FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Studie proveditelnosti integrační kontrolní jednotky chytré domácnosti
Student:	Jan Matějka
Vedoucí:	Ing. Josef Pavlíček, Ph.D.
Studijní program:	Informatika
Studijní obor:	Informační systémy a management
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce letního semestru 2018/19

Pokyny pro vypracování

Cílem bakalářské práce je provést studii proveditelnosti implementace kontrolní řídící jednotky učené pro inteligentní domy na platformě Rapsberry pi3.

Studie musí obsahovat:
- analýzu problému,
- rešerši podobných řešení (např. Jablotron),
- návrh vlastní architektury řešení z pohledu HW - tj. navrhnout pro základní desku Rapsberry pi3, užití vhodného GMS modulu včetně vhodné antény, návrh propojení libovolního čidla s řídící deskou s pomocí rozhraní základní desky. Řešení umožní při zaslání logické 1 z analogových zařízení odelat SMS na definované tel. číslo a současně uložit správu o čase kdy k incidentu došlo, zprávě která byla zaslána, identifikátoru hlásícího zařízení. Tyto zprávy budou uloženy v databázovém systém a bude možno je v budoucnu využít s pomocí JEE aplikací,
- ekonomické porovnání s řešením např. Jablotron či jinou konkurencí po dohodě s vedoucím práce,
- zhodnocení závěrů vyplývajících ze studie proveditelnosti.

Technické zařízení dodá školitel.

Seznam odborné literatury

Dodá vedoucí práce.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
děkan

V Praze dne 10. ledna 2018

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Bakalářská práce

Studie proveditelnosti integrační kontrolní jednotky chytré domácnosti

Jan Matějka

Katedra softwarového inženýrství
Vedoucí práce: Ing. Josef Pavlíček, Ph.D.

14. května 2018

Poděkování

Tímto bych rád poděkoval panu Ing. Josefu Pavlíčkovi, Ph.D. za jeho rady a odborné vedení této práce. Dále bych rád poděkoval panu Ing. Petru Hanzlíkovi za oponenturu této práce. Samozřejmě bych také rád poděkoval mé rodině a přátelům za podporu a trpělivost při studiu.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 14. května 2018 .....

Odkaz na tuto práci

Matějka, Jan. *Studie proveditelnosti integrační kontrolní jednotky chytré domácnosti*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2018.

Abstrakt

Tato bakalářská práce se zabývá studií proveditelnosti integrované řídící jednotky chytré domácnosti, která řeší problém přijímání analogového signálu na sériovém portu a jeho zpracováním.

Cílem bakalářské práce je tedy analýza vestavěného systému chytré domácnosti na platformě Rasberry PI 3 a následný návrh a implementace zpracování analogového signálu na sériovém portu a odesílání SMS zpráv přímo ze zařízení. Tento systém je využitelný jako automatizovaný řídící systém bez nutnosti zásahu člověka, který umí detekovat neočekávané situace pomocí čidel. Na tyto situace bude integrovaná jednotka reagovat odesláním SMS zprávy na přednastavené telefonní číslo a bude mít předpřipravené rozhraní pro komunikaci s JEE aplikací prostřednictvím internetu.

Klíčová slova studie proveditelnosti, analýza vestavěného systému, návrh a implementace zpracování analogového signálu, zpracování signálu z externího čidla, odesílání SMS, chytrá domácnost, Python, Raspberry PI 3

Abstract

This bachelor thesis deals with smart home integration control unit feasibility study, what solves problem of receiving analog signal on the serial port and the processing.

The aim of bachelor thesis is analysis of embedded system smart home on the Rasberry PI 3 platform and follow-up design and implementation processing of analog signal on a serial port and sending SMS message direct from this device. This system is usable for automatization control system without need to manage a person, which can detect unexpected situation via sensors. For this situation integration unit will react by sending SMS message to set a phone number and will have at prepared interface for communication with JEE application via the Internet.

Keywords feasibility study, analysis of embedded system, design and implementation of processing analog signal, processing of signal from external sensor, sending SMS, smart home, Python, Raspberry PI 3

Obsah

Seznam obrázků

Seznam tabulek

Úvod

V dnešní době rychlého rozvoje nových technologií se hodí využít tyto technologie k všeobecnému prospěchu. Představte si, že vám vaše domácnost připomene, že jste něco zapomněli a vy tak předejdete velkým škodám na majetku. Zároveň může zachraňovat životy, pokud například při požáru automaticky spustí protipožární systém. To dává takovémuto řídícímu systému velký potenciál. Proto se v této práci věnuji řídící jednotce chytré domácnosti, která může být užitečnou součástí v každodenním užívání vašeho domova.

Cílem rešerše je tedy analýza vestavěného systému (dále jen zařízení, které v této práci používám) postaveného ze součástek zahrnující platformu Rapsberry PI 3, GSM anténu (vysílač a přijímač pracující na stejné frekvenci jako mobilní telefony) a čidlo zajišťující přenos dat z přírodních jevů (změna teploty, přímé sluneční záření apod.) na elektrický signál. Dále se v této práci zabývám srovnáním již hotových řešení, které na tuzemském trhu nabízí jen pár firem.

Cílem praktické části je navrhnout a implementovat integrovanou řídící jednotku chytré domácnosti, která bude zpracovávat požadavek, který přijal na sériovém portu. Dále je cílem porovnat produkty na českém trhu s již hotovými řešeními a celý postup a veškeré poznatky zpracovat ve studii proveditelnosti, která celý vývoj zdokumentuje.

Kapitola 1

Cíl práce

Cílem práce je analyzovat a navrhnout vhodnou integrační jednotku chytré domácnosti, která bude pokrývat detekci signálu z čidel a posílat informace prostřednictvím SMS zpráv a RESTful požadavků do informačního systému. Tento informační systém bude tyto požadavky zpracovávat a informace z nich ukládat ve formě využitelných dat. Data budou později využita k analýzám a predikci hrozícího nebezpečí.

1.1 Motivace

Toto téma mě zaujalo zejména proto, že nabízí široké uplatnění v praxi, může šetřit peníze nebo zachránit lidský život. Dále se mi na tématu líbilo, že jsem se s ničím srovnatelným na technologické úrovni dosud nesetkal a tak pro mě byla veliká výzva tento projekt dokončit.

1.2 Analýza

V analýze si kladu za cíl, aby byly jednoduše a srozumitelně popsané principy, na kterých tento projekt stojí a kdokoliv, kdo na něm bude chtít nadále pracovat, měl usnadněnou práci. Zejména v porovnání, které není vždy snadné a je nutné se v oblasti orientovat tak, aby byla zajištěna plná kompatibilita součástí. Kompatibilita je v tomto smyslu brána nejen jako schopnost něco fyzicky propojit, ale také schopnost zajistit komunikaci mezi rozhraními programovacích prvků.

1.3 Návrh a architektura

V návrhu jsem si kladl za cíl pokrýt řešením veškeré funkční a nefunkční požadavky kladené na tento projekt. V architektuře jsem cílil na dodržování zásad

návrhových a architektonických vzorů, zejména snadnou použitelnost, udržovatelnost a rozšířitelnost, jak z pohledu hardwaru, tak softwaru. Při návrhu jsem využíval znalostí z analýzy.

Kapitola 2

Analýza problému

2.1 Detekce signálu na sériovém portu

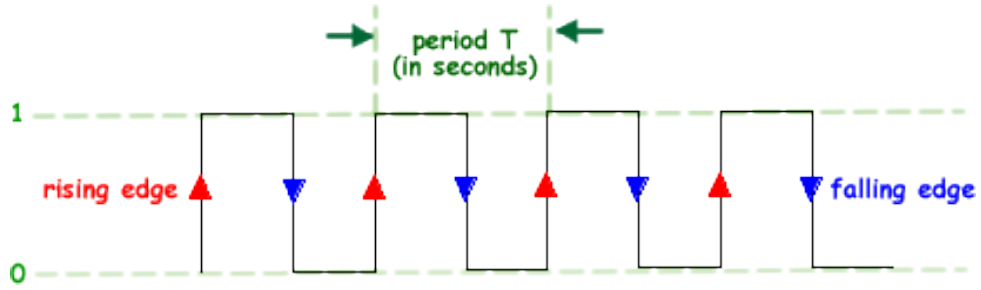
V úvodu je třeba definovat, jakým způsobem lze řešit detekci signálu na platformě Raspberry PI. Musíme si nejdříve alespoň v krátkosti vysvětlit, jak je signál reprezentován v elektrotechnické terminologii, abychom veškeré poznatky dokázali správně zasadit do souvislostí. O elektrickém signálu se učí již na základních školách a čtenář si o něm může důkladně přečíst v [7]. Nám bude stačit elektrický signál vnímat jako posloupnost elektrických nábojů (hodnot odlišných napětí), které produkuje elektronické zařízení a umožňuje tak přenášet informace po vodičích. Tento rozdíl pak vnímáme jako logickou 0 (nulu) a 1 (jedničku). Na základě toho pak vyhodnocujeme, jestli byl obvod propojen nebo nikoliv. Detekce se zaměřuje na již zmíněné změny, na takzvané hrany (anglicky edge).

2.1.1 Hrana

Hrana je změna z nulového napětí na provozní napětí přístroje a naopak v jednom časovém okamžiku (v řádu desítek milisekund) [8]. Snazší představu nám umožní obrázek 2.1, na kterém je zobrazen graf, kde osa x znázorňuje časovou osu a osa y znázorňuje napětí na vodiči. Pro úplnost jen doplňme, že graf je dvourozměrné zobrazení, kde je znázorněn vztah mezi veličinami na ose x a ose y. [9]

2.2 Zpracování signálu ze sériového portu

Většina elektronických zařízení, která nepoužívají žádný složitější procesor ke zpracování signálu, jsou spojená s metodami, které se musí využít k měření elektrického napětí. Toto měření zahrnuje použití tranzistorů a klopných obvodů viz. [8].



Obrázek 2.1: Graf znázorňující hrany. [1]

U Raspberry PI využívající operační systém Raspbian tato povinnost odpadá díky vestavěnému interpretu jazyka Python, rozšířeným o knihovny zajišťující tuto činnost. Pohodlné je to zejména proto, že už se nemusíme zabývat tím, co již bylo vyřešeno, čímž je měření přijatého elektrického signálu, ale můžeme se soustředit na vlastní práci a problémy spojené s ní. Tento způsob oddělení problémů nám dává možnost členit naši práci tak, aby se v ní programátor snadno orientoval.

Problém s detekcí elektrického signálu zahrnuje také paralelní programování. Podstata je v tom, že k zařízení může být připojeno více čidel a tak musí být zajištěna nezávislost mezi nimi. Toho docílíme tak, že využijeme takzvaná vlákna. Podle [10] je vlákno prostředek, který umožňuje spuštění nové instance části kódu, kdy toto vlákno se zpracovává zároveň vedle hlavního vlákna, které jej vytvořilo a zajišťuje paralelní zpracování programu v procesoru. Díky tomu docílíme toho, že řídící jednotka bude spolehlivě obsluhovat větší počet čidel. Bez paralelního zpracování by systém nemusel pracovat zcela správně, jelikož by docházelo k časově závislým chybám [11]. Časově závislá chyba je situace, kdy nastane nějaký jev a my jej nedokážeme ovlivnit, v jaký časový okamžik se vykoná. Tedy pokud neumožníme paralelní zpracování signálu, může se stát, že ve špatnou chvíli, kdy jsme přijali signál, bude zařízení tento signál ignorovat a tím pádem nic nezpracuje.

Jako vědecko-technologická, přesto realistická vize do budoucna by mohla být funkce, kdy zařízení detekuje neoprávněné vniknutí do budovy nebo vstup na pozemek. V tomto případě by mohla řídící jednotka aktivovat dron (letu schopné zařízení opatřeno 4 a více vrtulemi, které se samo řídí ve vzduchu [12]), který by byl opatřen záchytnými sítěmi a zloděje tak bez násilí pochytal. V současné době tato funkce není schopná provozu, jelikož naváděcí algoritmy (AI - Artificial Intelligence [13]) nejsou na tak vysoké technologické úrovni, aby spolehlivě tuto funkci zastávaly a také by tato funkce narážela v tuzemském právním systému, kde je stanoveno, že člověk nemůže podniknout takové kroky, které by vedly k neoprávněnému ublížení osob viz. [14].

2.3 Odesílání SMS zpráv

Princip odesílání SMS zpráv prostřednictvím mobilní sítě GSM vznikl v 80. letech minulého století a dodnes se dennodenně využívá k nejrůznějším účelům [15]. Primární účel byl zajistit komunikaci mezi lidmi, ale v průběhu času se komunikace dotáhla do fáze, kdy mezi sebou komunikují 2 přístroje, prostřednictvím zpráv si vyměňují data a na jejich základě vykonávají přednastavené instrukce.

V tomto projektu jsou SMS zprávy využívány takovým způsobem, že zprávu odesílá integrační jednotka a příjemcem je uživatel, který na základě této zprávy učiní patřičné kroky, aby zabránil blížícímu se nebezpečí. To má za následek zvýšení bezpečnosti a předcházení rozsáhlých škod na majetku a lidských životech.

SMS zpráva ponese důležité a stručné informace o tom, co se stalo, aby uživatel, který zprávu přijal, ihned věděl, jaké kroky má podniknout, aby nevznikly další problémy. Tvar SMS zprávy může být přibližně takovýto: „Chata Březany - dešťový senzor: Došlo k zaznamenání větší hladiny vody. Hrozí bezprostřední nebezpečí povodně!“. Na základě této informace je majitel chaty schopen uvědomit obyvatele Březan nebo integrované záchranné složky o tom, že hladina vody není v normálním stavu a schyluje se k zatopení blízkého okolí chaty.

