat sami. Jednou z problematik je typ přenosu dat mezi jednotlivými zařízeními.

2.4.1 Přenos dat

KNX Špičkovým způsobem přenosu dat mezi řídicími systémy je sběrnice KNX. Tato specializovaná norma, která se úzce zaměřuje na automatizaci budov, je pevně zapojená kabeláž po celém objektu, v kterém se budou nacházet jednotlivá čidla, kde každé čidlo je na tuto sběrnici napojeno. Toto řešení je poměrně drahé a také se příliš nehodí, pokud má být chytrá domácnost instalována do hotového objektu. Ve většině případů je tak přenos zajištěn bezdrátovým spojením. Zde budou okomentovány alespoň technologie Z-Wave, Zigbee a Wi-Fi.

Z-Wave Z-Wave je síťová technologie s nízkou spotřebou vyvinuta na začátku 21. století v Dánsku pro účely ovládání domácích spotřebičů na dálku. Technologie pracuje v přenosovém pásmu 800–950 MHz, kde však stěny, podlahy a podobně budou znatelně snižovat dosah. [11] V této technologii je vyráběno přes několik tisíc zařízení. Obecně lze Z-Wave pohodlněji využít v Americe či v Kanadě, kde je sortiment prodávaných produktů podporujících tuto technologii širší, než v Evropě.

Zigbee Bezdrátová komunikační technologie Zigbee vychází ze standardu IEEE 802.15.4 (IEEE 802 je rodina IEEE standardů pro sítě LAN a MAN), pracuje na bezlicenčních pásmech 868 MHz, 902–928 MHz a 2,4 GHz. Rovněž jako technologie Bluetooth (IEEE

802.15.1) je Zigbee určeno pro bezdrátový přenos dat do vzdálenosti maximálně 75 metrů. Tato technologie cílila v první řadě na využití v průmyslu a senzorových sítích. [13] Jelikož bývají zařízení s technologií Zigbee levnější alternativou k Z-Wave, staly se oblíbenými i u uživatelů – a to i přes nutnost mít speciální rozbočovač nebo software podporující Zigbee. Zajímavostí je, že Amazon's Echo Plus (tedy hlasový asistent Alexa) má jako jediný chytrý reproduktor zabudované Zigbee rádio. [11]

WiFi WiFi je bezdrátový komunikační standard pracující na frekvenci 2,4 GHz/5 GHz. Dosah technologie WiFi je rozhodně nižší než u předchozích bezdrátových technologií – i proto se doporučuje při použití WiFi v chytré domácnosti investovat do kvalitního zařízení či do opakovačů signálu. Velký počet zařízení připojitelných zároveň do WiFi sítě a jejich malá cena v porovnání s předchozími technologiemi je ale vykoupena vyšší spotřebou elektrické energie [17]

Bluetooth Bluetooth technologii zde musíme zmínit, protože je velmi rozšířenou metodou přenosu ve světě bezdrátových sítí. Kvůli své malé šířce pásma a krátkému dosahu je pro projekty chytrých domácností hůře použitelná. Jeho inovovaná varianta se sníženou spotřebou energie zvaná BLE (*Bluetooth Low Energy*) ale pro některá čidla s malým datovým průtokem může stačit, pokud budeme uvažovat přenos na kratší vzdálenost. Data z čidla mohou být pomocí Bluetooth Low Energy odeslána na jednotku, která tato data přijme také technologií Bluetooth, ale následně je do brány odešle přes WiFi nebo jiným spojením.

2.4.2 Nároky na hardware

Nároky na hardware velmi úzce souvisejí s počtem připojených zařízení a množstvím komunikovaných dat. Pro běžnou chytrou domácnost obsahující celkově maximálně desítky čidel a aktuátorů ale na gateway, tedy centrální bod pro připojení všech zařízení, stačí jednodeskový počítač Raspberry Pi s operační pamětí 4 GB a WiFi/Ethernet adaptérem. Pro cloudovou část, kde budou data zpracována a vizualizována by pak mělo být dostačující opět Raspberry Pi, případně lze využívat virtuální privátní server (VPS), který lze zakoupit u mnoha poskytovatelů serverhostingu. Virtuální server také řeší otázku přístupu z vnější sítě. VPS je totiž běžně dodáván jako fyzický stroj s vlastními vyhrazenými prostředky a statickou IP adresou, přes kterou lze řídicí aplikaci zpřístupnit do sítě Internet.

2.4.3 Užívané normy a komunikační protokoly

IEC normy Mluvíme-li o distribuovaných řídicích systémech, nejpoužívanější jsou dle článku *Usage of IEC Standards for Distributed Control Systems Modelling* [30] normy IEC 61131 a IEC 61499. Standard IEC 61499 poskytuje možnosti pro snadné modelování distribuovaných řídicích systémů, je celkově novějším standardem a i proto je méně využíván.

TCP/IP Sada protokolů TCP/IP zahrnuje celou hierarchii internetového přenosu od přenosového média až po jednotlivé protokoly aplikační vrstvy. Stěžejní typy přenosů L1 vrstvy byly probrány výše. Některá čidla fungují zjednodušeně na L3 vrstvě výměnou TCP/UDP paketů, většina ale běží nad L4 aplikační vrstvou a jejími protokoly, z nichž si detailně popíšeme alespoň ty nejpoužívanější a uvedeme konkrétní příklady užití při konstrukci chytrých domácností.

HTTP HTTP (*Hypertext Transfer Protocol*) je jednoduchý bezstavový protokol pro přenos libovolných dat mezi klientem a serverem. Vznikal současně se vznikem systému WWW (*World Wide Web*) pro potřeby přenosu webových stránek ke klientovi. V začátcích dokázal zpracovat požadavek a odeslat požadovaný dokument, což je dnešní metoda GET. Později ale přibyly před obsah požadavku i odpovědi takzvané hlavičky, které specifikují podrobné informace (stavový kód, velikost souboru apod.), protokol byl také rozšířen o další metody (například POST), která umožňuje kromě požadavku posílat na server také data z formulářů. Nad metodami protokolu HTTP vznikla jednoduchá architektura REST API, která umožňuje za pomocí těchto metod snadno nakládat s daty na serveru. Na této architektuře dnes pracuje mnoho chytrých bezdrátových komponent, a to z důvodu snadné implementace i obsluhy uživatelem.

MQTT MQTT je pro svoji jednoduchost a nenáročnost jeden z velmi používaných protokolů v oblasti komunikace chytrých zařízení. Protokol funguje na principu publikování zpráv na server do takzvaného tématu, což je jednoznačný identifikátor v hierarchické struktuře všech témat. Každé zařízení, které chce toto téma číst se pak takzvaně přihlásí k odběru zpráv z daného tématu. Toto komunikační schéma se dle jmen operací nazývá *publish–subscribe* a jeho výhodou je možné použití u nevýkonných senzorových sítí, které si mezi sebou prostřednictvím serveru (takzvaný *broker*) mohou vyměňovat velká množství zpráv, ale každé čidlo vůbec nemusí sledovat komunikaci ve všech tématech. Protokol specifikuje strukturu dat, ne však jejich obsah, což znamená, že jejich interpretace pak záleží na konečném příjemci – nejčastěji se však používají klasické textové zprávy či strukturovaný formát typu JSON.

AMQP Funkce a principy protokolu AMQP se částečně překrývají s protokolem MQTT, nadto je ale dále rozšiřují. AMQP je otevřeným protokolem, standardizován normou ISO/IEC 19464, který byl vyvinut v odvětví bankovnictví. I proto je zaměřen na přenos větších objemů dat, s kterými umožňuje manipulovat více způsoby – stejně jako MQTT lze využívat mechanismus publish-subscribe, dále však AMQP nabízí mimo jiné fronty zpráv či algoritmus *round robin*. Kvůli větším přenosům dat má větší nároky na šířku pásma a v porovnání s jednodušším MQTT má také vyšší přenosovou dobu. [31]

2.5 Vlastní software realizace

Eclipse IoT je otevřená komunita, která vyvíjí a poskytuje open-source stavební bloky pro odvětví IoT a M2M (tzv. Machine to Machine). Hlavní zaměření těchto bloků je implementace standardů a vývoj frameworků použitelných pro tvorbu IoT řešení. Cílem Eclipse IoT je touto prací podporovat rozvoj otevřeného internetu věcí se schopností vzájemné spolupráce jeho prvků. První projekt pod hlavičkou Eclipse IoT vznikl v roce 2012, v současné době již existuje 12 různých projektů zaměřených na IoT a M2M.

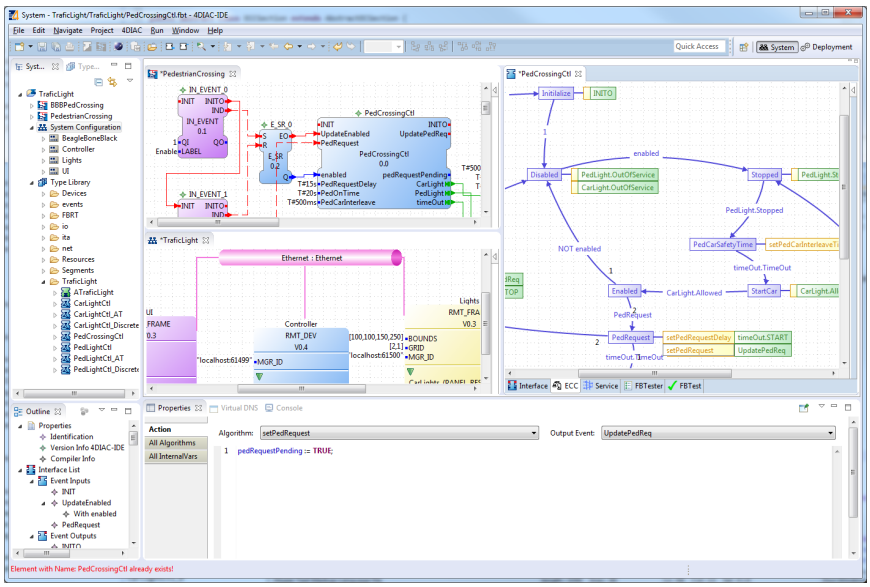
2.5.1 Eclipse 4DIAC

Eclipse 4DIAC je PLC framework vyvinutý pro automatizaci a řízení v průmyslu. Poskytuje možnosti pro modelování distribuovaných řídicích systémů v průmyslu dle normy IEC 61499[1]. Ta definuje obecný model pro distribuované řídicí systémy.

[1] https://iec61499.com/

Eclipse 4DIAC poskytuje grafické uživatelské rozhraní pro návrh a modelování systémů. Práce s editorem funguje na principu propojování funkčních bloků, z kterých následně vzniká celá síť funkčních bloků. Funkční bloky si může uživatel dopředu definovat, testovat jejich funkčnost zadáním testovacích hodnot nebo spuštěním sady testovacích scénářů. Dokončené řešení pak může být aplikováno na zařízení dle standardu IEC 61499.

Kromě editoru je součástí projektu i Eclipse 4DIAC FORTE, což je implementace běhového prostředí právě pro funkční bloky vytvořené v souladu s normou zmíněnou dříve, ale také další subaplikace. 4DIAC FORTE umožňuje komunikaci pomocí internetové technologie Ethernet, podporuje také protokol MQTT, používaný v internetu věcí.

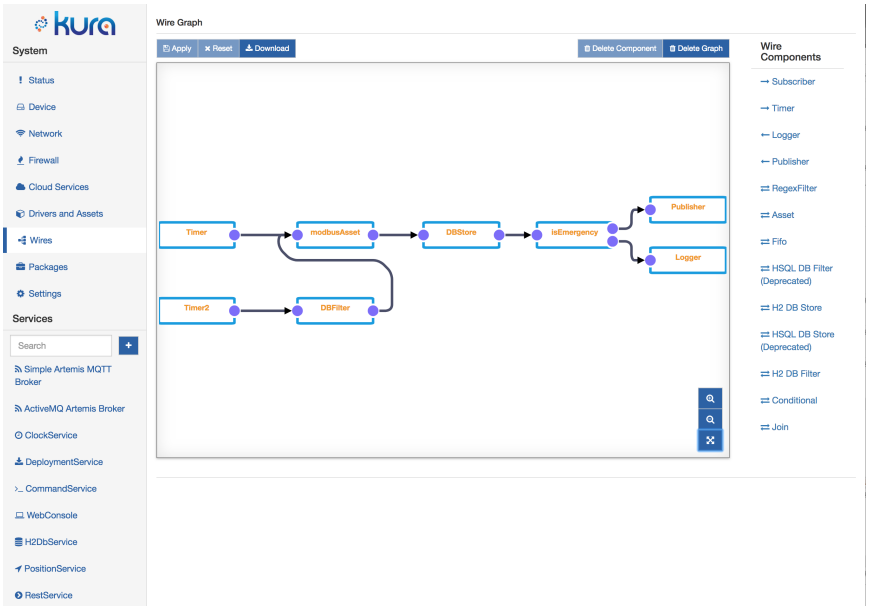


Obrázek 2.1: IDE projektu Eclipse 4DIAC, převzato z [19].

2.5.2 Eclipse Kura

Eclipse Kura je rozšiřitelný framework napsaný v jazyce Java, který je vhodný pro vývoj edge IoT gateway. Snaží se shromažďovat a propojovat otevřené implementace pro zásadní služby M2M/IoT. Eclipse Kura běží na Java Virtual Machine, kde využívá dynamický modulární systém OSGi umožňující manipulaci s moduly za běhu, což zjednodušuje proces uživatelského vytváření znovupoužitelných bloků. Pro svoji oblíbenost je Eclipse Kura připraven pro nasazení na desku Raspberry Pi.

Mezi zásadní součásti Eclipse Kura patří API potřebné pro komunikaci s hardwarovým rozhraním, webové grafické rozhraní pro práci s toky dat pomocí propojování jednotlivých bloků (časovač, logování, publikování na server atp.) a API pro komunikaci se vzdáleným serverem pomocí MQTT protokolu. Komunikační hardwarové API lze využít pro komunikaci se sériovými porty, USB, GPS nebo k portům připojeným jako GPIO/PWM/I2C/SPI.



Obrázek 2.2: IDE Eclipse Kura umožňující propojování bloků, převzato z [18].

2.5.3 Eclipse Hono

Eclipse Hono je projekt zajišťující IoT konektivitu do cloudu. V IoT se vyskytuje více standardů komunikace, což při nutnosti implementace každého způsobu zvlášť snižuje možnosti škálovatelnosti celého řešení. Proto Eclipse Hono implementuje rozhraní pro tyto standardy (komunikační protokoly HTTP, MQTT, CoAP a AMQP 1.0) tak, aby byla tato komunikace zjednodušena na jeden sjednocený formát. Zprávy jsou rozděleny Eclipse Hono má také připraveno specifické API pro připojení message brokeru Apache Kafka.

2.5.4 Shrnutí

Eclipse 4DIAC je vhodnou aplikace spíše pro průmyslové řízení a distribuované řídicí systémy, proto se pro řešení chytré domácnosti příliš nehodí. Je ale zajímavou aplikací pro porovnání s jinými aplikacemi, například lze vidět, že užívá princip modelování a spojování funkčních bloků, což odráží potřebu práce s toky dat dle jejich faktorů, která je u řídicích systémů nutností. Na rozdíl od toho, Eclipse Kura může být velmi přínosný pro práci na gateway a komunikaci s čidly na nejnižší úrovni. Eclipse Hono může dost pomoci na cloudové straně architektury, kde je potřeba do jednoho místa vést několik různých toků dat různých formátů.

2.6 Hardware realizace

2.6.1 Senzory, aktuátory, termostaty

Teploměr Teploměr je přístroj k měření okolní teploty. Ta se určuje pomocí metod, založených na mnohých principech, například elektrického odporu či teplotní roztažnosti. [12]

Termostat Termostat je zařízení, které za účelem udržování stálé požadované teploty spíná a rozpíná obvod tepelného vytápění. Pro správnou funkci je uvnitř termostatu teploměr, podle kterého je vytápění regulováno.

Spínací relé Relé funguje jako spínač, nejčastěji ovládaný elektromagneticky, či elektronicky (v takovém případě mluvíme o takzvaném *Solid state relé*). Ve chvíli připojení proudu na spínací kontakt relé sepne hlavní řízený obvod.

Bezpečnostní kamera Kamera funguje jako fotoaparát, který nepřetržitě snímá mnoho obrazů za sebe a tím vytváří pocit pohyblivého záznamu. Součástí toho může být i záznam zvukové stopy, obojí je následně ukládáno na úložiště nebo odesíláno jako živý přenos přes podporované médium.

Alarm Alarm je určen k upozornění či spuštění poplachu v případě, že jsou porušeny předem dané podmínky, například přes pohybové čidlo projde osoba a do 30 sekund nezadá správný bezpečností kód na klávesnici. Zařízení může o narušení prostoru informovat zvukovým či světelným signálem, zprávou majiteli nebo přímo bezpečnostní službě.

Pohybové čidlo Pohybové neboli PIR čidlo má za úkol detekovat pohyb v střeženém prostoru. Princip je založen na rozpoznání infračerveného záření, které některé objekty včetně člověka vyzařují ve větším množství. [24]

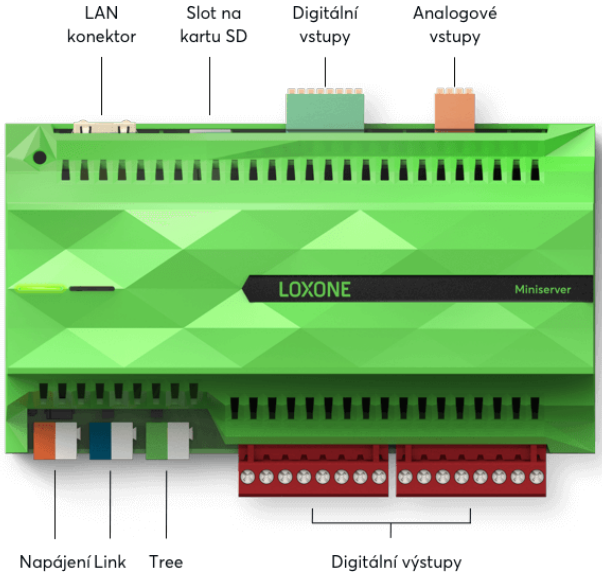
Požární čidlo Rozlišujeme dva typy požárních čidel – kouřové a teplotní. První z nich funguje s výhodou optického čidla instalovaného uvnitř, v případě vzniku kouře toto senzor rozpozná a spustí alarm. Druhý typ hlídá teplotu v prostoru instalace a poplach vyhlásí při jejím překročení nad stanovenou hodnotu. [25]

2.6.2 Centrální řídicí jednotky

Společnosti, zabývající se realizací chytrých domů, bytů, průmyslových budov a dalších automatizací si velmi často vyvíjejí svá vlastní kompletní řešení včetně vlastních komponent i hlavní řídicí jednotky. Nevýhodou je neschopnost zákazníka kombinovat součástky od více výrobců – tedy je naprosto závislý na výrobcích od vybraného řešení, což může vést k tomu, že zákazník bude chtít využívat komponentu, kterou společnost nenabízí, nebo si za kteroukoli součást může určovat libovolnou cenu. Na druhou stranu, výhodou užívání těchto „řešení na míru“ od specializovaných společností je spolehlivá kompatibilita, kterou si tyto firmy pečlivě hlídají. Řídicí jednotky si společnosti samy navrhují podle svých požadavků vlastních a případně zákazníků. Některé společnosti také vyvinuly základní jednotku a tu umožňují rozšiřovat o další moduly, které si zákazník může dokoupit.

Loxone Firma Loxone s.r.o. pochází z Rakouska a od roku 2009 podniká na poli inteligentních domů včetně vlastního hardware. V současné době poskytují dva typy řídicích center – jednotku *Miniserver*, které je určeno především do novostaveb a které poskytuje kromě klasických analogových vstupů, digitálních vstupů/výstupů a LAN portu hlavně 2 typy rozhraní. *Loxone Link*, určený pro připojení až dalších 30 rozšiřujících jednotek Miniserveru a *Loxone Tree*, které je připraveno pro až 50 čidel či aktuátorů. Tato jednotka bez dalších rozšíření (mezi které patří i modul *Loxone Air*, potřebný pro připojení bezdrátových komponent) stojí kolem 15 tisíc korun bez DPH. [2] Pro rekonstrukce a jiné případy,

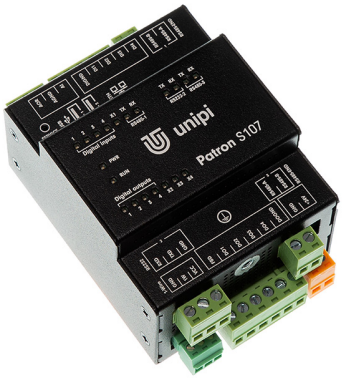
kde je jednotka *Miniserver* nevhodná, poskytuje Loxone také druhý typ *Miniserver Go*, který místo rozhraní *Loxone Tree* obsahuje *Loxone Air* pro připojení až 128 bezdrátových inteligentních zařízení. I tato jednotka samozřejmě umožňuje rozšiřování pomocí připojení dalších modulů přes rozhraní *Loxone Link*.



Obrázek 2.3: Centrální jednotka *Loxone Miniserver*, převzato z [3].