2.4 Využitelná čidla

Elektronických čidel je na českém trhu velké množství. Jedním z prodejců je společnost GM Electronic s.r.o [16]. Ta nabízí sortiment od klasických spínačů přes čidla rovnováhy až po čidla detekující plyny. Jejich proškolený personál dokáže správně odhadnout požadavky a potřeby svých zákazníků a v případě nejasností poradit.

V tomto projektu jsem si v rámci základního testování funkčnosti vystačil s jednoduchým spínačem, jelikož je to univerzální a levný prostředek. Spínač slouží jako příprava pro skutečná čidla, například detekující hladinu vody v nádobě prostřednictvím plováku, který v případě zvednutí vodní hladiny sepne spínač a tím zařídí propojení obvodu. Podle mých informací proběhlo další testování u zákazníka na chromové vaně, která je opatřena přečerpávacím zařízením a na tomto zařízení bylo připojeno čidlo komunikující s řídící jednotkou. Zákazník na základě splněných akceptačních testů toto řešení akceptoval.

2.4.1 Klasické spínače

Klasické spínače jsou obvykle součástky z nevodivého obalu (nejčastěji plastu) a vodivého materiálu vnitřku, který se jednoduchým mechanismem při použití kinetické [17] nebo mechanické energie [18] pohne a tím vytvoří pohyb mechanismu, který vodivý materiál propojí. Příklad jednoduchého spínače je

Obrázek 2.2: Obrázek jednoduchého spínače. [2]

na obrázku 2.2. Jednoduché spínače v kombinaci s vhodnou konstrukcí lze využít například jako detekci zvýšené hladiny vody na vodních tocích a jejich okolí.

2.4.2 Plynová a kouřová čidla

Mezi nejběžnější plyny, se kterými se lze v praxi setkat, patří podle [19] sloučeniny typu:

- zemní plyn, označován také jako CNG, který je tvořen z methanu (CH_4), ethanu (C_2H_6), propanu (C_3H_8) a butanu (C_4H_{10}),
- zkapalněný ropný plyn, označován jako LPG, který je tvořen z propanu (C_3H_8) a butanu (C_4H_{10}) a
- kouř (spojení kapaliny a pevných látek rozptýlených v plynu).

Na trhu lze sehnat čidla, která detekují jednu z vybraných sloučenin nebo jejich kombinaci, což se vyplatí zvážit při uspokojování potřeb podle požadavků konkrétního zákazníka. K vytápění budov a jako palivo v domácnosti se nejčastěji využívá zkapalněný ropný plyn (LPG), tedy pro použití v domácnosti bude nejvhodnější. Čidlo zaznamenávající větší výskyt zkapalněného ropného plynu bude sloužit jako bezpečnostní systém, zamezující obyvatelům domu nadýchat se unikajícího plynu. Největší problém je v tom, že při větší koncentraci se člověk tohoto plynu nadýchá v nebezpečné míře dříve, než ho zpozoruje pomocí čichu.

Kapitola 3

Analýza technologií použitých při vývoji

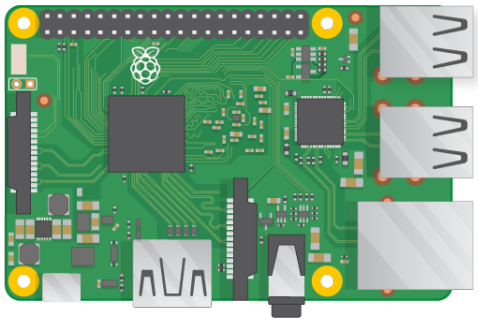
3.1 Platforma Raspberry PI 3

Mezi nejdůležitější pojmy patří platforma Raspberry PI 3, což je malý počítačový systém o rozměrech několika centimetrů, který v dnešní době disponuje takovým výpočetním výkonem, že bez sebemenších problémů dokáže efektivně řídit a ovládat celou domácnost. Veškerou specifikaci Raspberry PI naleznete na oficiálních stránkách komunity, která se vývojem Raspberry PI zabývá [20].

3.1.1 Hardware Raspberry PI 3

Z pohledu hardwaru Raspberry PI 3 nabízí:

- 1.2GHz 64-bit quad-core ARMv8 CPU (výkonný procesor),
- 802.11n Wireless LAN (bezdrátové připojení k internetu),
- Bluetooth 4.1 (bezdrátové připojení k ostatním zařízením na vzdálenost několika metrů),
- 4 USB porty,
- 40 GPIO piny (sběrnice sériového portu) - 28 z nich lze využít jako vstupně-výstupní zařízení,
- Full HDMI port (připojení k monitoru),
- Ethernet port (připojení k internetu pomocí kabelového vedení),
- Combined 3.5mm audio jack a composite video port (sdílený port pro vstupní a výstupní zvukový signál),



Obrázek 3.1: Obrázek s pohledem na Raspberry PI 3 ze shora. [3]

- Camera interface (CSI) (sběrnice pro připojení kamery),
- Display interface (DSI) (sběrnice pro připojení malé obrazovky, která může být i dotyková) a
- Micro SD card slot (slot pro paměťovou kartu typu Micro SD).[3]

Z těchto parametrů je patrné, že za cenu, za kterou se Raspberry PI 3 prodává a rozměry, kterými disponuje, nabízí slušný výkon pro provoz více úloh najednou v chytré domácnosti i průmyslu. Cena samostatného produktu Raspberry PI 3 je uvedena v kapitole Finanční analýza v sekci Kalkulace ceny pořizovacích nákladů na vlastní integrační jednotku. Raspberry PI 3 je znázorněn na obrázku 3.1.

3.1.2 Software Raspberry PI 3

Z pohledu softwaru Raspberry PI 3 nenabízí od prodejce žádný software, kromě ovladačů jednotlivých součástek na desce, které jsou v čipech nahrané již z výroby. Software, pomocí kterého lze Raspberry PI 3 ovládat, je libovolný operační systém s nízkými požadavky na výkon hardwaru. Typicky to bývají Unix-based systémy založené na jádru operačního systému Unix. V nabídce jsou však i další distribuce nebo operační systém od firmy Microsoft. Raspberry PI Foundation UK doporučuje využít těchto operačních systémů:

- Raspbian
- Ubuntu Mate,
- Snappy Ubuntu Core,

- Windows 10 IoT Core,
- OSMC,
- Libreelec,
- Pinet,
- Risc OS,
- Weather Station a
- Ichigojam RPi.

Více o operačních systémech se lze dočíst v [21].

Obraz s operačním systémem (soubor s aplikací operačního systému) se nahraje na libovolnou Micro SD s dostatečnou paměťovou kapacitou, kterou si uživatel zjistí při rozhodování, jaký operační systém na Raspberry PI 3 nasadí a k instalaci může použít návod na oficiálních stránkách Raspberry PI Foundation [22]. Dále je možné využít volně dostupný nástroj **etcher.io**, který paměťovou kartu zformátuje a nahraje na ni vybraný obraz operačního systému. [23]

V tomto projektu je využíván operační systém Rasbian Lite, což je operační systém založený na distribuci Debian a Lite verze znamená, že je bez grafického rozhraní pouze jako terminálový klient, což pro potřeby tohoto projektu je zcela dostatečné. Navíc má tato distribuce v sobě již implementované ovladače pro ovládání GPIO sběrnice sériového portu, takže není třeba řešit další problémy spojené s kompatibilitou rozhraní.

3.2 Programovací jazyk Python

Při vývoji byl využit jazyk Python verze 3.0, jelikož je aktuálně ve stabilní verzi a je zajištěna podpora externích knihoven zmíněných dále v této kapitole. [24]

Jazyk Python je beztypový programovací jazyk, který díky svým vlastnostem umožňuje snadné naučení se principům tohoto jazyka a zkracuje čas na vývoj aplikací. Tento programovací jazyk disponuje principy objektově-orientovaného programování (OOP) a umožňuje použití softwarových návrhových vzorů, které z objektově-orientovaného programování vycházejí. Více informací k objektově-orientovaným principům v programování, pokud je čtenář nezná, se dozví v [25], aby mu problematika zapadala do kontextu.

K implementační části práce, která zahrnuje zajistit naslouchání na sériovém portu a vygenerování SMS zprávy byl použit programovací jazyk Python, který poskytuje použití všech využitelných konstruktů programovacího jazyka, které k této práci potřebuji. Zejména třídu **Serial**, která se nachází v knihovně **serial**. Třída **Serial** ve své implementaci obsahuje metody:

- **__init__()**, konstruktor třídy Serial, který nastaví základní konfiguraci zařízení sériového portu.
- **read()**, která zajišťuje naslouchání na sériovém portu, podle nastavené konfigurace. Metoda vrací přijatá data, která zařízení sériového portu získala.
- **write()**, která zajišťuje odeslání dat po sériovém portu, podle nastavené konfigurace. Metoda vrací počet dat, které se podařilo odeslat po sériovém portu.
- **open()**, která umožňuje otevřít spojení se zařízením připojeným k sériovému portu.
- **close()**, která umožňuje zavřít spojení se zařízením připojeným k sériovému portu. Vhodné použití je, když aplikace využívá zařízení ve více částech programu a je třeba zamezit vícenásobnému použití z důvodu zahlcení požadavků na zařízení.
- a další. [26]

Třída **Serial** se v tomto projektu využívá ke komunikaci s GSM modulem, který je připojen k sériovému portu nebo USB portu. Na typu připojení modulu nezáleží, jelikož z pohledu rozhraní třídy Serial se se zařízením pracuje stejně.

3.2.1 Knihovna RPi.GPIO

Knihovna RPi.GPIO je open-source (open-source licence umožňuje libovolné využívání, změnu a šíření uměleckého díla viz. [27]) knihovnou, která umožňuje přímou interakci s GPIO rozhraním na platformě Raspberry PI. Rozhraní této knihovny obsahuje několik funkcí, které se starají o přímé ovládání GPIO sběrnice. Jsou to funkce:

- **setmode(mode)**, která nastaví režim rozložení pinů podle používaného typu. *mode* může být buď *BCM* nebo *BOARD*. Vysvětlení zkratek je popsáno v této kapitole v sekci Platforma Raspberry PI 3.
- **setup(channel, type)**, která nastaví číslo pinu *channel* na vstupní nebo výstupní zařízení podle typu *type*. *type* může být buď *IN* nebo *OUT*.
- **output(channel, state)**, která pro pin, nastavený jako výstupní zařízení, nastaví výstupní napětí buď logickou 1 nebo 0. Pokud je metoda volána nad vstupním pinem, je vyhozena Výjimka (ošetření chybového stavu, který zabrání havarování programu [28]). *state* může být buď *LOW* nebo *HIGH*.

- **input(channel)**, která v okamžiku volání zjistí logickou hodnotu naměřenou na pinu a vrací ji jako návratovou hodnotu. *channel* je číslo pinu a návratová hodnota je typu *int* (Integer [29]).

- **add_event_detect(channel, mode, callback, bounce_time)**, která nastaví na pinu *channel* událost *callback*, která se zavolá, podle nastaveného modu *mode*. Tato událost slouží k vyvolání funkce, která je předána jako parametr *callback*, pokud dojde k detekci hrany. Typ hrany je parametr *mode* a může být typu *RISING*, *FALLING* nebo *BOTH* podle typu hrany, kterou chceme detekovat. *callback* parametr je reference na libovolnou funkci nebo metodu třídy se správnou strukturou. Tato funkce musí mít jeden parametr, kterému se předává číslo pinu *channel*. *bounce_time* je hodnota času, po kterou má událost čekat, než vyvolá *callback*. Tento parametr je však nepovinný a lze tuto prodlevu, která bude ignorovat falešné zákmity na vodiči, řešit jiným způsobem. Pod jiným způsobem si čtenář může představit využití třídy *Timer* z knihovny **threading** popisované v následující podsekci.

- **cleanup()**, která zruší veškerá předchozí nastavení. Je vhodné jí využít z důvodu bezchybného ukončení aplikace.[30]

3.2.2 Knihovna threading

Knihovna threading je vestavěnou knihovnou jazyka Python a umožňuje paralelní programování aplikací. O paralelním programování jsme si řekli v kapitole Analýza problému v sekci Zpracování signálu ze sériového portu, takže jej nebudu dále v tomto odstavci rozebírat na teoretické úrovni.

Knihovna threading poskytuje rozhraní obsahující mnoho tříd a funkcí. V tomto projektu se zdá být využitelná třída **threading.Thread**, která ze svých potomků vytvoří paralelizovatelné objekty a třída **threading.Timer**, která umožňuje časovou prodlevu na nově vytvořeném vlákně, než spustí funkci, která je předána jako jeden z parametrů.

Více o knihovně si čtenář může přečíst v [31].

3.2.2.1 Thread

Třída Thread reprezentuje vlákno, které se při zavolání konstruktoru vytvoří a zpracovává se paralelně vedle hlavního vlákna.

V tomto projektu je tato třída využívána jako předek a slouží k rozšiřování vlastností svých potomků podle objektově-orientovaného návrhu.

Vedle konstruktoru je ještě nezbytné volat metodu *start()*, která spustí metodu *run()*, která má ve svém těle definované chování, které se spouští paralelně v novém vlákně.

3.2.2.2 Timer

Třída Timer reprezentuje časovač, který se spustí na novém vlákně a po uplynutí nastaveného času prodlevy spustí funkci, která je předána jako jeden z parametrů konstruktoru.

3.2.3 Knihovna requests

Knihovna requests je externí knihovnou od Kennetha Reitze, která vznikla pod open-source licencí *GPL License* a umožňuje snadné vytváření HTTP požadavků pro komunikaci s webovými službami [32].