Unipi technology Česká společnost sídlící v Brně, *Unipi technology*, vyvíjí, vyrábí a dodává na tuzemský i zahraniční trh řídicí jednotky, senzory a další inteligentní zařízení včetně software. Firma vznikla pro potřeby vývoje systému, který by řídil budovu datacentra mateřské firmy s cílem uspořit co nejvíce energie. Velmi brzy po nástupu Raspberry Pi v roce 2013 se firma rozhodla pro využití těchto počítačů a v roce 2014 vydala první řídicí jednotku s názvem *Unipi 1.1*. Dnes jsou nejoblíbenějšími 2 řady jednotek – *Unipi Neuron*, který běží na Raspberry Pi. [27] Nejprodávanější model *Unipi Neuron S103* nabízí digitální i analogové vstupy/výstupy, sběrnici 1-Wire a sériové rozhraní RS485 připravené pro komunikaci protokolem Modbus, to vše za cenu kolem 6 tisíc korun bez DPH [28] Novější produktová řada *Unipi Patron* přechází od Raspberry Pi k vlastním počítačům osazeným procesorem od společnosti NXP. Vlajkový model *S107* je vyráběno se stejnými rozhraními jako model *S103* rozšířené o nově přidané sériové rozhraní RS232. Tato jednotka se cenově pohybuje kolem 9 tisíc korun bez DPH.

Jiná hardware řešení Samozřejmostí je možnost využít jiná hardware řešení, čímž není myšlen pouze vývoj vlastní počítačové desky s všemi požadovanými rozhraními, kde je proces od návrhu až po realizaci velmi náročný. V dnešní době existuje velký výběrem hardware produktů, které nám k realizaci chytré domácnosti mohou stačit. Kromě velmi oblíbených jednodeskových počítačů Raspberry Pi dnes lze využít například i výkonnější směrovač Turris Omnia. Český produkt společnosti CZ.NIC tak se svým 1,6 GHz ARM procesorem, 2 GB RAM a operačním systémemem OpenWRT dokáže kromě zprostředkování internetové konektivity, provozu NAS či tiskového serveru provozovat i centrální řídicí jednotku pro chytrou domácnost. [9] Mimo klasické rozhraní LAN a WiFi jsou k dispozici také GPIO



Obrázek 2.4: Centrální jednotka *Unipi Patron*, převzato z [29].

vývody a 2 mini PCIe sloty pro připojení dalších modulů. Cena tohoto směrovače je kolem 6 tisíc korun bez DPH.

Obrázek 2.5: Výkonný směrovač společnosti CZ.NIC *Turris Omnia, model 2020*, převzato z [8].

Kapitola 3

Technologie a komponenty pro implementaci

3.1 Specifikace požadavků na aplikaci

Toto konkrétní řešení řídicí aplikace chytré domácnosti počítá s tím, že bude umět komunikovat se senzory jako jsou teploměr, vlhkoměr, pohybové čidlo, kamera a také s aktuátory, které budou řídit garážové dveře, termostaty či osvětlení domu. Současně je vhodné, aby řešení umožňovalo i práci na logické vrstvě systému. Tedy například nastavení automatického rozsvícení osvětlení v domu podle intenzity svitu v místnosti apod.

3.2 Centrální řídicí jednotka

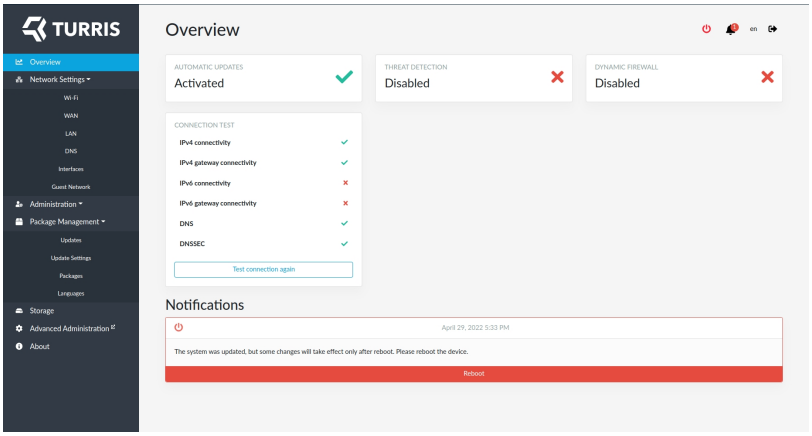
Pro praktickou realizaci centrální řídicí jednotky byl vybrán router Turris Omnia, který byl v kapitole 2.6.2 uveden jako možná alternativa k počítačům Raspberry Pi. Směrovač běží na linuxové distribuci OpenWRT, která je vyvíjena jako operační systém pro síťové prvky a další jednoduchá embedded zařízení. I přes tuto specializaci ale nabízí více než 3000 balíčků [6] dostupné přes balíčkovací systém `opkg`.

3.2.1 Zprovoznění Turris Omnia

Po připojení směrovače Turris Omnia 2020 do elektrické sítě jej musíme správně nakonfigurovat pro další práci. K administraci routeru máme několik možností a je velmi vhodné tohoto využívat. Standardní cestou ovládání je přístup pomocí protokolu SSH. Pohodlnější způsob nastavení ale nabízí spíše webové uživatelské rozhraní, tedy dokonce dvě různé aplikace.

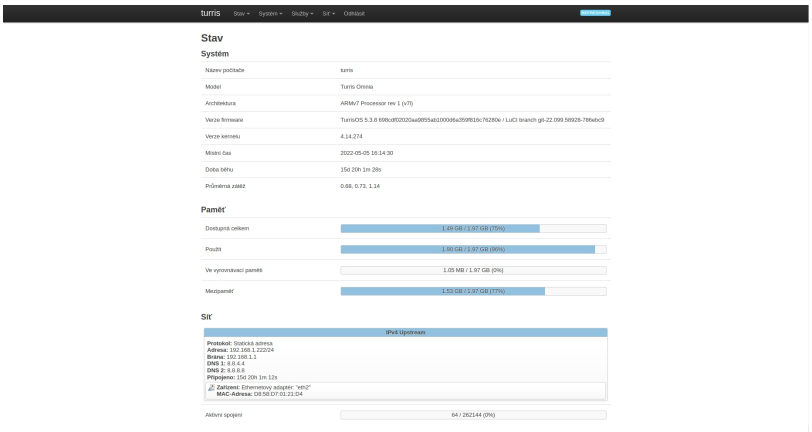
reForis Grafické rozhraní reForis je vyvíjena přímo autorem routeru společností CZ.NIC a obsahuje pouze ty nejzákladnější operace se zařízením tak, aby je byl schopen provádět i běžný uživatel bez větších znalostí práce se síťovými zařízeními. Kromě úplně prvotního nastavení funkčnosti směrovače se zde nastavuje heslo, které se sdílí mezi oběma webovými rozhraními i přístupem přes terminálovou aplikaci `ssh`, najdeme tu také nastavení WAN, LAN, WiFi nebo DNS, zálohování systému a v neposlední řadě také přehled s možností instalace základních balíčků. Mezi balíčky pro NAS či OpenVPN se zde nachází sada služeb

LXC pro virtualizaci se sdíleným jádrem, což budeme potřebovat pro vytváření kontejnerů s jednotlivými implementacemi.



Obrázek 3.1: Vzhled základního webového uživatelského rozhraní reForis.

LuCi Webové rozhraní LuCi je připraveno pro přístup k administraci zařízení běžících na operačním systému OpenWRT. Funkčnost aplikace reForis tak rozšiřuje o další konfigurovatelné podrobnosti a přidává nastavení například statických tras, firewallu, plánování úloh službou cron nebo poskytuje přístup k záznamům z jádra a dalším systémovým logům. Z tohoto systému lze také vytvářet a dále spravovat LXC kontejnery, to je ale možné provádět i přes příkazovou řádku, která k tomu byla v práci zvolena jako uživatelsky příjemnější, protože grafické rozhraní nedokázalo vrátit uživateli zpětnou vazbu o chybě v nastavení konfiguračního souboru daného kontejneru.



Obrázek 3.2: Vzhled pokročilého webového uživatelského rozhraní LuCi.

3.2.2 Kontejnerová řešení

Pro oddělení více implementací řídicích aplikací pro chytrou domácnost byl zvolen princip kontejnerizace. K tomu využijeme službu LXC, která nám umožní vytvořit více kontejnerů na sdíleném jádře, přičemž nespornou výhodou je schopnost každého kontejneru běžet na jiném operačním systému a mít přesnou konfiguraci programů v potřebných verzích tak,

aniž by se implementace chytrých domácností vzájemně překrývaly a tím ovlivňovaly. I když bychom mohli spatřit komplikaci v souběžném provozu více řídicích systémů, je toto spíše výhodou pro uživatele – každý z nich může využívat systém, který mu z pohledu grafického uživatelského rozhraní vyhovuje více.

Nelze pominout také bezpečnostní stránku z hlediska vnějšího přístupu uživatelů. Vzhledem k tomu, že centrální jednotka poběží na směrovači, můžeme z jednoho místa spravovat přístupy do jednotlivých kontejnerů v závislosti na stanovených podmínkách. Můžeme tak stanovit firewall pravidla pro přístup z vybraných IP adres, doplnit autentizací a podobně. Tímto můžeme řešit typ útoků MITM (*man-in-the-middle*), kdy by se útočník mohl dostat ke komunikaci mezi řídicí jednotkou a prvky nižší úrovně chytré domácnosti a kde by následně mohl toto ovlivňovat a převzít kontrolu nad chytrou domácností.

Pro vytvoření a správu kontejnerů budeme využívat příkazovou řádku na směrovači přes protokol SSH. Kontejner vytvoříme příkazem `lxc-create -n jméno -t download`, kde vlajka `-n` představuje jméno a `-t` typ šablony kontejneru, v tomto případě tedy `download` značí, že se stáhne seznam dostupných obrazů operačních systémů. V následujících krocích vybereme architekturu a distribuci OS. Po procesu vytvoření kontejneru můžeme všechny tyto zobrazit příkazem `lxc-ls`. [1]

```
1  lxc.arch = armv7l
2  lxc.include = /usr/share/lxc/config/common.conf
3  lxc.hook.start-host = /usr/share/lxc/hooks/systemd-workaround
4  lxc.net.0.type = veth
5  lxc.net.0.link = br-lan
6  lxc.net.0.flags = up
7  lxc.net.0.name = eth0
8  lxc.net.0.hwaddr = 42:bb:3e:38:28:7d
9  lxc.rootfs.path = dir:/srv/lxc/nodered/rootfs
10 lxc.uts.name = nodered
```

Zdrojový kód 1: Základní konfigurace LXC kontejneru [1]

Ve zdrojovém kódu počáteční konfigurace každého kontejneru můžeme vidět základní nastavení jeho architektury, úvodního procesu (jímž je démon `systemd`), síťového rozhraní, složky souborového systému a samotný název – tento kontejner je využíván pro aplikaci Node-RED. Tento soubor budeme později dále rozšiřovat o zvýšení práv kontejneru a vytvoření přípojného bodu USB zařízení. Konfigurační soubor leží ve složce vyhrazené pro kontejner, na stejné úrovni jako složka se souborovým systémem, zde se tedy soubor nachází v cestě `/srv/lxc/nodered`. Pokud chceme automaticky kontejner spustit po startu směrovače bez nutnosti ručního spouštění příkazem `lxc-start název_kontejneru`, do souboru `/etc/config/lxc-auto` doplníme následující kód pro každý takový kontejner.

3.2.3 Statické DHCP zápůjčky a mapování na jména

Na naší jednotce Turris Omnia je nakonfigurován a spuštěn DHCP server. Abychom u kontejnerů a dalších prvků chytré domácnosti nemuseli řešit proměnlivé adresování způsobené vypršením doby zápůjčky IP adresy, využijeme k tomuto účelu statické přiřazení IP adres,

```
config container
option name nodered
option timeout 60
```

Zdrojový kód 2: Automatické spuštění kontejneru po startu LXC služeb. [1]

jejich nastavení umožňuje webové rozhraní LuCi. Zde na úrovni MAC adresy spárujeme konkrétní kontejner či prvek domácnosti s fixní IP adresou s nekonečnou dobou zápůjčky.

V návaznosti na toto propojení ještě můžeme v LuCi vytvořit také lokální doménová jména, abychom k zařízením v síti mohli přistupovat přes slovní názvy a nemuseli si pamatovat všechny jejich IP adresy.

3.3 Užívané aktory a senzory

Aby bylo možné zkoušet a pozorovat řídicí systémy chytré domácnosti při fungování nad reálnými daty, byly pořízeny následující prvky chytré domácnosti, jejichž výběr takřka pokrývá ty nejzásadnější požadované komponenty. Při nákupu byl brán ohled na dostatečnou otevřenost tak, aby k těmto zařízením bylo dodáno aplikační rozhraní, skrz které bude možné jej propojit přímo s řídicím centrem bez nutnosti využití dalších převáděcích jednotek (bran).

3.3.1 Aktory

Žárovka Yeelight Chytrá LED Yeelight W3 je WiFi žárovka s nastavitelnou úrovní jasu a barvou světla (v modelu RGB), komunikující prostřednictvím UDP a TCP požadavků, jejichž obsahem je kód ve formátu JSON obsahující id požadavku, název invokované metody a dodatečné parametry. Například pro přepnutí stavu žárovky do opačného stavu odešleme aplikací `telnet` na IP adresu zařízení na port 55443 požadavek ve formátu `{"id":1,"method":"toggle","params":[]}`.

WiFi relé Shelly Spínacím relé Shelly 1 lze ovládat světla, celé okruhy elektroinstalace i další zařízení. Relé je napájeno z elektrické sítě napětím 230 V, spínání je řízeno WiFi, a to pomocí připraveného webového REST API, na které jsou zasílány HTTP požadavky. Některé koncové body jsou připraveny na dotazy typu GET, na které je odpovězeno údaji o relé, některé akceptují požadavek typu POST s doplňujícími daty. Součástí firmware je i webové rozhraní, přes které lze komponentu spravovat stejně jako přes REST API.

Obrázek 3.3: Vzhled webového uživatelského rozhraní relé Shelly.

2 kanálové USB relé Tento typ relé je spínán přes sběrnici USB a současně je odtud napájen napětím 5 V. Pro použití není nutná instalace ovladače, protože spadá do třídy zařízení HID (*Human Interface Device*), k jeho ovládání stačí instalace knihovny pro komunikaci s těmito zařízeními a program pro příkazový řádek `usbrelay`. Abychom mohli se zařízením pracovat i uvnitř kontejneru, musíme do jeho konfiguračního souboru vložit přípojný bod s tímto zařízením.

```
1 lxc.cgroup.devices.allow = c 189:* rwm
2 lxc.cgroup.devices.allow = c 249:* rwm
3 lxc.mount.entry = /dev/bus/usb dev/bus/usb none bind,optional,create=dir 0 0
4 lxc.mount.entry = /dev/hidraw0 dev/hidraw0 none bind,optional,create=file 0 0
```

Zdrojový kód 3: Zpřístupnění USB zařízení uvnitř LXC kontejneru [10]

3.3.2 Senzory

Venkovní a vnitřní teploměr Teploměr pro měření venkovní i vnitřní teploty byl vytvořen podle návodu k projektu Temp2IoT od autora 100przt dostupného v repozitáři GitHub [1]. Dvě digitální teplotní čidla DS18B20 se přes komunikační sběrnici 1-Wire připojí na GPIO vývod WiFi modulu s čipem ESP8266. Uvnitř této jednotky je spuštěn firmware naprogramovaný v jazyce C++ a prostředí Arduino, který periodicky získává data ze senzorů a sdílí je na vlastním webovém serveru s grafickým rozhraním i REST API. V případě potřeby získání dat pak stačí zaslat HTTP požadavek na toto rozhraní.

PIR Čidlo Pohybové čidlo na svém výstupu vrací hodnotu 1 nebo 0 v závislosti na detekovaném pohybu. Protože již existuje funkční WiFi modul s webovým serverem, je nejjed-

[1] https://github.com/100przt/Temp2IoT

nodušší tohoto využít a připojit výstup PIR čidla na jiný GPIO vývod modulu. K správné funkčnosti je třeba doplnit kód o čtení z GPIO a aktualizaci hodnoty na výstupním REST API.


```
{
        "systemname": "",
        "secure_counter": 115710,
        "firmware": "2.3.04-b",
        "sensors": [
        {
                "name": "Uvnitř",
                "value": 24.1875,
                "unit": "Celsius",
                "time": "Fri Apr  15 11:21:38 2022"
        },
        {
                "name": "Venku",
                "value": 18.6875,
                "unit": "Celsius",
                "time": "Fri Apr  15 11:21:38 2022"
        }
        ],
        "PIR": true
}
```


Zdrojový kód 4: Výstup ve formátu JSON z REST API pro teploměry a PIR čidlo.

Vnitřní kamera WiFi kamera TP-LINK Tapo C200 poskytuje živý přenos protokolem RTSP (*Real Time Stream Protocol*). Tento přenos je pak nutné dále zpracovat, například knihovnou `ffmpeg`.

3.3.3 Cenový přehled komponent

Produkt	**Cena s DPH**
WiFi router Turris Omnia 2020	7 502 Kč
LED žárovka Yeelight LED Smart Bulb W3 (color)	447 Kč
WiFi spínač Shelly 1, spínací modul 1 × 16A	337 Kč
2 × DS18B20 Digitální vodotěsné čidlo teploty 1 m	112 Kč
IoT ESP8266 Lua NodeMcu V2 WIFI modul	138 Kč
IP kamera TP-Link Tapo C200 - bílá	740 Kč
Modul relé USB, 2 kanálový s Attiny45	125 Kč
Detektor pohybu, modul PIR HC-SR505	65 Kč
Celková cena	**9466 Kč**

Tabulka 3.1: Součástky zakoupené k realizace chytré domácnosti, duben 2022.

Kapitola 4

Řídicí aplikace sestavená z Eclipse IoT

Cílem této kapitoly je navrhnout a následně implementovat řídicí aplikaci chytré domácnosti s využitím projektů *Eclipse IoT*. Tato značka sdružuje komunitu vývojářů pracujících na mnoha aplikacích napříč odvětvím Internetu věcí. Velký důraz je v Eclipse IoT kladen na open-source přístup, což má vliv na využití standardních komunikačních protokolů. Z toho důvodu je možné vyvinuté projekty mezi sebou spojovat do větších celků a vytvářet komplexní řešení pro chytré domácnosti i průmysl.

Aby bylo možné sestavit architekturu řídicího systému, je nutné se zaměřit na komunikaci mezi koncovými prvky (senzory, aktory ad.) s logickou a prezentační vrstvou. Pokud jde o nízkoúrovňové prvky, připojené například přes sběrnici I2C, bude komunikace zprostředkována přes gateway (vstupně-výstupní bránu) do cloudu. To znamená, že čidla budou připojena přímo do centrální řídicí jednotky v domě. Pokud bude prvek schopen přímo komunikovat skrz protokoly HTTP nebo MQTT, může být připojen přímo do cloudu.

4.1 Testované aplikace

Před návrhem finálního systému autor otestoval několik projektů Eclipse IoT aby měl o těchto přehled a mohl zvolit co nejvýhodnější kombinaci aplikací. Přestože bylo původně plánováno využívat prostředí LXC kontejnerů, z důvodu dostupnosti většiny aplikací Eclipse IoT na serveru Docker Hub bylo přistoupeno k využití těchto kontejnerů spravovaných programem Docker.

4.1.1 Eclipse Kura

Možnosti využití Eclipse Kura byly rozsáhle popsány v kapitole 2.5.2, nejdůležitější je ale její specializace na gateway a zpracování dat od čidel připojených přes sériové linky, sběrnice I2C, SPI a podobně. Často je tak tato aplikace instalována na zařízení Raspberry Pi, které tvoří vstupně-výstupní bránu, případně pokud by bylo třeba, může být těchto hardwarových bran v domě více, každé bude zodpovědné za svůj podsystém senzorů/aktorů a na každém bude spuštěna instance Eclipse Kura.

Připravený obraz kontejneru je dostupný na Docker Hubu pod názvem eclipse/kura, ale pouze v architektuře `x86_64`, což neumožňuje provoz na směrovači Turris Omnia s procesorem typu ARM, proto bude tato i všechny následující aplikace spouštěny na autorově notebooku s procesorem Intel nad operačním systémem OS Ubuntu 20.04 LTS.

Po spuštění kontejneru `docker run -d -p 8085:8080 -t -name kuraDocker eclipse/kura` se v prohlížeči na adrese `localhost:8085` spustí webové rozhraní, v jehož horní části levého menu se nachází síťová nastavení zařízení, konfigurace brány firewall a také služby pro odesílání dat do cloudu, takzvané *Cloud services*. V této části je třeba zadat adresu a autentizační údaje k brokerovi, který data předává cloudové službě. Takových brokerů může běžet několik současně. Spodní část navigace pak představují jednotlivé služby, zajišťující komunikaci s různými druhy čidel podle komunikačního standardu.

4.1.2 Eclipse Hono a Apache Kafka

Aplikace Eclipse Hono, diskutována v kapitole 2.5.3, poskytuje online sandbox, na kterém si může vývojář přes terminál vyzkoušet fungování aplikace, a to od vytvoření klienta až k simulaci zasílání zpráv do jednotlivých adaptérů.