Rozhraní této knihovny obsahuje třídu *requests*, která nabízí k využití metody:

- **get(url)**, která reprezentuje GET metodu využívající URL adresu k přenášení dat,
- **post(url, data)**, která reprezentuje POST metodu využívající tělo požadavku k přenášení dat,
- **put(url)**, která reprezentuje PUT metodu pracující stejně jako GET, jen se liší v typu požadavku,
- **delete(url)**, která reprezentuje DELETE metodu pracující stejně jako GET a PUT, opět lišící se jen v typu požadavku.

Návratovou hodnotou těchto metod je objekt typu **Response**, který obsahuje informace, které získal jako odpověď na zaslaný požadavek. Informace, které lze získat jsou například:

- hlavička odpovědi,
- číselná hodnota odpovědi,
- data v textovém formátu,
- data v JSON formátu a další.

3.3 GSM moduly na trhu

Dalším pojmem je GSM modul (Global System for Mobile Communications), což je elektrické zařízení, které přijímá a vytváří rádiové vlny na frekvenci 900 a 1800 MHz [15], zajišťující komunikaci v bezdrátové síti. Průměrný čtenář se s touto technologií setkává dennodenně, pokud používá mobilní telefon. Zařízení je tedy velice univerzální a snadno použitelné. GSM modul je buď připevněn přímo na desku Rapsberry PI 3 a propojen prostřednictvím sériového portu nebo jej lze připojit prostřednictvím USB portu.

V této práci používám GSM modul k odesílání SMS zpráv na přednastavené telefonní číslo, což slouží jako zdroj informace k mimořádné situaci. SMS zpráva je zkratkou (Short Message Service), což je krátká zpráva o délce max. 160 znaků bez použití diakritiky. [15]

Komunikace s GSM modulem probíhá skrze takzvané AT příkazy (Attention commands), které se posílají pomocí rozhraní třídy **Serial** knihovny *serial*. AT příkazy mají tvar: „AT_command=value“, kde AT_command je typ daného příkazu a value je hodnota, která se vztahuje k tomuto příkazu. Prostřednictvím těchto příkazů probíhá veškerá konfigurace nastavení i ovládání, jako je přijímání a odesílání SMS zpráv nebo zprostředkování a vytáčení telefonických hovorů. [33]

Co se týče kompatibility zařízení s Raspberry PI, tak většina výrobců garantuje získání ovladačů pro bezproblémovou komunikaci. Zejména u GPIO socket modulů může nastat problém s nekompatibilitou ovladačů a s podporou ze strany operačního systému Raspbian. Je nutné dát pozor na to, jak výrobce garantuje podporu s patřičnou verzí jádra operačního systému, aby se uživatel vyhnul pozdějším problémům.

3.3.1 USB moduly

Obrovskou výhodou USB modulů je, že GPIO sběrnici můžeme plně využít k připojení až 28 čidel (počet vychází ze složení pinů, které lze využít jako vstupní zařízení, zbytek slouží jako zdroj napětí nebo k programování EEPROM paměti) a detekovat tak mnohem více situací než s moduly připevněnými k GPIO sběrnici. EEPROM je nad rámec tohoto tématu, takže si jej nebudeme popisovat. Nevýhodou ovšem je, že USB modul není tak pevně připevněn k základní desce Raspberry PI, takže hrozí, že při špatné manipulaci nebo upevnění se může modul odpojit, čímž pozbude účinnosti.

USB moduly se od sebe vlastnostmi (zejména rozměry) příliš neliší, takže zájemce při pořizování zohlední nejspíše jen cenu. Odlišnosti se týkají jen v použitých komponentách přímo v zařízení, což může ovlivnit spolehlivost a životnost produktu.

3.3.1.1 Huawei E3531 3G Modem

Tento USB modul od čínského výrobce Huawei je lehký a tenký. Z výroby je opatřen nejnovějšími ovladači, které zajistí kompatibilitu s nejpoužívanějšími operačními systémy. Modul je aktuálně k sehnání na [4] a je zobrazen na obrázku 3.2.

3.3.1.2 D-Link DWM-157 3G Modem

Tento USB modul je kompaktní s rozměry srovnatelnými s Huawei E3531 3G Modem. Vyrábí jej celosvětově rozšířená společnost D-Link s centrálou v Taiwanu zaměřující se na výrobu síťových prvků pro počítačové sítě. Taktéž

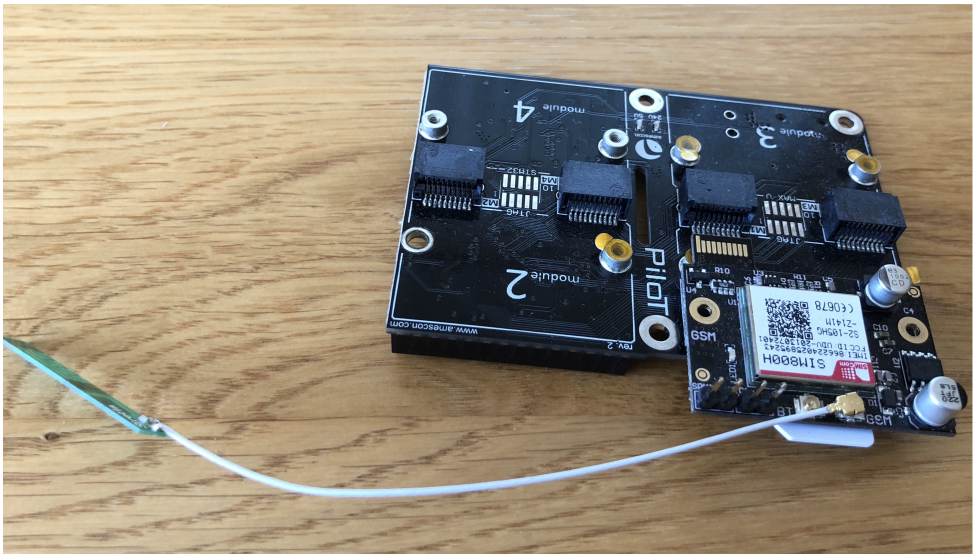
Obrázek 3.2: Obrázek Huawei E3531 3G Modemu. [4]

Obrázek 3.3: Obrázek D-Link DWM-157 3G Modemu. [5]

má zajištěnu podporu ze strany výrobce. Modul je aktuálně k sehnání na [5] a je zobrazen na obrázku 3.3.

3.3.2 GPIO socket moduly

Jak již bylo řečeno, moduly připevněné k GPIO sběrnici zabírají více GPIO pinů, tedy je omezen počet připojených čidel. Navíc většina modulů přidá na velikosti základní desky Raspberry PI, což přináší problémy s výběrem a koupí správné velikosti krytu, aby byla jednotka chráněná proti prachu a nečistotám. U USB modulu problém s krytem nenastane, jelikož všechny kryty na trhu mají z výroby předpřipravené výřezy pro všechny konektory.



Obrázek 3.4: Obrázek Amescon PiIoT rozšiřující desky a 2G GSM modulu.

3.3.2.1 Amescon PiIoT Module

PiIoT modul dříve označovaný jako RasPiComm+ vyrábí rakouský výrobce Amescon Software GmbH. Tento modul se dělí na:

- rozšiřující desku (PiIoT main board)
- samotné komunikační zařízení - například PiIoT GSM modul

Rozšiřující deska obsahuje 4 sloty k připevnění GSM, GPS a dalších modulů, které lze využít k nejrůznějším potřebám. Při zapojení rozšiřující desky zbude volných 8 vstupně-výstupních pinů, což může značně omezovat využití. Je to právě z důvodu možnosti zapojit více komunikačních zařízení do rozšiřující desky. [34]

3.3.2.2 Itead RPi SIM800 GSM/GPRS Module

Tento modul je produktem čínské společnosti ITEAD Intelligent Systems Co., Ltd. a jeho velkou výhodou je, že přestože k ovládání využívá GPIO sběrnici, tak umožňuje na tento modul připojit další zařízení a plně tuto sběrnici využít. V případě výběru tohoto zařízení jsme schopni připojit až 15 čidel, což je více než u Amescon PiIoT Modulu. V porovnání s USB moduly je to výrazně méně, ale na druhou stranu nehrozí samovolné odpojení od základní desky Raspberry PI, což je slabinou USB modulů. Modul je tedy vhodnou alternativou pro průmyslové využití tam, kde je třeba zajistit, aby byl GSM modul pevně připevněn k základní desce integrační řídící jednotky a při tom

Obrázek 3.5: Obrázek Itead RPi SIM800 GSM/GPRS Modulu. [6]

řídící jednotka dokázala obsluhovat větší množství senzorů. Modul je zobrazen na obrázku 3.5. [6]

Kapitola 4

Analýza současných řešení na trhu a konkurence

V současné době se vývojem zařízení pro chytrou domácnost zabývá jen pár tuzemských firem. Například společnosti Jablotron Group a.s. (dále jen Jablotron) a Selax Electronics s.r.o. (dále jen Selax Electronics). Produktům těchto společností se budeme podrobně věnovat v této kapitole.

Bohužel z důvodu malé konkurence na trhu si zařízení dlouhodobě drží cenu a mnohdy jsou velice nákladnou záležitostí pro průměrné zákazníky. V tom vidím obrovskou příležitost přiblížit trh i těmto zákazníkům a pomocí vlastního produktu si postavit stabilní místo na trhu.

V budoucnosti by mohlo toto zařízení detekovat i neoprávněné vniknutí osob na soukromý pozemek a volit na to adekvátní reakci. Představme si, že by mohl vylétnout dron ze střechy, analyzovat počet pachatelů a poslat na policejní ústřednu informaci o celkové situaci. Případně by mohl dron pochytat pachatele prostřednictvím záchytných sítí, aby jim při zásahu nebylo ublíženo na zdraví. Toto řešení je veskrze vizí a zahrnuje spoustu nevyřešených mezních situací a právních problémů. Zejména, kdybyste tímto způsobem chytali pachatele, mohl by na vás podat trestní oznámení za to, že jste jej zadržel proti jeho svobodné vůli a celý incident by se tak obrátil proti vám. K tomu by se hodila mírná změna v legislativě, která by majitele takovéhoto systému chránila.

4.1 Produkt společnosti Jablotron

Produkty společnosti Jablotron Group a.s. nabízí kompletní řešení, které zahrnuje řídící jednotku opatřenou GSM vysílačem (zařízení odesílající SMS zprávy prostřednictvím GSM sítě), napojenou na webovou a mobilní aplikaci, kde má uživatel přehled o tom, zda nevznikl požár nebo nevzniklo zatopení vodou. Webová aplikace je program, běžící na serveru (výkonném počítači)

provozovatele tohoto systému a je přístupná prostřednictvím webového prohlížeče fakticky odkudkoliv, kde je funkční internetové připojení. Mobilní aplikace je program běžící na mobilních zařízení, tedy je přizpůsobená na menší přístroje, s čímž je spojen upravený vzhled, aby i na těchto zařízeních byly veškeré informace čitelné. Rovněž přistupuje na server, kde přijímá data a vypisuje je uživateli.

Dále tato aplikace nabízí ovládání na dálku, kdy si uživatel může zapnout topení v budově a přijede tak do již vytápěných prostor. Další funkcí je aktivace ohřevu vody v nádobě na ohřev užitkové vody o objemu několika desítek až stovek litrů. V situaci, kdy zákazník vlastní chatu nebo chalupu na venkově a má malé děti či vnoučata, může být toto řešení velice příjemné a uvítají jej i příbuzní. [35]

V kapitole Finanční analýza navazuji na kalkulaci ceny za pořízení tohoto řešení pro koncového zákazníka.

4.2 Produkt společnosti Selax Electronics

Produkty společnosti Selax Electronics s.r.o nenabízí tak komplexní služby jako společnost Jablotron. Z tohoto důvodu se také odvíjí konečná cena, která je pro konečné zákazníky příznivější. Společnost Selax Electronics se zabývá primárně GSM hlásiči a alarmy, takže nenabízí žádné propojení s informačním systémem, ke kterému lze přistupovat z webové nebo mobilní aplikace. Selax Electronics nabízí pouze zpoplatněnou mobilní aplikaci pro chytré telefony s operačním systémem Android, která přímo prostřednictvím SMS ovládá a nastavuje GSM hlásič. Hlásič tak nevyžaduje připojení k internetu. Za mobilní aplikaci se platí jednorázový poplatek při pořizování. [36]

V kapitole Finanční analýza navazuji na kalkulaci ceny za pořízení tohoto řešení pro koncového zákazníka.

Kapitola 5

Finanční analýza

V této kapitole si rozebereme finanční stránku tohoto projektu. Čtenář tak bude mít přehled o celkové kalkulaci současných řešení na trhu a o nákladech, potřebných k sestavení vlastního produktu. Výstupem bude srovnání, zda má smysl se vývojem tohoto produktu zabývat nebo raději využít již stávajících řešení, která jsou k dostání do několika pracovních dnů (zpravidla do týdne). Nutno podotknout, že největším problémem tohoto projektu nejsou finance nebo technologická omezení, ale čas na něm strávený.

5.1 Ceny aktuálních srovnatelných produktů na trhu

Jelikož nelze volně dohledat přesné tabulkové ceny, za které lze produkt pořídit včetně montáže a údržby, byl jsem nucen prostřednictvím nezávazné poptávky skrze e-mailovou komunikaci získat cenovou kalkulaci pořizovacích nákladů na srovnatelné produkty, které nabízejí společnosti Jablotron a Selax Electronics. U společnosti Jablotron se o dodání, montáž a podporu starají smluvní partneři. Společnost Selax Electronics si dodání, montáž i podporu zajišťuje sama nebo k tomu využívá taktéž smluvních partnerů. Zákazník tak má možnost využít dodavatele ze svého blízkého okolí.

5.1.1 Řešení společnosti Jablotron Group a.s.

Celkové náklady na pořízení produktu od společnosti Jablotron Group a.s., kde dodání zajistí smluvní partner **PCO Service s.r.o** [37] činí **13 262 Kč bez DPH**, po případě **PV elektronické systémy - Pavel Váverka** [38] činí **12 945 Kč bez DPH**. Jednotlivé položky jsou popsány pro:

- PCO Service s.r.o v tabulce 5.1
- PV elektronické systémy - Pavel Váverka v tabulce 5.2

1x JA 101K- ústředna	7 710 Kč
1x záložní akumulátor 2,6 AH	360 Kč
1x klávesnice JA 114E - LCD	1 817 Kč
1x JA 110P detektor pohybu - sběrnice	496 Kč
1x JA 110ST požární detektor - sběrnice	836 Kč
1x GS 133 - detektor úniku plynu	881 Kč
1x JA 111H - modul pro připojení detektoru plynu	315 Kč
1x JA110A siréna vnitřní	487 Kč
1x JA 110F - sběrnicový detektor zaplavení	360 Kč
Celkem	**13 262 Kč**

Tabulka 5.1: Tabulka kalkulace ceny řešení od společnosti Jablotron Group a.s. dodané společností PCO Service s.r.o.

1x Univerzální GSM komunikátor a ovladač	3 490 Kč
1x Zálohovací modul pro GD-04K	546 Kč
1x Detektor hořlavých plynů	881 Kč
1x Kombinovaný detektor kouře a teplot se sirénou	728 Kč
1x Detektor zaplavení s jednou sondou pro vodivé neagresivní kapaliny	2 649 Kč
1x Zálohovaný napájecí zdroj v boxu	2 100 Kč
1x Zálohovací akumulátor 12V/18Ah	1 051 Kč
1x Předpokládaný drobný materiál pro propojení GSM modulu a ostatních komponentů	1 500 Kč
Celkem	**12 945 Kč**

Tabulka 5.2: Tabulka kalkulace ceny řešení od společnosti Jablotron Group a.s. dodané společností PV elektronické systémy - Pavel Váverka.

V této kalkulaci nejsou započteny náklady na montáž ani zprovoznění systému.

5.1.2 Řešení společnosti Selax Electronics s.r.o.

Celkové náklady na pořízení produktu od společnosti **Selax Electronics s.r.o.** činí **8 904 Kč včetně DPH**. Jednotlivé položky jsou popsány v tabulce 5.3. V této kalkulaci není započteno mechanické uchycení snímače hladiny vody ani montáž včetně oživení systému.

1x GSM drátová i bezdrátová ústředna	4 990 Kč
1x kouřový detektor	484 Kč
1x detektor úniku plynu	1 440 Kč
1x snímač hladiny s napojením na elektroniku	1 490 Kč
2x bezdrátová klíčenka	500 Kč
Celkem	**8 904 Kč**

Tabulka 5.3: Tabulka kalkulace ceny řešení od společnosti Selax Electronics.

5.2 Kalkulace ceny pořizovacích nákladů na vlastní integrační jednotku

V této sekci jsou podrobně uvedeny náklady na vývoj vlastní integrační jednotky chytré domácnosti. Tyto náklady zahrnují pořizovací náklady na hardware a vyhotovení softwaru. Náklady na údržbu, podporu a součinnost zákazníka není nutné započítávat, jelikož se nejedná o klasický výpočet ceny pro zákazníka ve studii proveditelnosti, ale interní vývoj prototypu řídící jednotky uvnitř organizace.

Celkové náklady se dělí na fixní a variabilní. Fixními náklady budou náklady na software, jelikož jednou vytvořená aplikace bude univerzální aplikací, která se bude nahrávat na všechny řídící jednotky, které z pohledu hardwaru lze doobjednávat podle potřeby později. Variabilní náklady budou náklady na hardware, jelikož si budeme moci určit cenu, podle předpokládaného prodeje produktů v předem určeném časovém období.

5.2.1 Náklady na hardware

Při pořizování hardwaru, který budeme potřebovat k vývoji, musíme zohlednit následující nezbytné komponenty:

- základní desku Raspberry PI 3,
- GSM modul,
- anténa k GSM modulu,
- napájení pro Raspberry PI a
- příslušenství (kabelové vedení, senzory atd.).

Podle cen uvedených v [34] je podrobná kalkulace **včetně DPH** rozepsána v tabulce 5.4. Cena příslušenství je odhadnutá a zahrnuje samostatná čidla, kabelové vedení a součástky potřebné na konstrukci mechanických a ochranných částí čidel. V případě, že bychom místo PiIoT rozšiřující desky a GSM modulu využili Itead RPi SIM800 GSM/GPRS Module, který s kurzem k dnešnímu dni stojí přibližně **800 Kč včetně DPH**, snížily by se celkové náklady

základní desky Raspberry PI	813 Kč,
PiIoT rozšiřující deska	1 398 Kč
PiIoT GSM modul	1 140 Kč,
anténa k GSM modulu	124 Kč
napájení pro Raspberry PI	233 Kč
příslušenství	cca. 1 000 Kč
Celkem	**4 708 Kč**

Tabulka 5.4: Tabulka výpočtu nákladů na hardware.

na **2 846 Kč včetně DPH**. Pokud bychom nebyli vázáni na GSM moduly připevněné na GPIO sběrnici a využili USB modul Huawei E3531 3G Modem s cenou **490 Kč včetně DPH**, činily by celkové náklady **2 536 Kč včetně DPH**.

5.2.2 Náklady na software

Při výpočtu nákladů na software musíme zohlednit pracnost, kterou bude vývoj aplikace, která poběží na Raspberry PI, stát. Tyto náklady vypočítáme tak, že vezmeme dobu, kterou programátor na projektu strávil a vynásobíme ji denními náklady na tohoto programátora.

Programování softwarové implementace tohoto projektu zabralo **15 pracovních dní**. S hodinovou sazbou **500 Kč** na mzdu programátora vychází náklady na jeden pracovní den **4 000 Kč**. V součtu to dává **60 000 Kč** na vývoj softwaru.

5.3 Určení prodejní ceny

Aby se prodej vlastního produktu vyplatil, musí být příjmy z prodeje vyšší než vynaložené náklady. Tato podmínka bude splněna v případě, že se podaří prodat průměrně alespoň **1 ks/měsíc** v průběhu jednoho roku, kdy počítáme s výnosem jednoho produktu **5 100 Kč** a náklady **2 536 Kč** (levnější varianta s USB modulem) nebo s výnosem **7 500 Kč** a náklady **4 708 Kč** (varianta se současně využívaným hardwarem). Výpočet vychází ze vzorce pro výpočet zisku, což jsou výnosy a od nich odečten součet fixních a variabilních nákladů. Z našeho odhadu vychází zisk **64 Kč/ks** (levnější varianta) nebo **292 Kč/ks** (varianta se současně využívaným hardwarem).

Pokud si bude společnost moci dovolit zvýšit riziko a zvednout cenu produktu, pak se i zvýší zisk z jednoho prodaného kusu.

Kapitola 6

Návrh a architektura hardwaru

6.1 Raspberry PI 3

Jako řešení problému se nabízí využít nejnovější verzi platformy Raspberry PI, kterou je Raspberry PI 3 a osadit ji vhodným GSM modulem připevněným přímo k základní desce na sériový port. Pro práci jsem si vybral Itead Raspberry Pi GSM Board, který splňuje veškeré nároky na použití v GSM síti a je dobře cenově dostupný. Navíc pracuje na frekvenci 800 MHz, což je pro toto řešení optimální.

6.2 Operační systém Raspbian Lite

Operačním systémem použitým v tomto projektu je Raspbian Lite. Je to odlehčená verze operačního systému Raspbian bez grafického rozhraní, které nabízí ovládání pouze prostřednictvím příkazového řádku. Navíc je přímo přizpůsobený pro spuštění na Raspberry PI. Největší výhodou absence grafického rozhraní jsou nižší nároky na výkon, což znamená, že lze využít výkon například ke zpracování informací.

Operační systém Raspbian vychází z Unix-based systémů, přesněji řečeno vychází z distribuce Debian. V tomto projektu jsem z důvodu kompatibility s PiIoT rozšiřující deskou využil distribuci s verzí Kernelu (jádra operačního systému [39]) 4.4.

6.3 PiIoT GSM modul

PiIoT GSM modul je produktem rakouské společnosti Amescon Software GmbH a k připojení k základní desce Raspberry PI je nutná rozšiřující deska PiIoT main board (dříve RasPiComm+ main board) opatřena 4 sloty pro umístění jednotlivých modulů, což značí, že umožňuje používání více komunikačních

modulů najednou. Bez ohledu na to, zda to bude více stejných nebo různých modulů.

GSM modul pro naše použití však stačí pouze jeden. Jeden z toho důvodu, že pokud budeme odesílat více SMS zpráv najednou v jeden časový okamžik, jsme tuto činnost schopni takzvaně serializovat díky vlastnostem jazyka Python tak, že GSM modul bude SMS zprávy odesílat postupně. Díky tomu se vyhneme problémům s řízením vláken a ušetříme peníze za další GSM modul.

PiIoT GSM modul jsme pro tento projekt zvolili proto, že zákazník nejprve projevil zájem i o aplikaci GPS modulu, ale po několika jednáních tento požadavek zhodnotil jako nepotřebný. V rámci výzkumu a vývoje jsme tento modul využili i pro případ, že zákazník své požadavky opět změní.

Kapitola 7

Návrh a architektura softwaru

7.1 Business proces model

Business proces model zahrnuje 4 základní procesy, které pokrývají nejdůležitější vlastnosti aplikace. Základní pohled na procesy je znázorněn na obrázku 7.1.

Aplikaci je nutné ovládat vstupem od uživatele, dále musí umět správně nastavit aplikaci, aby správně vykonávala svou činnost a také musí zpracovávat signál a generovat upozornění, pokud detekuje validní signál ze sériového portu. Aplikace běží po celou dobu spuštění, dokud není ukončena.

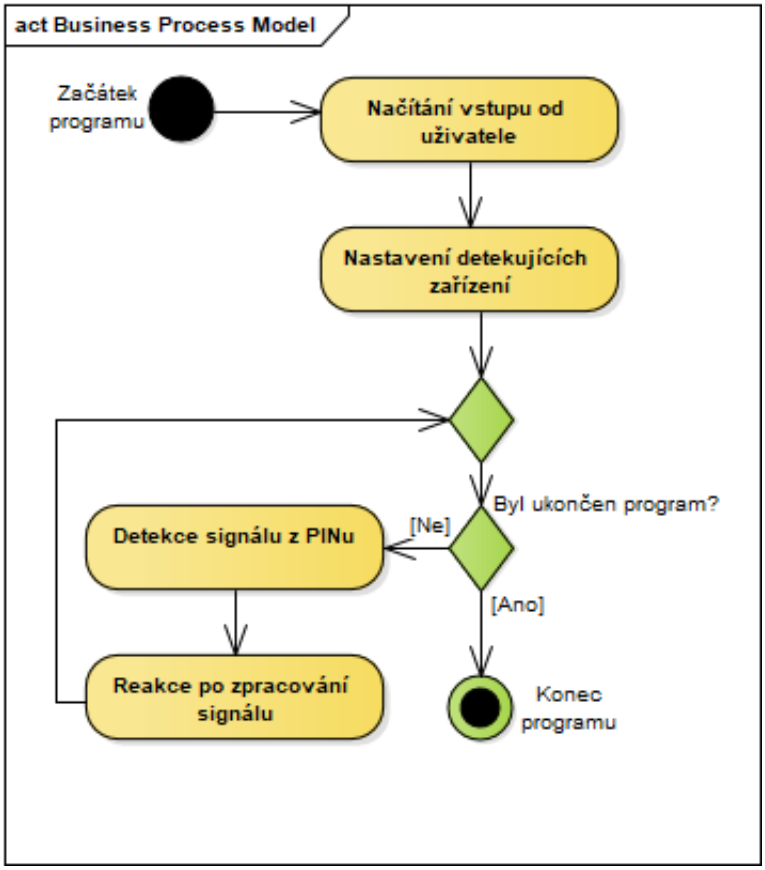
Základními procesy tedy jsou **načítání vstupu od uživatele**, **nastavení detekujících zařízení**, **detekce signálu z PINu** a **reakce po zpracování signálu**. Každému z těchto procesů se dále věnuji podrobněji v této kapitole.

7.1.1 Načítání vstupu od uživatele

Tento proces je znázorněn na obrázku 7.2 a zahrnuje **nastavení GSM modulu** podle parametru zadaného při spuštění a další načítání parametrů od uživatele. Uživatel nejprve zadá **informace o senzoru**, a pokud jsou tyto informace validní, přejde k **načítání informací o oznamovacích zařízeních**, kterých může být pro každý senzor více. Zároveň se zde provádí kontrola na duplicity, tedy je zajištěno, že jedno unikátní oznamovací zařízení se v aplikaci vyskytuje pouze jednou. Načítání parametrů probíhá v cyklu, dokud není ukončen vstup od uživatele.