Takto vytvořený výstup může být k prezentační vrstvě přenášen brokerem Apache Kafka. Systém je využíván v odvětvích, kde z různých důvodů nestačí klasické brokery zpráv. Pokud by byl srovnán s Mosquitto brokerem, fungujícím nad MQTT, zásadní budou tyto rozdíly:

- Výměna zpráv přes Mosquitto funguje na principu *publish–subscribe*. To znamená, že příjemce se musí přihlásit k naslouchání na daném tématu, v opačném případě zprávy neobdrží. Apache Kafka umožňuje pracovat s toky dat i zpětně, takže pokud klient nebyl přihlášen k odběru tématu, nemusí to znamenat, že se k těmto zprávám již nedostane.
- Na rozdíl od Mosquitto brokera je systém Apache Kafka připraven na velkou škálovatelnost. V kombinaci s aplikací Apache Zookeeper lze vytvářet velké clustery mnoha brokerů, čímž lze dosáhnout opravdu velké propustnosti zpráv. [16]
- Výhodou těchto clusterů u Apache Kafka je také dostupnost dat. Protože jsou brokery propojeny v aplikaci Zookeeper, jsou schopny mezi sebou přenášet data v případě výpadku některého uzlu. [16]

Přestože software Apache Kafka nabízí celou řadu výhod, pro realizaci chytré domácnosti je mnohem více komplikovaný než některý z klasických MQTT brokerů. Také při pohledu na projekty Eclipse IoT lze pozorovat, že většina všech aplikací má implementovány MQTT brokery pro vzájemnou komunikaci, zatímco Apache Kafka je spíš výjimkou.

4.1.3 Eclipse Kapua

Aplikace poskytuje komunikační rozhraní, přes které přijímá data z vstupně-výstupních bran a ta agreguje pro jejich další využití na prezentační vrstvě. Aplikace je dostupná v repozitáři GitHub[1], pro její použití je třeba repozitář naklonovat pomocí `git clone`, poté lze spouštět v příkazové řádce shell skriptem `./deployment/docker/unix/docker-deploy.sh`, volaným v nejvyšší složce hierarchie.

4.1.4 Eclipse Streamsheets

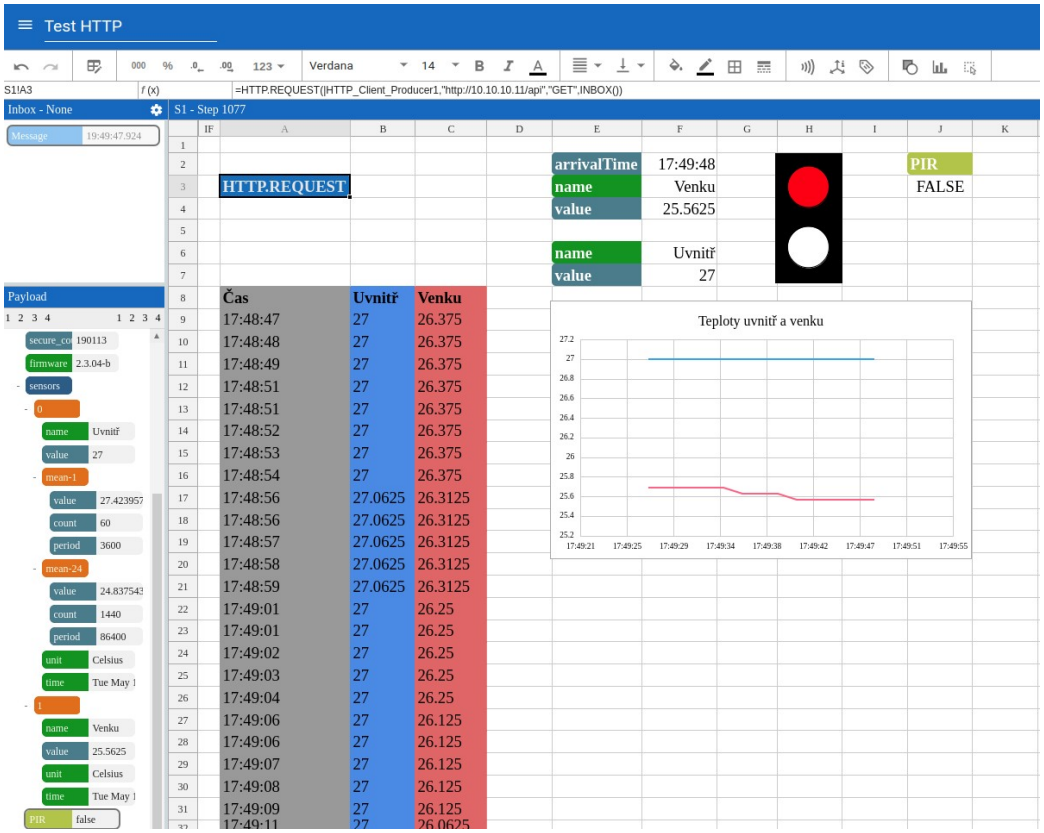
Eclipse Streamsheets poskytuje webové grafické rozhraní, v kterém umožňuje vizuálně prezentovat přijatá data formou tabulek, grafů i jiných objektů a lze přes ni také data odesí-

[1] https://github.com/eclipse/kapua

lat směrem k čidlům chytré domácnosti. Vzhled aplikace je graficky podobný standardním tabulkových procesorům, které uživatel využívá v běžném životě, což zvyšuje uživatelskou přívětivost a šanci, že uživatel pochopí práci s aplikací v krátkém čase. Aplikace je dostupná na Docker Hubu jako `cedalo/streamsheets`. Po instalaci je spuštěna příkazem `docker run -d -p 1888:1883 -p 27017:27017 -p 6379:6379 -p 8084:8080 -p 8086:8081 -p 8083:8083 -p 8088:8088 -p 9001:9000 cedalo/streamsheets:latest`, který spustí několik služeb, kde na portu `8086` běží webové rozhraní, do kterého se lze přihlásit s uživatelským jménem *admin* a heslem *1234*.

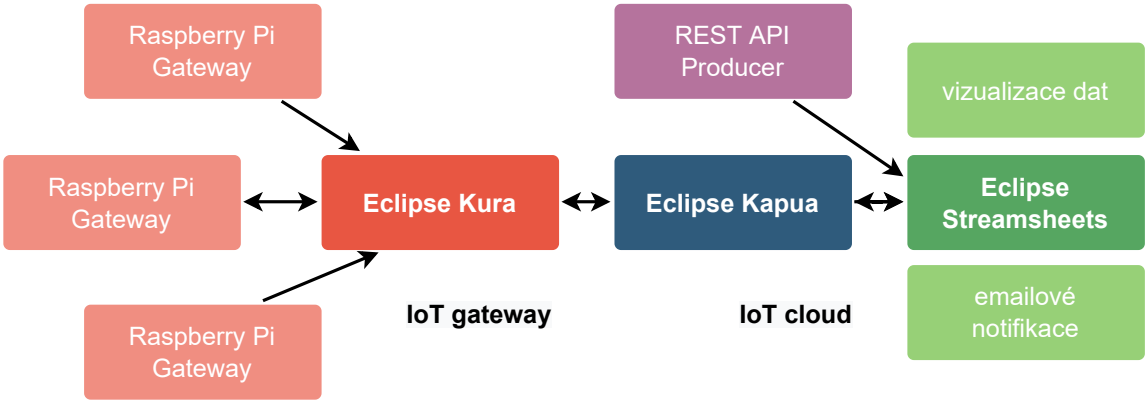
Po přihlášení může uživatel přidávat a editovat tabulky sestav, uvnitř kterých tvoří nástěnky s daty chytré domácnosti. K získání dat je nutné nakonfigurovat v každém listu konzumenta zpráv (v případě, že je čidlo schopno zasílat zprávy) nebo producenta, který bude v daných intervalech zasílat požadavek na nová data přes smluvený protokol.

Na obrázku 4.1 je vidět vzhled aplikace Eclipse Streamsheets podobající se aplikacím Microsoft Excel či Google Sheets. V editačním poli nad tabulkou se nachází vzorec `=HTTP.REQUEST(|HTTP_Client_Producer1,"http://10.10.10.11/api","GET",INBOX())`, který zajišťuje pravidelné získávání dat z REST API. V levém sloupci je vypsán seznam všech atributů navráceného objektu, které lze dále umisťovat do tabulek.



Obrázek 4.1: Vzhled webového rozhraní Eclipse Streamsheets, v kterém lze vytvářet ovládací panely.

4.2 Architektura výsledného řešení



Obrázek 4.2: Diagram architektury Smart home z projektů Eclipse IoT.

Pro výslednou architekturu byly vybrány aplikace *Eclipse Kura*, *Eclipse Kapua* a *Eclipse Streamsheets*. Na gateway zařízení je zprovozněna aplikace Eclipse Kura, která zajišťuje zpracování dat z čidel. Protože žádná z pořízených součástek k praktické realizaci chytré domácnosti nekomunikuje nabízenými standardy, je funkcionalita simulována pomocí modulu Example Publisher, který pravidelně odesílá data. Tato aplikace předává výstupním modulem data pomocí protokolu MQTT dále do cloudu, kde běží aplikace Eclipse Kapua. Pokud by v domácnosti existovala více než jedna gateway (pro případ většího počtu čidel napříč domem), musí všechny data odesílat do této cloudové části. Nad touto aplikací je instalována aplikace Eclipse Streamsheets, která se opět pomocí MQTT brokera připojí k datům v cloudu a z těchto bude následně možné vytvářet tabulkové přehledy.

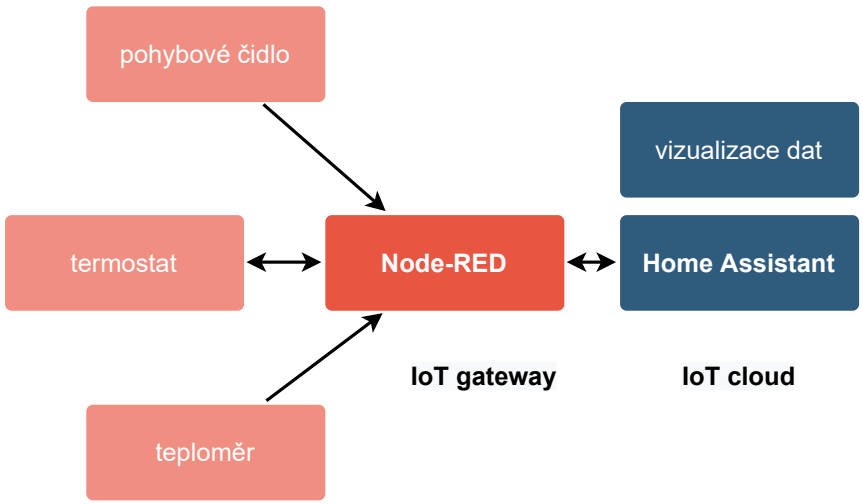
Prakticky je celé toto řešení realizováno Docker kontejnery, uvnitř kterých běží Eclipse aplikace nakonfigurované tak, aby se po zavolání startovacího skriptu spustily všechny kontejnery a ihned si začaly vyměňovat data.