7.1.2 Nastavení detekujících zařízení

Tento proces je znázorněn na obrázku 7.3 a zahrnuje **Nastavení zařízení** a **nastavení handlerů**. Nastavení zařízení zajistí správnou inicializaci nastavení připojených senzorů podle dat, která uživatel zadal v předchozím procesu

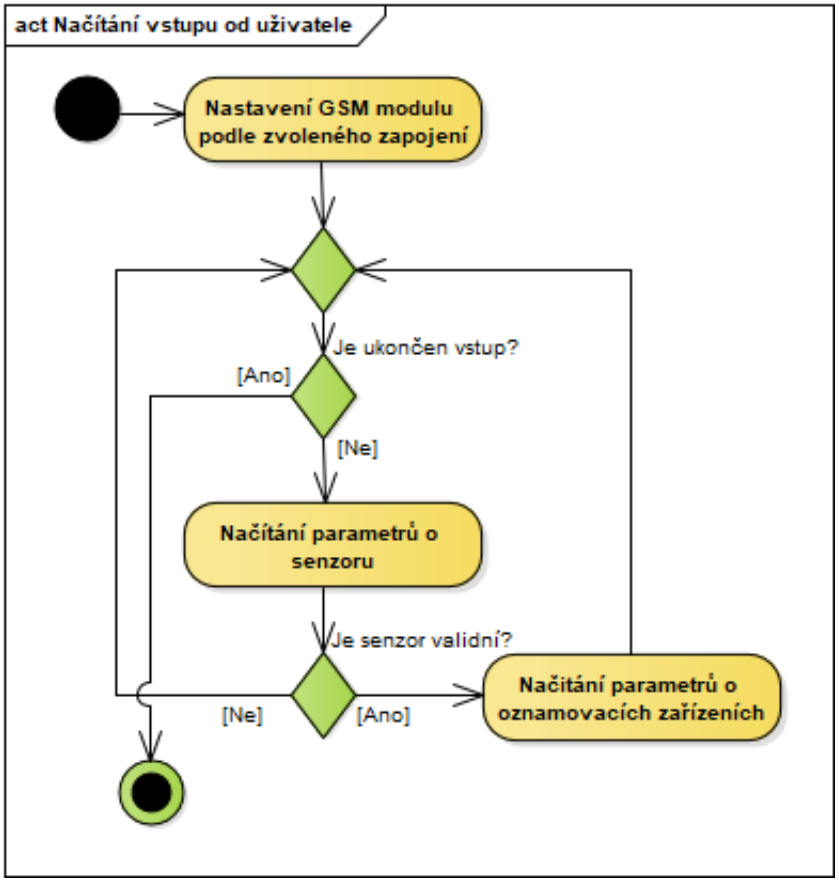


Obrázek 7.1: Business proces model projektu Detection.

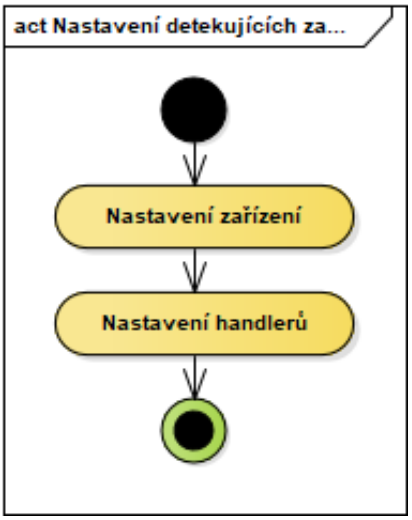
(Načítání informací o senzoru). Nastavení handlerů se postará o přidání události na pozadí aplikace, což zajistí neustálé a nezávislé čtení signálu na PINu paralelně s ostatním zpracováním v aplikaci.

7.1.3 Detekce signálu z PINu

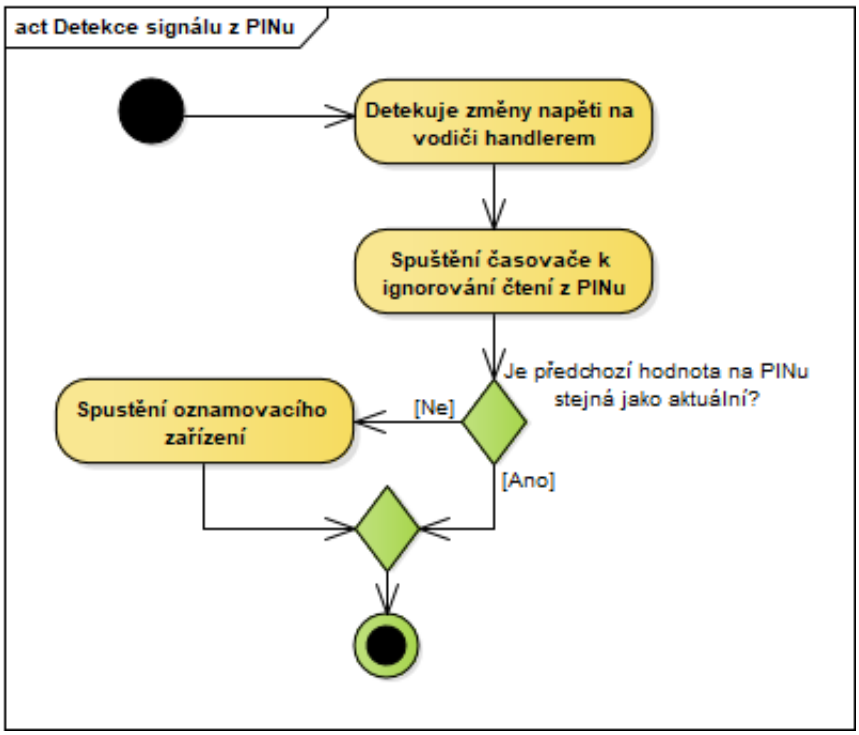
Tento proces je znázorněn na obrázku 7.4 a zahrnuje **detekci změny napětí na vodiči prostřednictvím handleru**, **spuštění časovače k ignorování čtení z PINu** a **spuštění oznamovacích zařízení**. V první fázi, kdy handler detekuje změnu napětí na PINu, odstartuje další fázi, ve které se spustí časovač, který zajistí, že u signálu ze sériového portu budou ignorovány falešné impulzy, které na vodiči vzniknou změnou odporu při propojení obvodu. Dále se provede další čtení z PINu, které v případě změny napětí aktivují spuštění oznamovacích zařízení nebo budou signál (falešné impulzy) ignorovat. Samotná oznamovací zařízení jsou přiřazena k senzorům a díky paralelizaci aplikace je třeba zajistit jejich serializaci prostřednictvím zámků, které



Obrázek 7.2: Business proces načítání vstupu od uživatele.



Obrázek 7.3: Business proces nastavení detekujících zařízení.



Obrázek 7.4: Business proces detekce signálu z PINu.

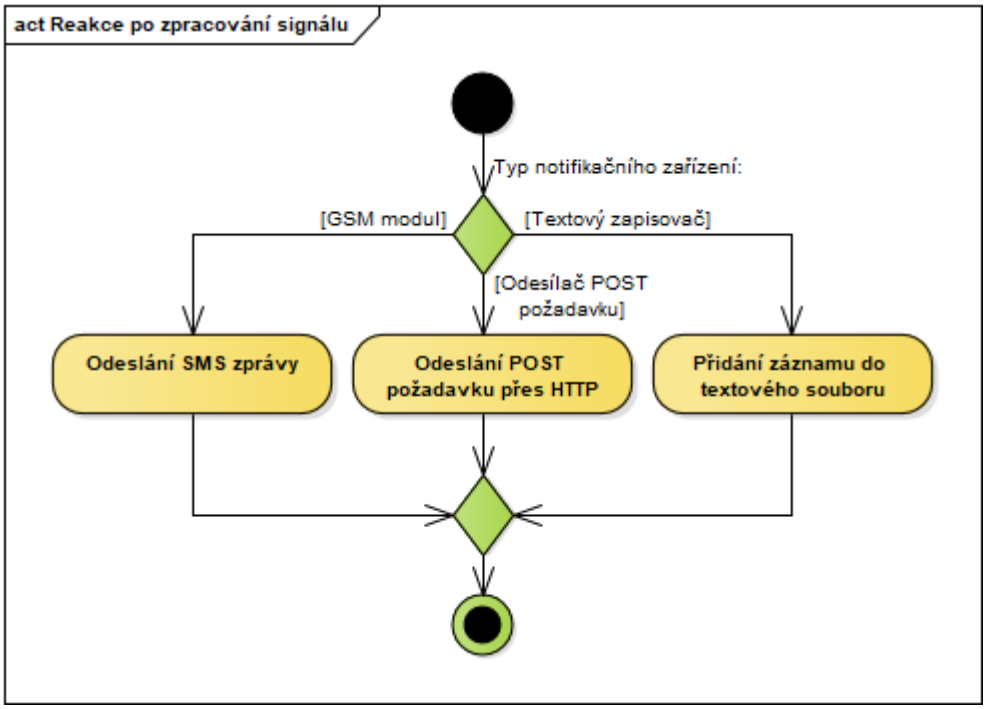
umožňují zablokovat přístup do části programu více vláknům v jeden časový okamžik.

7.1.4 Reakce po zpracování signálu

Tento proces je znázorněn na obrázku 7.5 a zahrnuje 3 procesy, které vykonávají oznamovací zařízení podle jejich specifikovaného typu. Jsou jimi **odeslání SMS zprávy**, **odeslání POST pořadavku přes HTTP** připojujícího se na **RESTful API** webové služby a **přidání záznamu do textového souboru**.

7.2 Model tříd

Model tříd zahrnuje logickou strukturu aplikace podle principů a zásad objektově orientovaného návrhu. Projekt Detection je v základní hierarchii rozčleněn do 7 balíčků, což zvýší přehlednost a oddělí od sebe třídy, které spolu přímo logikou nesouvisí. Je tomu tak proto, že v případě rozšiřování aplikace dalším definováním tříd, by se mohl stát návrh nepřehledným, čemuž se tímto oddělením snažíme předejít. Tento model je znázorněn na obrázku 7.6.



Obrázek 7.5: Business proces reakce po zpracování signálu.

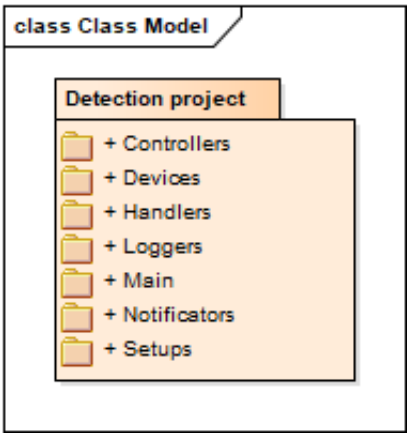
Balíček **Controllers** obsahuje třídy, které se starají o získávání vstupu od uživatele, balíček **Devices** obsahuje třídy reprezentující senzory, kolekci, která tyto senzory shromažďuje a třídy reprezentující oznamovací zařízení, která nejsou přímo vestavěná na platformě Raspberry PI. Balíček **Handlers** obsahuje třídy, které inicializují událostní handlery v aplikaci, balíček **Loggers** obsahuje třídy zajišťující zaznamenání chybových hlášek při neočekávaných situacích, které mohou za běhu aplikace nastat, balíček **Main** zahrnuje počátek aplikace, balíček **Notificators** obsahuje třídy reprezentující oznamovací zařízení a balíček **Setups** zahrnuje třídy, které se starají o správné nastavení před spuštěním aplikace.

7.2.1 Controllers

Tento balíček slouží k oddělení třídy Input, která načítá data od uživatele. Struktura balíčku je znázorněna na obrázku 7.7 a vlastnosti třídy popsané v tabulce 7.1.

7.2.2 Devices

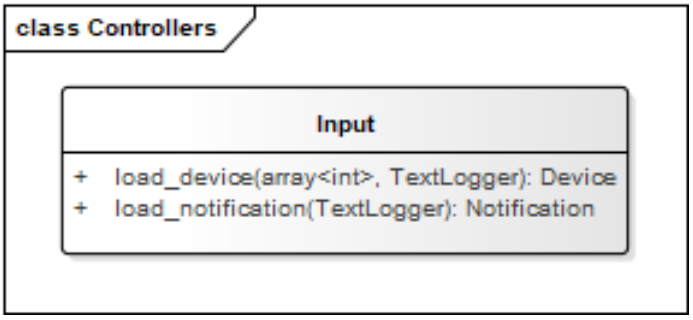
Tento balíček slouží k oddělení tříd Device, DevicePool a GSMModule. Třída Device reprezentuje senzor, díky kterému aplikace sbírá data a adekvátně



Obrázek 7.6: Model tříd projektu Detection.

Metody	
load_device	Metoda čte data ze standartního vstupu od uživatele a provádí validaci, zda je vstup ve správném formátu a odpovídá senzoru.
load_notification	Metoda čte data ze standartního vstupu od uživatele a provádí validaci, zda je vstup ve správném formátu a odpovídá oznamovacímu zařízení.

Tabulka 7.1: Tabulka vlastností třídy Input.



Obrázek 7.7: Model tříd v balíčku Controllers.

Atributy	
name	Atribut reprezentuje jméno senzoru, podle kterého se bude rozpoznávat u jakého senzoru došlo k sepnutí.
pin_number	Atribut reprezentuje číslo pinu, ke kterému je senzor připojený.
notifications	Atribut reprezentuje kolekci shromažďující notifikační zařízení, která budou aktivována v případě detekce signálu ze senzoru.
properties	Atribut reprezentuje pole informací, které bude využíváno pro sběr informací ve webové službě.
logger	Atribut reprezentuje instanci TextLoggeru a slouží k přidání chybové hlášky do souboru s chybami v případě neočekávané situace.
lock	Atribut reprezentuje instanci threading.Lock, což je zámek zajišťující serializaci v paralelní aplikaci.
Metody	
constructor	Metoda inicializuje novou instanci třídy Device a nastaví podle parametrů vlastní atributy.
run	Metoda slouží ke spouštění oznamovacích zařízení.
___eq___	Metoda slouží k porovnávání dvou objektů typu Device. Dva objekty jsou stejné v případě, že mají stejné jméno nebo číslo pinu.