Kapitola 5

Řídicí aplikace sestavená z Node-RED a Home Assistant

5.1 Výběr aplikací a konfigurace

Pro referenční řídicí systém byly vybrány dvě open-source nekomerční aplikace, které jsou pro konstrukci chytrých domů velmi používané. Obě řešení se svojí funkčností částečně překrývají, ale současně každá má své bližší zaměření. Zatímco Node-RED je editor, kde spojením uzlů lze vytvářet scénáře vhodné pro automatizace a tvorbu rutinních činností. Home Assistant se více zaměřuje na vytváření grafického uživatelského rozhraní, tedy nástěnek pro uživatele, z kterých je domácnost řízena. Společnou doménou obou řešení je pak množina rozhraní nabízených pro komunikaci s aktivními prvky. Výhodou je možnost kombinace a využití silných stránek každé z aplikací.



Obrázek 5.1: Diagram architektury Node-RED a Home Assistant.

5.2 Instalace Node-RED

Před samotnou instalací software Node-RED je nutné vytvořit LXC kontejner dle postupu v kapitole 3.2.2 a následně nastavit jeho automatické spouštění při zapnutí směrovače. Jako

šablona operačního systému pro kontejner byla vybrána **Ubuntu 20.04 LTS Focal**, jakožto operační systém založený na distribuci Debian, z které je vyvinut taktéž Raspbian, operační systém desek Raspberry Pi. To znamená, že ačkoli směrovač pracuje nad operačním systémem OpenWRT, v kontejneru je využíváno Ubuntu, které má s OS Raspbian společným balíčkovací systém `dpkg` a tedy lze předpokládat snadnou přenositelnost mezi distribucemi právě díky dostupnosti ve správci balíčků.

Vlastní instalaci Node-RED lze provést více způsoby.

1. Pomocí správce balíčků `npm` v případě, že je tento na zařízení dostupný.
2. Instalačním skriptem, který je k dispozici na stránkách aplikace a je propojen přímo se vzdálenými repozitáři.
3. Stažením obrazu z portálu *Docker Hub* a spuštěním přes aplikaci pro správu kontejnerů `docker`.
4. Sestavením přímo z kódů dostupných v repozitářích.
5. Spuštěním aplikace uvnitř cloudu pomocí některého z komerčních řešení *IBM Cloud*, *Microsoft Azure* a dalších.

V této práci bylo využito metody pomocí instalačního skriptu. Během tohoto postupu vyžadoval skript zadání uživatelského jména a hesla. Autentizace byla ale později úplně zrušena – toho bylo docíleno zakomentováním vlastnosti `adminAuth` v sekci *Security* uvnitř konfiguračního souboru pro Node-RED `setting.js`, umístěného na disku v cestě `/root/.node-red`. Uvnitř tohoto souboru se nachází mimo jiné konfigurace uživatelského rozhraní, síťová nastavení serveru včetně možnosti nasazení certifikátu pro provoz na HTTPS a je zde také cesta k souboru se všemi funkčními uzly aplikace. Ve výchozím nastavení tu figuruje také port 1880, na kterém aplikace běží.

Po instalaci je vhodné nastavit také automatické spouštění Node-RED pomocí systémového správce `systemd`, konkrétně zadáním `sudo systemctl enable nodered.service` do příkazové řádky. V opačném případě je nutné aplikaci spouštět přes terminál příkazem `node-red-start`, pro zastavení či restart lze využít příkazy `node-red-stop`, respektive `node-red-restart`. Po ručním spuštění se v terminálu objeví IP adresa a port, na kterém Node-RED běží. Tuto adresu stačí poté otevřít v internetovém prohlížeči, a pokud existuje správné mapování IP adres na doménová jména (viz kapitola 3.2.3), lze toto využít. V této práci je tak aplikace dostupná na URL adrese `http://nodered:1880`.

5.3 Konfigurace komponent

Při prvním spuštění webového rozhraní je k dispozici průvodce aplikací. Po jeho dokončení je vidět pracovní plocha aplikace. V horní části obrazovky nad mřížkou pro uzly se nachází pás karet, které lze využívat při rozdělení toků dat pro zvýšení přehlednosti. Aplikace funguje na principu přetažení uzlů z levé části do mřížky v hlavním prostoru. Tyto komponenty se poté nastavují v dialogovém okně zobrazeném po dvojitém kliku myší na uzel v mřížce. Uzly se propojují natažením spojovací čáry mezi dvěma komponentami. Po stažení aplikace se v levém panelu komponent nachází jejích základní výběr. Uživatel má ale možnost doinstalovat moduly, dostupné ze správce balíčků `npm`, které rozšiřují funkčnost Node-RED. Implementované funkce v těchto modulech se pak uživateli zobrazí jako nově

dostupné uzly v levém sloupci. Pro uložení práce a současné nasazení vytvořených toků musí uživatel kliknout na tlačítko *Deploy* v pravém horním rohu.

Všechny údaje o veškerých umístěných komponentách i vazbách mezi nimi se ukládají jako pole objektů do souboru `flows.json`. Ten je umístěn ve stejném adresáři jako soubor s nastavením aplikace. Zdrojový kód 5 představuje úsek z `flows.json`, který popisuje uložení komponenty pro HTTP požadavek. Každý z uzlů obsahuje nutné vlastnosti jako identifikátor a typ objektu, jeho `x` a `y` souřadnice umístění v mřížce nebo pole vazeb z dalším uzlům adresované právě pomocí ID objektů. V závislosti na typu objektu jsou pak ukládány další parametry, u tohoto HTTP požadavku například metoda, typ návratové hodnoty či URL adresa, kam bude zaslán. Výhodou tohoto způsobu uložení objektů je snadný export/import, čehož lze využít nejen při migraci těchto scénářů mezi zařízeními, ale také na internetových diskuzích, kde příznivci aplikace Node-RED vzájemně sdílí zdrojové kódy jimi vytvořených automatizací.

```
{
        "id": "c204f7728e37aef3",
        "type": "http request",
        "z": "87748fade4fbaeba",
        "name": "",
        "method": "GET",
        "ret": "txt",
        "paytoqs": "ignore",
        "url": "10.10.10.11/api",
        "tls": "",
        "persist": false,
        "proxy": "",
        "authType": "",
        "senderr": false,
        "x": 770,
        "y": 500,
        "wires": [
        [
        "4492a89c91d46be1"
        ]
        ]
}
```

Zdrojový kód 5: Uzel Node-RED pro HTTP požadavek uložený ve formátu JSON.

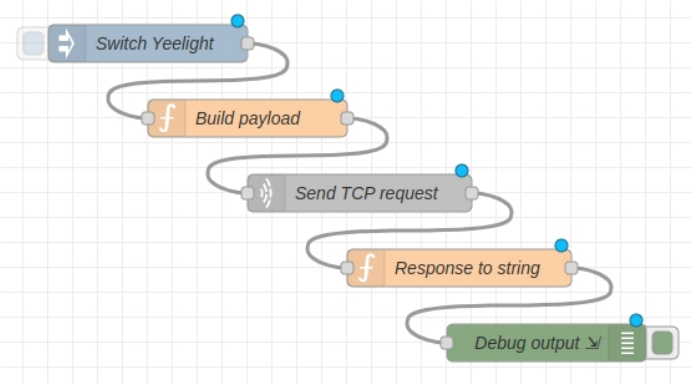
5.3.1 Žárovka Yeelight

Pro používání LED svítidla Yeelight s Node-RED byl nejdříve otestován dostupný modul `node-red-contrib-yeelight`. Jeho součástí je konfigurační uzel, který umožňuje zadat IP adresu žárovky. Do tohoto uzlu potom stačí poslat kód s volanou metodou a případnými parametry. Práce s tímto modulem byla sice pohodlná, ale po prvních momentech používání začal hlásit chybu připojení k Yeelight žárovce z důvodu vypršení času, což vedlo k pádu

aplikace Node-RED. Toto se opakovalo i po dalších změnách v nastavení, takže bylo rozhodnuto tento modul nevyužít a přistoupit k přímému ovládání žárovky protokoly TCP a UDP dle dodané specifikace společnosti Yeelight.

Před výslednou realizací pomocí prostředků Node-RED byla funkčnost otestována samostatně. Nejdříve bylo potřeba vyhledat dané zařízení v síti, což bylo řešeno jednoduchým UDP serverem implementovaným v Pythonu. Ten začal naslouchat na definovaném portu, odeslal HTTP požadavek typu M-SEARCH s hlavičkami dle dokumentace na určenou multicastovou adresu a poté očekával unicastový UDP datagram od všech WiFi svítidel, která se v síti nachází. Tyto odpovídají návratovým kódem *200 OK* ze svých IP adres na stejnou adresu i port jako v dotazu a uvnitř hlaviček zasílají podrobnosti o stavu daného zařízení – jeho model, verzi firmware, podporované názvy metod, jestli právě svítí, nastavenou intenzitu svitu v rozmezí 0–100, barevnost v modelu RGB a další parametry v závislosti na konkrétním typu zařízení. Přímé ovládání pomocí takto zjištěných metod bylo popsáno v odstavci o zařízení Yeelight v kapitole 3.3.1.

Jakmile byla otestována funkčnost ovládání dle specifikace, byl vytvořen scénář uvnitř Node-RED, který sestavil požadavek pro přepnutí stavu lampy, odeslal jej na její adresu a vypsal vrácenou odpověď do ladicího okna.



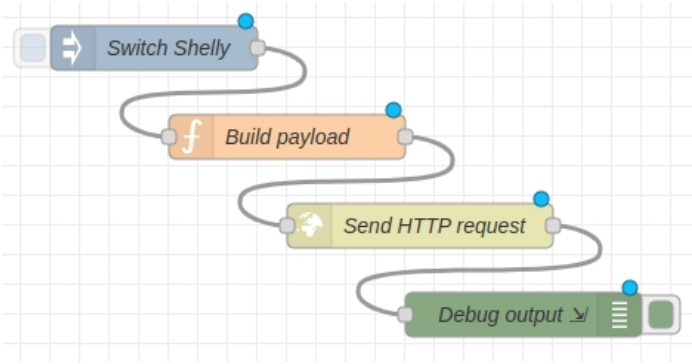
Obrázek 5.2: Scénář pro zapnutí/vypnutí žárovky Yeelight.