Tabulka 7.2: Tabulka vlastností třídy Device.

na ně reaguje, třída DevicePool je kontejner, který shromažďuje senzory a třída GSMModule zajišťuje komunikaci s GSM modulem, tedy zpracovává požadavek na odeslání SMS zprávy. Navíc třída GSMModule, díky svým atributům, umožňuje univerzálně využívat GSM moduly s různým typem připojení k Raspberry PI. Struktura balíčku je znázorněna na obrázku 7.8 a vlastnosti třídy Device popsané v tabulce 7.2, DevicePool v tabulce 7.3 a GSMModule v tabulce 7.4.

7.2.3 Handlers

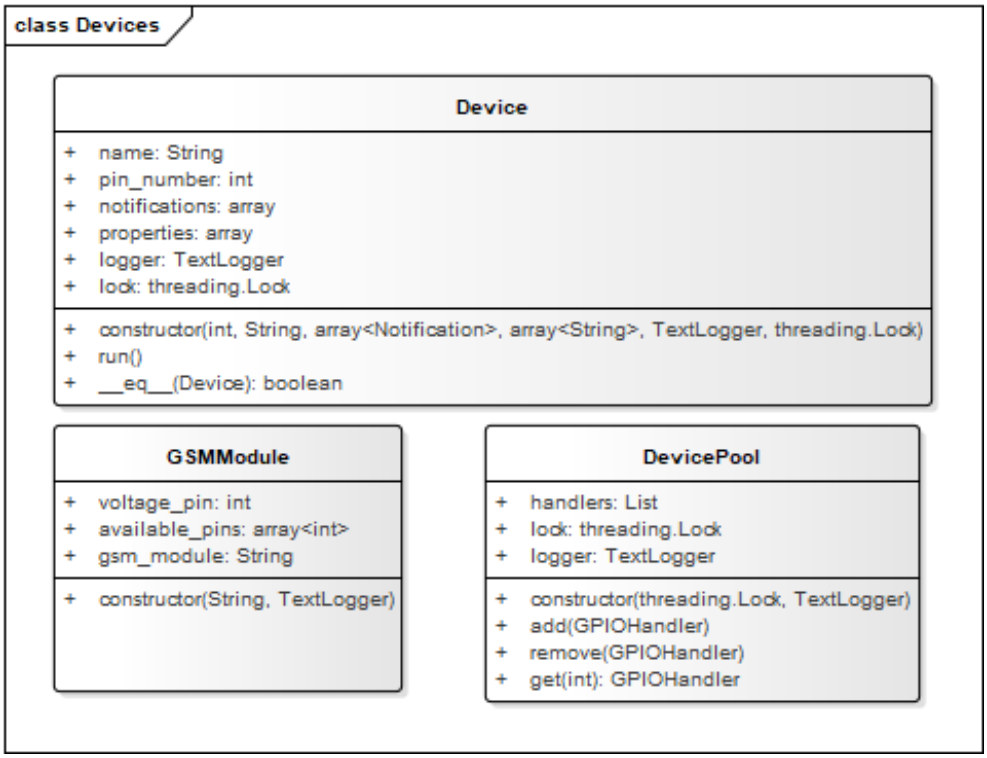
Tento balíček slouží k oddělení třídy GPIOHandler od ostatních. Třída GPIOHandler reprezentuje handler, díky kterému aplikace paralelně na pozadí naslouchá na zadaném pinu podle vlastností senzoru. Z důvodu paralelního běhu třída dědí od třídy threading.Thread ze standartní knihovny jazyka Python. Struktura balíčku je znázorněna na obrázku 7.9 a vlastnosti třídy GPIOHandler popsané v tabulce 7.5.

Atributy	
handlers	Atribut reprezentuje kolekci shromažďující instance GPIOHandlerů, které zajišťují detekci signálu na sériovém portu na pozadí aplikace.
logger	Atribut reprezentuje instanci TextLoggeru a slouží k přidání chybové hlášky do souboru s chybami v případě neočekávané situace.
lock	Atribut reprezentuje instanci threading.Lock, což je zámek zajišťující serializaci v paralelní aplikaci.
Metody	
constructor	Metoda inicializuje novou instanci třídy DevicePool a nastaví podle parametrů vlastní atributy.
add	Metoda slouží k přidání nového GPIOHandleru do kolekce handlers.
remove	Metoda slouží k odebírání zvoleného GPIOHandleru podle parametru z kolekce handlers.
get	Metoda slouží k vrácení GPIOHandleru z kolekce handlers podle parametru, kterým je číslo pinu.

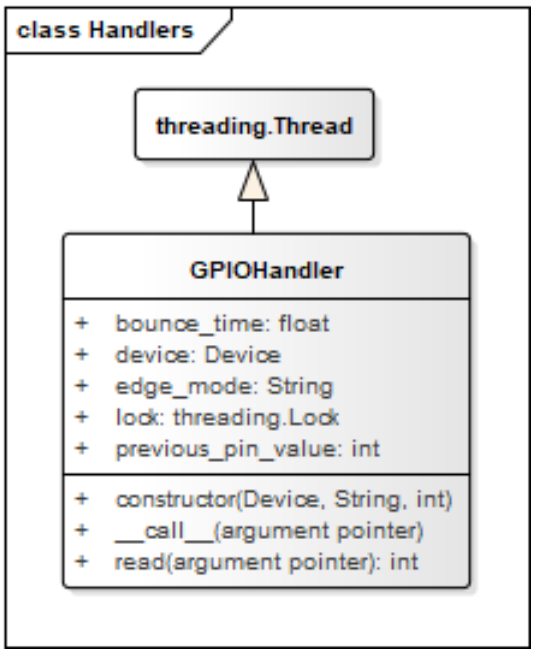
Tabulka 7.3: Tabulka vlastností třídy DevicePool.

Atributy	
voltage_pin	Atribut reprezentuje číslo pinu, který bude sloužit jako zdroj napětí v obvodu se senzory.
available_pins	Atribut reprezentuje pole čísel pinů, které lze využít ke snímání signálu z čidel. Atribut se využívá k odlišení, zda byl použit GSM modul připevněný k sériovému portu nebo prostřednictvím USB portu.
gsm_module	Atribut reprezentuje název zařízení používaný k identifikaci v operačním systému Raspbian. Tento atribut umožňuje univerzálně využívat GSM moduly s různými typy připojení.
Metody	
constructor	Metoda inicializuje novou instanci třídy GSMModule a nastaví podle parametru vlastní atribut gsm_module.

Tabulka 7.4: Tabulka vlastností třídy GSMModule.



Obrázek 7.8: Model tříd v balíčku Devices.



Obrázek 7.9: Model tříd v balíčku Handlers.

Atributy	
bounce_time	Atribut reprezentuje hodnotu v milisekundách, po kterou bude vlákno v metodě ___call___ čekat a tím ignorovat falešné impulzy na pinu, které by způsobily sepnutí senzoru.
device	Atribut reprezentuje instanci třídy Device, která nese vlastnosti senzoru.
edge_mode	Atribut reprezentuje textový řetězec s typem změny napětí (tzv. hrany). Možnosti jsou *rising* (zvýšení napětí), *falling* (zmenšení napětí) nebo *both* (obě předchozí).
lock	Atribut reprezentuje instanci threading.Lock, což je zámek zajišťující serializaci v paralelní aplikaci.
previous_pin_value	Atribut reprezentuje hodnotu napětí naměřenou na pinu před časovou prodlevou. Využívá se ke správné detekci signálu a k ignorování falešných impulzů na vodiči.
Metody	
constructor	Metoda inicializuje novou instanci třídy GPIOHandler a nastaví podle parametrů vlastní atributy.
___call___	Metoda se předává jako parametr při definování handleru v rozhraní RPi.GPIO knihovny. Zavolá se, když handler detekuje signál.
read	Metoda se postará o kontrolu, zda nedošlo k přijmutí falešného signálu, při čemž využívá hodnotu bounce_time k časové prodlevě a je vyvolána v metodě ___call___.

Tabulka 7.5: Tabulka vlastností třídy GPIOHandler.

7.2.4 Loggers

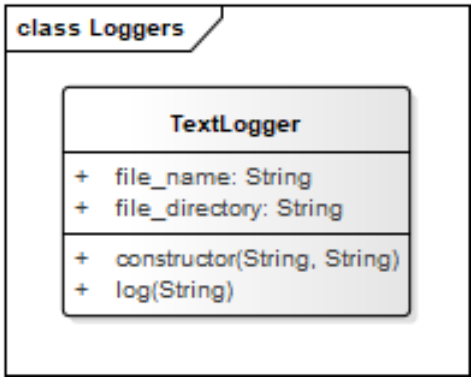
Tento balíček slouží k oddělení třídy TextLogger od ostatních. Třída TextLogger reprezentuje textový zapisovač chybových hlášek do souboru s daty o neočekávaných situacích. Struktura balíčku je znázorněna na obrázku 7.10 a vlastnosti třídy TextLogger popsané v tabulce 7.6.

7.2.5 Main

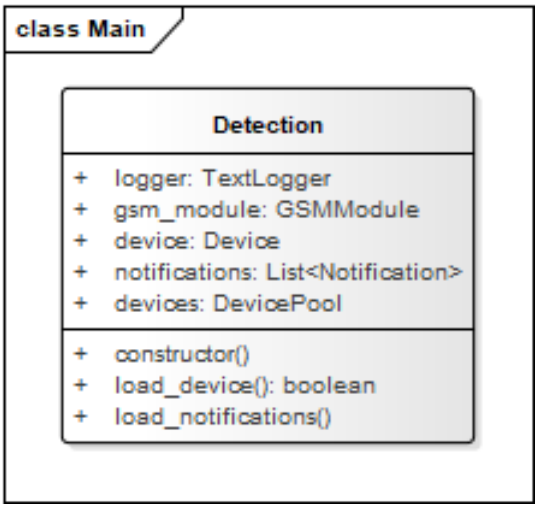
Tento balíček slouží k oddělení třídy Detection od ostatních. Třída Detection reprezentuje hlavní část aplikace. Struktura balíčku je znázorněna na obrázku 7.11 a vlastnosti třídy TextLogger popsané v tabulce 7.7.

Atributy	
file_name	Atribut reprezentuje jméno souboru, do kterého bude třída zapisovat chybové hlášky.
file_directory	Atribut reprezentuje jméno adresáře, ve kterém se nachází soubor se jménem v předchozím atributu.
Metody	
constructor	Metoda inicializuje novou instanci třídy TextLogger a nastaví podle parametrů vlastní atributy.
log	Metoda se postará o zapsání řádku s chybovou hláškou do souboru s chybami podle parametrů této třídy.

Tabulka 7.6: Tabulka vlastností třídy TextLogger.



Obrázek 7.10: Model tříd v balíčku Loggers.



Obrázek 7.11: Model tříd v balíčku Main.

Atributy	
logger	Atribut reprezentuje instanci TextLoggeru a slouží k přidání chybové hlášky do souboru s chybami v případě neočekávané situace.
gsm_module	Atribut reprezentuje instanci třídy GSMModule, prostřednictvím kterého se budou odesílat SMS zprávy s upozorněním.
device	Atribut reprezentuje instanci třídy Device a slouží k uložení informací o posledním zadaném senzoru uživatelem.
notifications	Atribut reprezentuje kolekci shromažďující veškeré oznamovací zařízení použité v aplikaci. Slouží k zajištění, aby v aplikaci nebyly duplicitní oznamovací zařízení.
devices	Atribut reprezentuje instanci DevicePool a slouží jako kontejner na všechny senzory používané v aplikaci. Zároveň slouží k zajištění, aby v aplikaci nebyly duplicitní senzory.
Metody	
constructor	Metoda inicializuje novou instanci třídy Detection a nastaví vlastní atributy na výchozí hodnoty.
load_device	Metoda se postará o načtení validních informací o senzoru.
load_notifications	Metoda se postará o načtení validních informací o oznamovacích zařízeních přiřazených k senzoru.

Tabulka 7.7: Tabulka vlastností třídy Detection.

7.2.6 Notifications

Tento balíček slouží k oddělení tříd Notification a jejich potomků PostRequest, SMSSender a TextInserter od ostatních. Třída Notification reprezentuje oznamovací zařízení, která jsou přiřazena k senzorům a v případě detekce signálu ze sériového portu aktivují svou činnost prostřednictvím metody *run*. Třída Notification má dále své potomky rozdělené podle typu oznámení, které produkují. Jsou jimi:

- PostRequest - třída, která reprezentuje POST požadavek pro komunikaci s RESTful API,
- SMSSender - třída, která reprezentuje odesílač SMS zpráv prostřednictvím GSM sítě a
- TextInserter - třída, která reprezentuje zapisovač dat do textového souboru.

Metody	
run	Metoda je abstraktní a její chování je definování v potomcích této třídy. Metoda slouží jako šablona pro univerzální využití oznamovací činnosti a aplikaci polymorfismu.
___eq___	Metoda je abstraktní a její chování je definování v potomcích této třídy. Metoda slouží jako šablona pro porovnávací operátor a aplikaci polymorfismu.

Tabulka 7.8: Tabulka vlastností třídy Notification.