5.3.2 WiFi relé Shelly

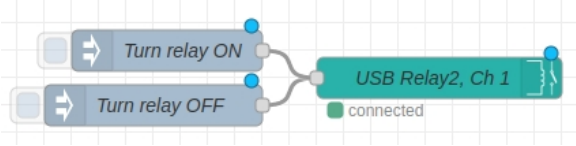
Vzhledem k jednoduchému ovládání WiFi relé Shelly 1 přes REST API bylo využito základních komponent bez nutnosti dalších modulů. Byl sestaven HTTP požadavek typu POST, v jehož těle se ve formátu *x-www-form-urlencoded* nachází jméno volané procedury a hodnota, v tomto případě funkce `turn` s parametrem `toggle` pro přepnutí stavu relé do opačného nezávisle na aktuálním.

5.3.3 2 kanálové USB relé

Pro řízení USB relé bylo použito řešení v podobě modulu `node-red-contrib-usb-hid-relay`, jenž zajišťoval komunikaci s USB HID zařízením. Pro jeho správnou funkčnost bylo třeba zpřístupnit USB zařízení uvnitř kontejneru pomocí tzv. mountování, nainstalovat v něm knihovny pro ovládání USB a přidělit přístup k těmto zařízením vytvořením pravidla pro udev (správce zařízení v jádru Linux). Ovládání je následně zjednodušeno na zaslání boolovské hodnoty do uzlu USB relé. Při hodnotě `true` je relé sepnuto a analogicky u hodnoty `false`.



Obrázek 5.3: Scénář pro zapnutí/vypnutí WiFi relé Shelly.



Obrázek 5.4: Scénář pro zapnutí/vypnutí USB relé.

5.4 Tvorba nástěnek

Modul `node-red-dashboard` umožňuje tvorbu nástěnek přímo uvnitř Node-RED. Ty lze skládat jak z výstupních prvků pro prezentaci hodnot z chytrých zařízení, tak ze vstupních ovládacích prvků pro jejich ovládání. Modul poskytuje pouze omezené množství těchto prvků, což ale může na většinu projektů dostačovat. Protože je v této práci využito kombinace Node-RED a Home Assistant, nebyl tento modul dále využíván.

5.5 Instalace Home Assistant

Před instalací aplikace Home Assistant bylo třeba připravit další LXC kontejner stejně jako v kapitole 5.2. Opět byl zvolen operační systém Ubuntu 22.04 LTS. Instalace byla provedena podle návodu v článku *Jak nainstalovat Home Assistant Core na router Turris Omnia*[1], který od kroku číslo 21 přesně popisuje celý postup. Po instalaci lze spustit webové rozhraní aplikace na portu 8123. Při prvním spuštění je nutné vytvořit uživatelský účet včetně zvolení hesla.

5.6 Konfigurace komponent

Stěžejní konfigurační soubor aplikace `configuration.yaml` je v kontejneru umístěn v cestě `/home/homeassistant/.homeassistant`. V tomto souboru se kromě základních nastavení nachází konfigurace všech entit, s kterými je dále manipulováno ve webovém rozhraní. Aby uživatel nemusel zadávat heslo při každém spuštění webu, je nutné zde definovat nový typ autentizace, který funguje na principu uvedení IP adresy důvěryhodné podsítě, v jejímž rozsahu aplikace nepožaduje přihlašování. [26]

[1] https://tatageek.blog/2021/12/23/jak-nainstalovat-home-assistant-core-na-router-turris-omnia/

```
homeassistant:
  auth_providers:
    - type: trusted_networks
    trusted_networks:
      - 10.10.10.0/24
```

Zdrojový kód 6: Konfigurace autentizace metodou důvěryhodné podsítě.

5.6.1 Žárovka Yeelight

Home Assistant umožňuje využívání takzvaných integrací, některé z nich lze nastavovat přímo z webového rozhraní bez nutnosti editace konfiguračního souboru. Mezi těmito integracemi se vyskytuje i Yeelight pro ovládání WiFi LED lampy. Po instalaci integrace stačilo zadat IP adresu a ostatní bylo vyřešeno na pozadí.

5.6.2 WiFi relé Shelly

WiFi relé bylo nakonfigurováno jako integrace `switch` na platformě REST. Parametry pro nastavení jsou využity stejně jako v kapitole 5.3.2.

5.6.3 2 kanálové USB relé

Oba typy relé využívají stejný typ integrace `switch`, který slouží pro dvoustavová zařízení, u řízení přes USB se však pracuje s příkazovou řádkou. Jak bylo řečeno u USB relé v kapitole 3.3.1, k ovládání lze využít terminálovou aplikaci `usbrelay`. Ta je v tomto případě volána z aplikace Home Assistant. Tento program musí být spouštěn s právy uživatele root, Home Assistant ale běží pod vlastním uživatelem. Problém lze vyřešit editací souboru `/etc/sudoers` a přidělením práv pro uživatele `homeassistant`.

5.6.4 Digitální teploměr a PIR čidlo

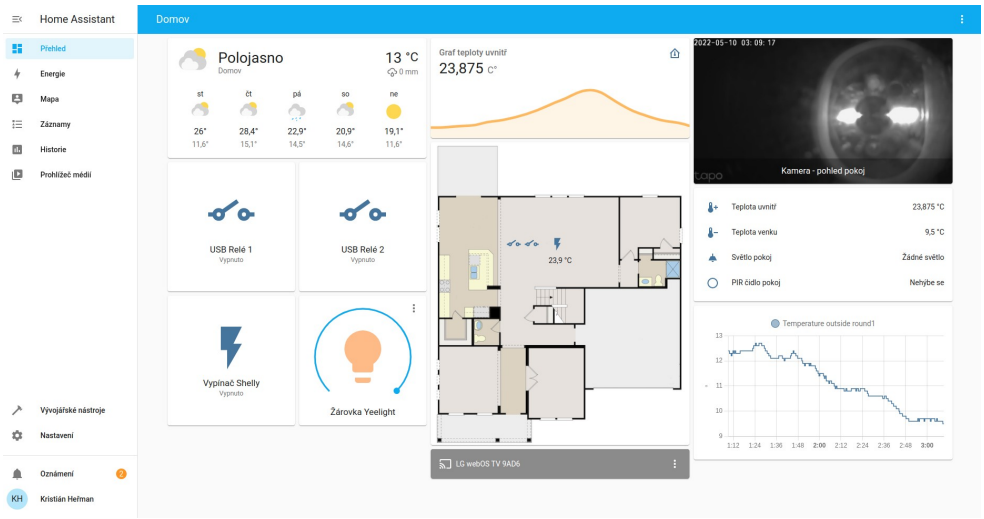
Data z digitálního teploměru lze získat snadno pomocí komponenty `sensor`. Ta využije opět REST API, s jehož pomocí obdrží JSON strukturu s daty. Po zpracování má aplikace k dispozici hodnoty vnitřní/venkovní teploty. U PIR čidla je využita komponenta `binary_sensor` pracující na stejném principu, pouze dokáže s výslednou binární hodnotou pracovat v grafickém rozhraní například změnou barvy ikonky nebo pokud je zadána takzvaná třída zařízení, může zobrazovat textový popis stavu – v případě pohybového čidla *Nehýbe se/V pohybu*.

5.6.5 Bezpečnostní kamera

Pro živý přenos z bezpečnostní kamery musí existovat entita typu `camera`, která z používaného modelu kamery *TP-LINK Tapo C200* dekóduje RTSP proud pomocí knihovny `ffmpeg` a připraví jej pro zobrazení v ovládacím panelu.

5.7 Tvorba nástěnek

Jakmile je entita zavedena v konfiguračním souboru, je možné ve webovém rozhraní vytvořit nástěnku. K úpravám ovládacího panelu se lze dostat v pravém horním rohu přes menu *Upravit ovládací panel*, *Přidat kartu*. Nyní je vybírána karta a po kliknutí nastaveny její atributy, například z které entity má čerpat data či její název na nástěnce.



Obrázek 5.5: Vzhled ovládacího panelu Home Assistant.

5.8 Propojení Node-RED a Home Assistant

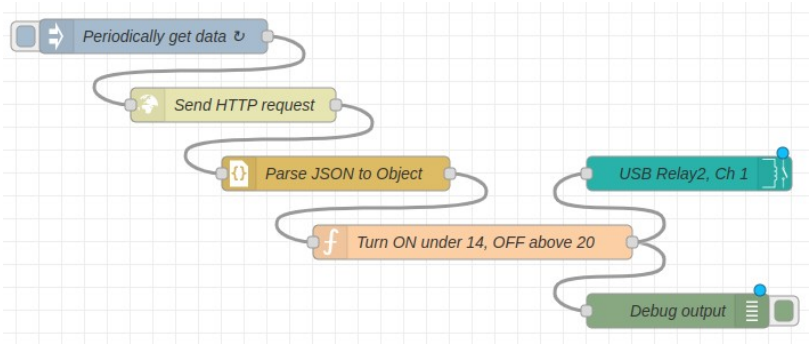
Automatizace mohou být vytvořeny čistě v Node-RED nebo lze využít definovaných čidel v Home Assistant, která pravidelně aktualizují svůj stav. Na změnu pak mohou Node-RED upozornit. Ten vykoná logickou část automatizace a svůj výsledek může opět předat zpátky aplikaci Home Assistant. Logickou úroveň tak může uživatel naprogramovat přímo v jazyce JavaScript, protože Node-RED poskytuje uzel s vlastní funkcí.

Pokud se uživatel rozhodne pro druhou z možností, musí obě aplikace propojit. K tomu slouží například modul `node-red-contrib-home-assistant-websocket`, jejž uživatel nainstaluje jako další rozšíření Node-RED a nastaví konfigurační uzel s údaji o serveru Home Assistant. Zásadní je IP adresa, port a přístupový token, který si uživatel dopředu v uživatelském rozhraní Home Assistant vygeneruje.

5.8.1 Vytváření automatizací

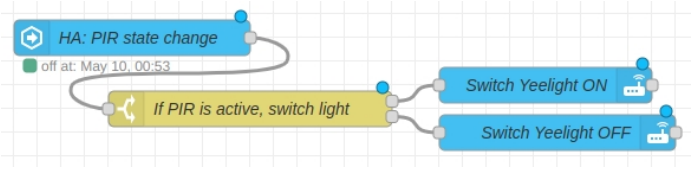
Na prvním příkladu automatizace je demonstrováno využití čistého Node-RED. Ten musí periodicky odesílat požadavek na data z teplotního čidla, ty zpracovat a na logické úrovni rozhodnout, jestli bude sepnuto relé a případně toto provést. Na toto relé může být v reálném řešení připojení třeba podlahové topení a touto automatizací jsme prakticky sestavili termostat.

U druhé možnosti je využita kombinace obou aplikací. Home Assistant sleduje pohybové čidlo a má o něm aktuální informace. Pouze v případě změny tak upozorní jednotku Node-RED, která data vyhodnotí a rozhodne, jestli se má zapnout či vypnout světlo v míst-



Obrázek 5.6: Scénář automatizace spínání topení dle teploty.

nosti. Tato logická vrstva ale nemusí vůbec umět světla obsluhovat, protože to zajistí Home Assistant přes své integrace.