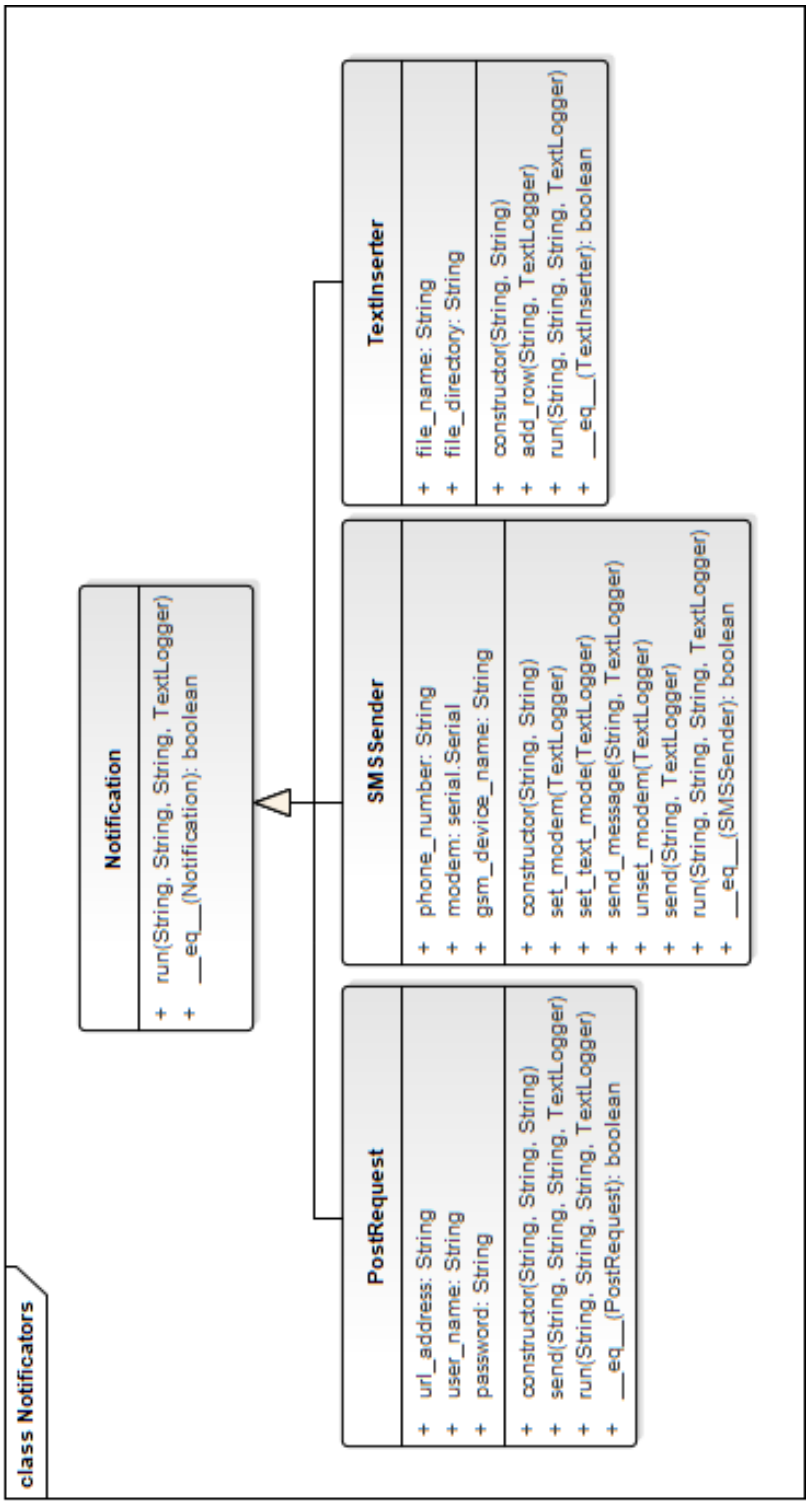
Atributy	
url_address	Atribut reprezentuje URL adresu webové služby, na kterou bude třída odesílat POST požadavky.
user_name	Atribut reprezentuje uživatelské jméno k ověření, zda je odesílatel požadavku oprávněn zasílat informace do informačního systému.
password	Atribut reprezentuje heslo párované k uživatelskému jménu a taktéž slouží k ověření oprávnění vkládat informace do informačního systému.
Metody	
constructor	Metoda inicializuje novou instanci třídy PostRequest a nastaví vlastní atributy.
send	Metoda je volána díky polymorfismu z metody *run* a zajistí správné odeslání POST požadavku na server.
run	Metoda slouží jako šablona pro aplikaci polymorfismu. V metodě je volána metoda *send*.
___eq___	Metoda slouží k porovnání dvou objektů typu PostRequest, zda jsou stejné. Objekty jsou stejné, pokud mají stejnou URL adresu, uživatelské jméno i heslo.

Tabulka 7.9: Tabulka vlastností třídy PostRequest.

Struktura balíčku je znázorněna na obrázku 7.12 a vlastnosti třídy Notification popsané v tabulce 7.8, PostRequest v tabulce 7.9, SMSSender v tabulce 7.10 a TextInserter v tabulce 7.11.

7.2.7 Setups

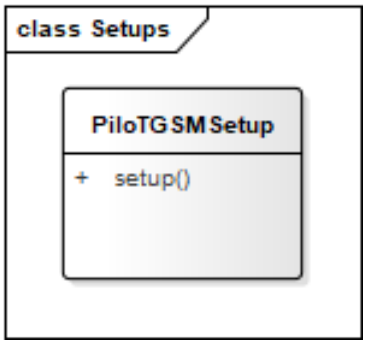
Tento balíček slouží k oddělení třídy PiIoTGSMSetup od ostatních. Třída PiIoTGSMSetup prostřednictvím metody *setup* aktivuje PiIoT GSM modul takovým způsobem, že modul bude schopen přijímat příkazy k manipulaci s tímto modulem. Struktura balíčku je znázorněna na obrázku 7.13 a vlastnosti třídy PiIoTGSMSetup popsané v tabulce 7.12.



Obrázek 7.12: Model tříd v balíčku Notificators.

Atributy	
phone_number	Atribut reprezentuje telefonní číslo příjemce, na které budou odeslané upozorňující SMS zprávy.
modem	Atribut reprezentuje instanci třídy serial.Serial, což je objekt, který přímo komunikuje se zařízením připojeným na sériový port.
gsm_device_name	Atribut reprezentuje jméno zařízení, které používá operační systém k identifikaci.
Metody	
constructor	Metoda inicializuje novou instanci třídy SMSSender a nastaví vlastní atributy.
set_modem	Metoda nastaví atributu *modem* instanci třídy *serial.Serial* a tím zajistí připojení k GSM modulu.
set_text_mode	Metoda nastaví textový režim, přes který se bude GSM modul ovládat.
send_message	Metoda pošle data SMS zprávy přímo do datové sběrnice GSM modulu.
unset_modem	Metoda ukončí spojení s GSM modulem, aby případně nebránilo ostatním aplikacím a součástem operačního systému se zařízením komunikovat.
send	Metoda je volána díky polymorfismu z metody *run* a zajistí správné odeslání SMS zprávy prostřednictvím GSM sítě.
run	Metoda slouží jako šablona pro aplikaci polymorfismu. V metodě je volána metoda *send*.
___eq___	Metoda slouží k porovnání dvou objektů typu SMSSender, zda jsou stejné. Objekty jsou stejné, pokud mají stejné telefonní číslo.

Tabulka 7.10: Tabulka vlastností třídy SMSSender.



Obrázek 7.13: Model tříd v balíčku Setups.

Atributy	
file_name	Atribut reprezentuje název souboru, do kterého se budou data o upozornění přidávat.
file_directory	Atribut reprezentuje jméno adresáře, ve kterém se nachází soubor se jménem předchozího atributu.
Metody	
constructor	Metoda inicializuje novou instanci třídy TextInserter a případně nastaví vlastní atributy.
add_row	Metoda je volána díky polymorfismu z metody *run* a zajistí správné přidání dat do textového souboru.
run	Metoda slouží jako šablona pro aplikaci polymorfismu. V metodě je volána metoda *add_row*.
___eq___	Metoda slouží k porovnání dvou objektů typu PostRequest, zda jsou stejné. Objekty jsou stejné, pokud mají stejný název souboru i adresáře.

Tabulka 7.11: Tabulka vlastností třídy TextInserter.

Metody	
setup	Metoda zajistí správné nastavení PiIoT modulu, aby se modul aktivoval a byl připraven k přijímání ovládacích příkazů.

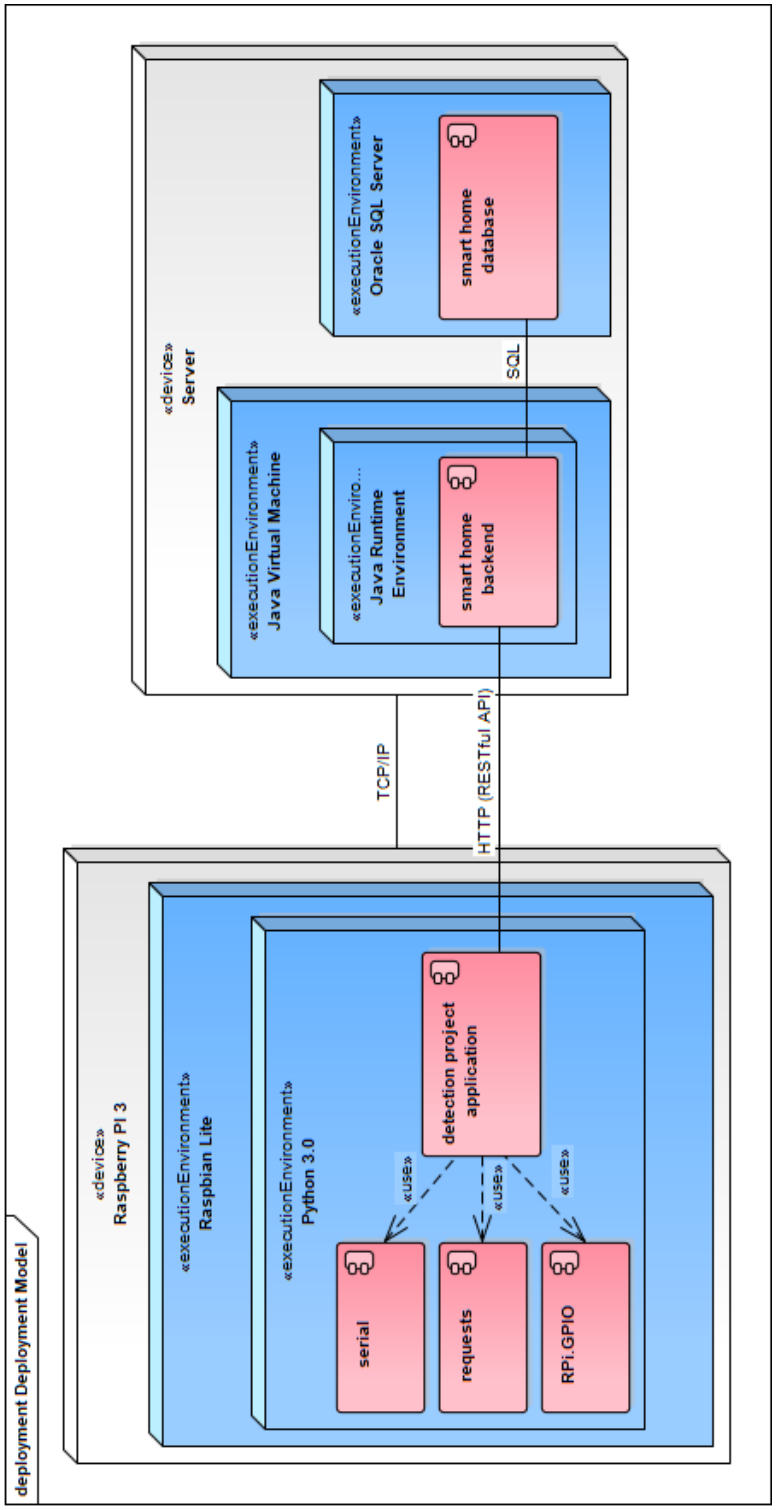
Tabulka 7.12: Tabulka vlastností třídy PiIoTGSMSetup.

7.3 Diagram nasazení

Diagram nasazení, znázorněn na obrázku 7.14, reprezentuje pohled na strukturu běhových prostředí, ve které aplikace běží a na které je aplikace přímo závislá. Diagram obsahuje 2 základní zařízení: **Raspberry PI**, na kterém běží aplikace Detection a **Server**, na kterém běží webová služba.

Na zařízení **Rasberry PI** je nahrán operační systém Raspbian Lite, který se ovládá pouze prostřednictvím terminálu, takže zařízení zbytečně nezatěžuje grafické prostředí a výkon procesoru se dá využít efektivněji na výpočty a zpracování. V operačním systému Raspbian Lite je již vestavěný interpret jazyka Python 3.0, avšak je dobré před prvním spuštění provést update na aktuální verzi interpretu a zajistit tak stabilní chod aplikace. Aplikace dále využívá knihovny třetích stran, které jsou popsané v následující sekci.

Na **serveru** je v tomto případě použito prostředí **Java Virtual Machine**, ve kterém je spuštěno prostředí **Java Runtime Environment** a v něm běží webová služba. Ale na serveru může být spuštěna libovolná webová služba umožňující připojení přes **RESTful API** prostřednictvím **HTTP protokolu**. Díky tomu je aplikace univerzální a podporuje přenositelnost mezi zařízeními.



Obrázek 7.14: Diagram nasazení projektu Detection.

7.3.1 Externí knihovny

Knihovny třetích stran, které v projektu Detection využívám, jsou **serial**, **requests** a **RPi.GPIO**.

Knihovna **serial** je nezbytná, pokud chceme využít GSM modul připojený na sériovém portu. Tato knihovna se stará o kompletní komunikaci mezi rozhraním jazyka Python a ovladačem patřičného GSM modulu.

Knihovna **requests** obsahuje zjednodušené rozhraní k využívání RESTful API požadavků. V aplikaci je toto rozhraní využito k odesílání informací od senzoru, který detekoval signál do webové služby, která tyto informace shromažďuje.