Obrázek 5.7: Scénář automatizace zapnutí světel při pohybu.

Kapitola 6

Srovnání

Řídicí aplikace vytvořená prostředky Eclipse IoT je sestavena z tří aplikací, kde každá zajišťuje část procesu ovládání chytré domácnosti. Při kompletaci do celkového řešení je systém nepřehledný, což je způsobeno více faktory.

Jedním z důvodů je počet současně spuštěných služeb při startu celého systému. V tu chvíli je uživatel zmaten množstvím obsazených portů, kdy přesně neví, na kterém se nachází jaký podsystém.

Dalším problémem je množství přihlašovacích údajů, které musí využívat a nelze se k systému přihlašovat přes jeden centrální přístupový bod.

Škoda je také vzhledu grafických uživatelských rozhraní jednotlivých aplikací – ta vypadají pokaždé úplně odlišně včetně rozmístění hlavních ovládacích prvků, což uživatele rozptyluje a zdržuje při práci. Je ale nutné se zamyslet nad tím, že aplikace byly vyvíjeny v průběhu více let, a to velkým množstvím lidí, kteří se na projektech Eclipse IoT podíleli, proto nelze očekávat, že bude grafické rozhraní sjednocené a vyladěné profesionálními UI/UX designery.

6.1 Vzájemné srovnání aplikací

Uživatelská přívětivost

Eclipse IoT: kombinace více uživatelských rozhraní matoucí
Node-RED: intuitivně využívá ovládání myší k přidávání a spojování uzlů
Home Assistant: přívětivost úprav ovládacího panelu ve webovém rozhraní, v konfiguračním souboru nikoli

Potřeba znalosti oboru

Eclipse IoT: pro konfiguraci téměř nutná
Node-RED: částečně, možnost využití nápovědy
Home Assistant: pouze při ruční editaci konfigurace

Tvorba automatizací

Eclipse IoT: složité, nutné přes funkce v tabulkovém editoru
Node-RED: intuitivní, ale pro složitější logiku potřeba znalosti JavaScriptu
Home Assistant: výhodné v kombinaci s Node-RED, samostatně také, ale pouze ručně v konfiguračním souboru

Kapitola 7

Závěr

Cílem této práce bylo navrhnout a sestavit řídicí aplikaci pro chytrou domácnost s využitím prostředků Eclipse IoT. K této aplikaci bylo také potřeba připravit referenční řešení pro srovnání, které bylo postaveno s pomocí softwarů Node-RED a Home Assistant. Na začátku byly stanoveny podmínky, že by aplikace měla být jednak co nejvíce použitelná pro chytrou domácnost, současně by ale měla být jednoduše instalovatelná a konfigurovatelná. Po důkladné prostudování rodiny projektů Eclipse IoT bylo přistoupeno k samotnému návrhu architektury a poté i sestavení aplikací do jednoho komplexního řešení.

Podmínka použitelnosti byla splněna, aplikace lze využít ke komunikaci s reálnými čidly, což bylo ověřeno na získávání a prezentaci dat z WiFi teploměru a pohybového čidla. Bohužel se však nepodařilo splnit druhou podmínku – jednoduchost instalace a konfigurace. To je způsobeno různorodostí aplikací, které nebyly vyvíjeny pro přímou spolupráci mezi sebou.

Ve srovnání s typickým řešením pomocí Node-RED a Home Assistant jsou ale jasně vidět výhody tohoto referenčního řešení. Je pro uživatele přehlednější, lépe se v něm dokáže zorientovat a nejsou na něj kladeny vysoké nároky co se týče znalostí v oblasti síťových technologií, práce s kontejnery a podobně.

Nelze ale říct, že by řešení navržené na základech Eclipse IoT nebylo použitelné. Pouze si jej s největší pravděpodobností nevybere běžný uživatel s minimálními znalostmi informačních technologií. Pro tyto je mnohem výhodnější využít rozšířeného referenčního řešení Node-RED v kombinaci s Home Assistantem, nebo si zakoupí některé z komerčních řešení dostupných na trhu.

Literatura

[1] *LXC - Turris Documentation*. Praha: CZ.NIC. Dostupné z: https://docs.turris.cz/geek/lxc/lxc/.

[2] *Miniserver & Extensions - Loxone*. Loxone Electronics. Dostupné z: https://www.loxone.com/cscz/produkty/miniserver-extensions/.

[3] *Miniserver – řídicí jednotka*. Loxone Electronics GmbH. Dostupné z: https://shop.loxone.com/cscz/miniserver.html.

[4] *Nauč se programovat v Node-RED*. Dostupné z: https://www.hardwario.com/cs/education/tutorials/co-je-node-red/.

[5] *Node-RED: Home Assistant Community Add-on*. Dostupné z: https://community.home-assistant.io/t/home-assistant-community-add-on-node-red/55023.

[6] *[OpenWrt Wiki]: Welcome to the OpenWrt Project*. Dostupné z: https://openwrt.org/.

[7] *Overview of Hono Components*. Dostupné z: https://www.eclipse.org/hono/getting-started-kafka/.

[8] *Turris - Představení*. Praha: CZ.NIC. Dostupné z: https://www.turris.cz/cs/omnia/predstaveni/.

[9] *Turris - Specifikace*. Praha: CZ.NIC. Dostupné z: https://www.turris.cz/cs/omnia/specifikace/.

[10] *USB Passthrough to an LXC (Proxmox)*. Konpat Ta Preechakul. Dostupné z: https://medium.com/@konpat/usb-passthrough-to-an-lxc-proxmox-15482674f11d.

[11] *Z-Wave vs Zigbee vs WiFi | Which Is Best For Your Home?* Bedford: HomeSeer, 1999. Dostupné z: https://homeseer.com/zwave-vs-zigbee-vs-wifi/.

[12] *Thermometer*. San Francisco (CA): Wikimedia Foundation, 2001-. Dostupné z: https://en.wikipedia.org/wiki/Thermometer.

[13] *ZigBee*. San Francisco (CA): Wikimedia Foundation, 2001-. Dostupné z: https://cs.wikipedia.org/wiki/ZigBee.

[14] *MQTT - univerzální protokol pro cloudové a IoT aplikace*. Praha: HW Group, 2003. Dostupné z: https://www.hw-group.com/cs/podpora/mqtt-univerzalni-protokol-pro-cloudove-a-iot-aplikace.

[15] *Future generation computer systems.* Elsevier B.V, 2017. ISSN 0167-739X.

[16] *What is Zookeeper and why is it needed for Apache Kafka? - CloudKarafka, Apache Kafka Message streaming as a Service.* Stockholm: Elin Vinka, 2018. Dostupné z: https://www.cloudkarafka.com/blog/cloudkarafka-what-is-zookeeper.html.

[17] *Zigbee vs Z-Wave vs WiFi vs Bluetooth: What's Best?!* 2019. Dostupné z: https://www.smarthomepoint.com/zigbee-zwave-wifi-bluetooth-comparison/.

[18] *Eclipse Kura.* Ottawa: Eclipse Foundation, 2021. Dostupné z: https://www.eclipse.org/kura/.

[19] *4diac IDE.* Ottawa: Eclipse Foundation, 2022. Dostupné z: https://www.eclipse.org/4diac/en_ide.php.

[20] *Co je Home Assistant.* 2022. Dostupné z: https://tatageek.blog/2020/05/26/co-je-home-assistant/.

[21] *Google Assistant Now Has 500 Million Users, Rivaling Amazon Alexa.* New York: Insider Inc., 2022. Dostupné z: https://www.businessinsider.com/google-assistant-500-million-users-challenges-amazon-alexa-2020-1.

[22] *Node-RED.* San Francisco: OpenJS Foundation, 2022. Dostupné z: https://nodered.org/.

[23] *Node-RED vs. Home Assistant: Why not use both?* 2022. Dostupné z: https://home-assistant-guide.com/2020/09/27/node-red-vs-home-assistant-why-not-use-both/.

[24] *Co je PIR čidlo a jak funguje?* Praha: K&V ELEKTRO a.s., c1998-2022. Dostupné z: https://www.kvelektro.cz/blog/clanek/pir-cidlo.

[25] *Jak funguje kouřový požární hlásič.* Praha: [b.n.], c2007-2022. Dostupné z: https://www.zabezpecovaci-zarizeni.cz/pozarni-detektory/jak-funguje-kourovy-pozarni-hlasic-%5Bb062%5D.

[26] *Authentication Providers: Home Assistant.* C2014-2022. Dostupné z: https://www.home-assistant.io/docs/authentication/providers.

[27] *Kontakty.* Brno: Unipi technology, c2014-2022. Dostupné z: https://www.unipi.technology/cs/kontakty.

[28] *Unipi Neuron S103.* Brno: Unipi technology, c2014-2022. Dostupné z: https://www.unipi.technology/cs/unipi-neuron-s103-p93.

[29] *Unipi Patron S107.* Brno: Unipi technology, c2014-2022. Dostupné z: https://www.unipi.technology/cs/unipi-patron-s107-p381.

[30] Bezak, T. a Stremy, M. Usage of IEC Standards for Distributed Control Systems Modelling. *Applied Mechanics and Materials.* 2012, 220-223, s. 2672–2677. DOI: 10.4028/www.scientific.net/AMM.220-223.2672. ISSN 1662-7482. Dostupné z: https://www.scientific.net/AMM.220-223.2672.

[31] Cressler, C. *AMQP vs MQTT: Comparing Instant Messaging Protocols.* Denver: Cometchat, 2009. Dostupné z: https://www.cometchat.com/blog/amqp-vs-mqtt-comparing-instant-messaging-protocols.

[32] Gotkin, K. When Computers Were Amateur. *IEEE Annals of the History of Computing*. 2014, sv. 36, č. 2, s. 4–14. DOI: 10.1109/MAHC.2014.32. ISSN 1058-6180. Dostupné z: http://ieeexplore.ieee.org/document/6828557/.

[33] Hui, T. K., Sherratt, R. S. a Sánchez, D. D. Major requirements for building Smart Homes in Smart Cities based on Internet of Things technologies. *Future generation computer systems*. Elsevier B.V. 2017, sv. 76, s. 358–369. DOI: 10.1016/j.future.2016.10.026. ISSN 0167-739X. Dostupné z: databázeSciencedirect.

[34] Liška, R. *Vrstvy TCP/IP*. Praha: FJFI ČVUT, 2018. Dostupné z: http://kfe.fjfi.cvut.cz/~liska/unix/node18.html.

Obsah DVD

Seznam souborů a složek na přiloženém DVD:

- složka *text* – text práce v *pdf* a její zdrojové soubory ve formátu *tex*,
- složka *src* – zdrojové kódy a skripty aplikací,
- *README.txt* – popis instalace aplikace.