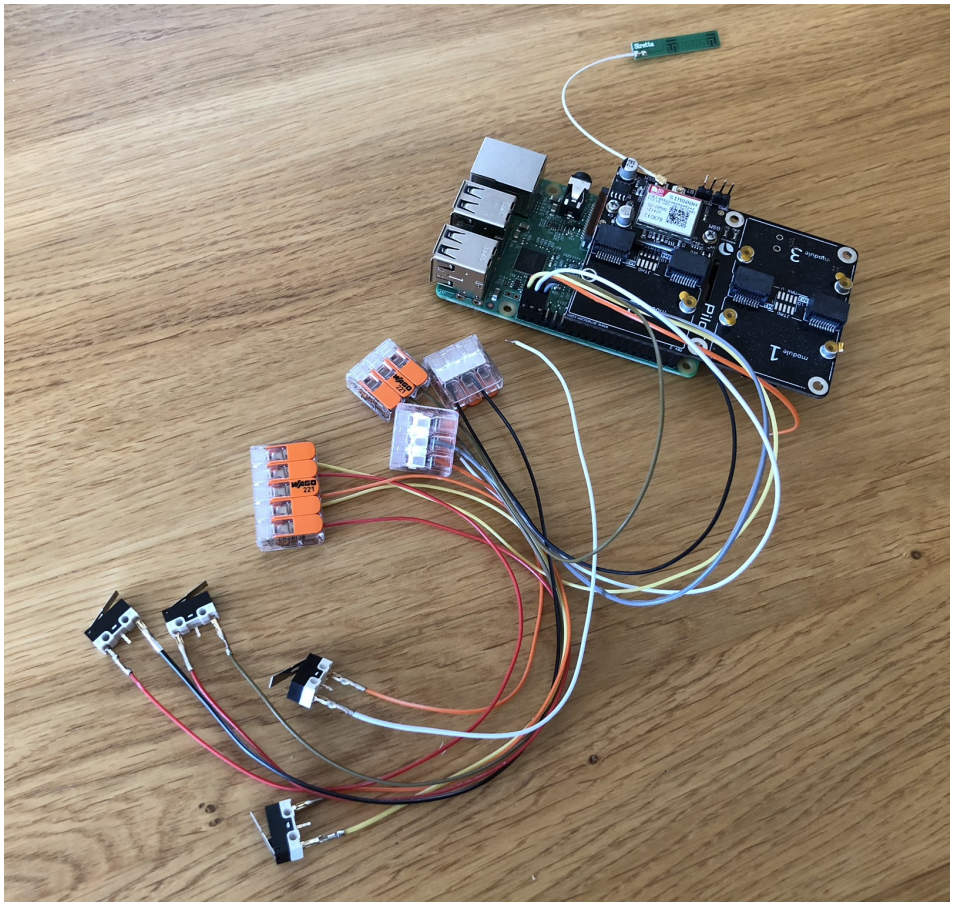
Knihovna **RPi.GPIO** nabízí příjemné rozhraní pro práci s GPIO piny, na kterých aplikace detekuje signál. Programátor se tak nemusí zabývat přímo hardwarovým zpracováním signálu, ale pouze použije vestavěných funkcí této knihovny.

Kapitola 8

Implementace

V této kapitole je popsána implementace této práce.

Na základě analýzy problému a poznatků uvedených v ní byl vytvořen návrh z pohledu softwaru a hardwaru. Podle návrhu a analýzy vhodných technologií byla vypracována implementace. Výsledná implementace je na úrovni hardwaru i softwaru. Softwarová implementace je uvedená v příloze na přiložené SD kartě ve formě Python projektu. Hardwarová implementace je ve formě prototypu znázorněna na obrázku 8.1.



Obrázek 8.1: Prototyp hardwarové implementace projektu Detection.

Kapitola 9

Vyhodnocení

Tato kapitola slouží jako manažerské shrnutí všech podstatných náležitostí v této práci. Je to stručná struktura pro manažery a vedoucí pracovníky, kteří by se na tomto projektu podíleli a díky informacím v této kapitole se snadno rozhodovali, jak s projektem naloží.

9.1 Zadání projektu

Zadáním tohoto projektu bylo navrhnout integrační řídící jednotku chytré domácnosti, která bude snímat elektrický signál na sériovém portu GPIO sběrnice na platformě Raspberry PI 3, tento signál zpracovávat a produkovat podle nastavení při spuštění oznámení ve formě zápisu do textového souboru, odesláním SMS zpráv na telefonní číslo a POST požadavek na rozhraní RESTful API webové služby.

9.2 Řešení problému

Na základě zadání je problém řešen na dvou úrovních:

- softwaru
- hardwaru.

Softwarové řešení zahrnuje vývoj aplikace v jazyce Python 3.0 s vestavěnou knihovnou threading a externími knihovnami RPi.GPIO a requests. Threading umožňuje použití vláken v aplikaci, což dělá ze sekvenčně zpracovávané aplikace aplikaci paralelní. RPi.GPIO poskytuje rozhraní k ovládání GPIO sběrnice umístěné na desce Raspberry PI a requests knihovna poskytuje zjednodušené rozhraní pro posílání požadavků prostřednictvím HTTP protokolu i jeho šifrované verze HTTPS.

Hardwarové řešení zahrnuje propojení desky Raspberry PI3, libovolným obvodem zakončeným čidly a připojeným GSM module prostřednictvím USB portu nebo připevněného k sériovému portu prostřednictvím GPIO sběrnice.

9.3 Etapy projektu

Tento projekt je rozdělen na 3 základní etapy:

- Návrh a vyhotovení vhodných čidel podle požadavků zákazníka
- Vytvoření aplikace s objektově-orientovanými prvky založenou na paralelním programování
- Zajistit odesílání informací prostřednictvím SMS zpráv a POST požadavků na RESTful API informačního systému

9.3.1 1. etapa

Tato etapa byla odhadnuta na 2 pracovní dny a zahrnuje čas strávený na získání požadavků od zákazníka a konstrukci vhodných čidel.

9.3.2 2. etapa

Tato etapa byla odhadnuta na 8 pracovních dní a zahrnuje čas strávený návrhem a implementací aplikace.

9.3.3 3. etapa

Tato etapa byla odhadnuta na 5 pracovních dní a zahrnuje čas strávený na zajištění funkční konfigurace a zprovoznění odesílání informací přes SMS zprávy a RESTful API.

9.4 Cena

Kalkulace cen vychází z podrobnějšího rozboru v kapitole Finanční analýza. V této sekci je pouze pro stručnost shrnuto několik zásadních informací cenové kalkulace projektu.

9.4.1 Kalkulace ceny za vývoj

Cena za vývoj tohoto projektu zahrnuje fixní náklady (vývoj softwaru), což činí **60 000 Kč** a variabilní náklady (pořízení hardwaru), které činí **2 536 Kč** za levnější variantu s USB GSM modulem nebo **4 708 Kč** za variantu použitou v tomto projektu s GSM modulem připojeným na GPIO sběrnici.

9.4.2 Kalkulace ceny pro zákazníka

Podle vypočítaných nákladů jsme určili prodejní cenu na **5 100 Kč** za levnější variantu a **7 500 Kč** za variantu použitou v této práci, aby byl projekt ziskový.

9.5 Součinnost

Součinnost zákazníka zahrnuje pouze specifikaci požadavků, co vše má integrační jednotka splňovat a umožnit vstup do budovy a na pozemek z důvodu montáže zařízení. Doba získávání požadavků se pohybuje mezi 10 minutami až 2 hodinami a doba montáže od 1 hodiny do 8 hodin práce.

Celková součinnost se zákazníkem je odhadnuta maximálně na 1 pracovní den a 1 zaměstnance od zákazníka.

Závěr

Cílem práce bylo analyzovat a navrhnout integrovanou řídící jednotku chytré domácnosti, porovnat s produkty na českém trhu a celý postup a veškeré poznatky zpracovat ve studii proveditelnosti v ucelené formě. V práci jsem také popsal princip zachytávání analogového signálu na sériovém portu integrovaného přímo na platformě Rapsberry PI 3, jeho programové zpracování pomocí programovacího jazyku Python a odeslání krátké SMS zprávy prostřednictvím GSM modulu. Dále jsem popisoval případy, kdy je zařízení využitelné v praxi, protože může předejít větším škodám nebo zachraňovat lidské životy.

Tento projekt byl testován v reálných podmínkách přímo u zákazníka na chromové vaně, kde slouží jako alarm při chybě v přečerpávacím systému. Podle provedených testů, které se týkaly spolehlivosti, byla aplikace akceptována, což značí, že projekt splnil všechna očekávání, které si zákazník stanovil.

V této práci jsme si díky finanční analýze a porovnání konkurenčních řešení ukázali, že při vhodně zvolené finanční strategii může být tento projekt ziskový. Na základě získaných poznatků aplikovaných v této práci jsem také poukázal na smysluplnost tohoto projektu.

V budoucnosti by mohlo toto zařízení detekovat i neoprávněné vniknutí osob a zvolit na to adekvátní reakci. V práci jsem popisoval, že by mohl vylétnout dron ze střechy, analyzovat počet pachatelů a poslat na policejní ústřednu informaci o celkové situaci, případně pochytat pachatele prostřednictvím záchytných sítí, aby jim při zásahu příliš neublížil.

Literatura

[1] Cyberphysics, G. G.: *A Cyberphysics Page: Clock Signals*. [online], [cit. 12.4.2018]. Dostupné z: http://www.cyberphysics.co.uk/topics/electronics/clock.html

[2] Půhoný, J.: *Půhy.cz*. [online], [cit. 25.4.2018]. Dostupné z: https://static.puhy.cz/images/c_176999.jpg

[3] UK, R. P. F.: *Raspberry Pi - Raspberry Pi Hardware Guide requirements | Raspberry Pi Learning Resources*. [online], [cit. 26.4.2018]. Dostupné z: https://www.raspberrypi.org/learning/hardware-guide/components/raspberry-pi/

[4] trade s.r.o., P.: *Huawei E3531 White*. [online], [cit. 13.5.2018]. Dostupné z: https://www.paoli.cz/huawei-e3531-white.html?listtype=searchfulltext&searchparamfull=%20HUAWEI%20E3531%20WHITE

[5] s.r.o., C.: *D-Link DWM-157, 3G USB adapter*. [online], [cit. 13.5.2018]. Dostupné z: https://www.czc.cz/d-link-dwm-157-3g-usb-adapter/131807/produkt

[6] Ltd., I. I. S. C.: *RPI SIM800 GSM/GPRS ADD-ON V2.0*. [online], [cit. 13.5.2018]. Dostupné z: https://www.itead.cc/wiki/RPI_SIM800_GSM/GPRS_ADD-ON_V2.0

[7] Zahradílek, K.; Korytář, M.: *Elektrické signály*. [online], [cit. 8.4.2018]. Dostupné z: https://user.unob.cz/zaplatilek/zel/Tema05.htm

[8] Storey, N.: *Electrical & Electronic Systems*. 2004, [cit. 8.4.2018]. Dostupné z: https://books.google.cz/books?id=JhkgMq_vjmkC&lpg=PA191&ots=2Waika2kcQ&dq=edge%20electric%20signal&hl=cs&pg=PA191#v=onepage&q=edge%20electric%20signal&f=false

[9] Havrlant, L.: *Graf funkce*. [online], [cit. 12.4.2018]. Dostupné z: https://matematika.cz/graf-funkce

[10] Point, T. . T.: *Python - Multithreaded Programming*. [online], [cit. 13.4.2018]. Dostupné z: https://www.tutorialspoint.com/python/python_multithreading.htm

[11] Trdlička, J.: *Přednáška OSY - Procesy a vlákna. Časově závislé chyby. Kritické sekce*. [online], [cit. 13.4.2018]. Dostupné z: https://edux.fit.cvut.cz/courses/BI-OSY/_media/lectures/02/bi-osy-p02-threads.pdf

[12] Droneweb.cz: *Co je to dron?* [online], [cit. 14.4.2018]. Dostupné z: http://www.droneweb.cz/co-je-dron

[13] Russell, S. J.; Norvig, P.: *Artificial intelligence*. 2010, [cit. 17.4.2018].

[14] České republiky, P.: *Krajní nouze, § 28, Trestní zákoník č. 40/2009 Sb.* 2009, [cit. 18.4.2018]. Dostupné z: http://www.psp.cz/sqw/sbirka.sqw?r=2009&cz=40

[15] Koten, V.; Chod, J.: *Model GSM*. 2008, [cit. 22.4.2018]. Dostupné z: http://cvut.summon.serialssolutions.com/#!/search?bookMark=ePnHCXMw42JgAfZbUzkZOEGXeeUouAf78gCFgP0pbgZZN9cQZw_d5LLSknjo6EQ8uLwEdlJAV0LilwcAREgYzA

[16] s.r.o., G. E.: *GM Electronic - webové stránky*. [online], [cit. 22.4.2018]. Dostupné z: https://www.gme.cz

[17] Krynický, M.: *Kinetická energie*. [cit. 22.4.2018]. Dostupné z: http://www.ucebnice.krynicky.cz/Fyzika/1_Mechanika/5_Prace_a_energie/1504_Kineticka_energie.pdf

[18] Krynický, M.: *Mechanická práce I.* [cit. 23.4.2018]. Dostupné z: http://www.ucebnice.krynicky.cz/Fyzika/1_Mechanika/5_Prace_a_energie/1501_Mechanicka_prace_I.pdf

[19] Fikr, J.; Kahovec, J.: *Názvosloví organické chemie*. 2004, [cit. 11.5.2018].

[20] UK, R. P. F.: *Raspberry Pi Foundation UK*. [online], [cit. 25.4.2018]. Dostupné z: https://www.raspberrypi.org

[21] UK, R. P. F.: *Raspberry Pi Downloads*. [online], [cit. 28.4.2018]. Dostupné z: https://www.raspberrypi.org/downloads/

[22] UK, R. P. F.: *Raspberry Pi Software Guide*. [online], [cit. 1.5.2018]. Dostupné z: https://www.raspberrypi.org/learning/software-guide/

[23] UK, R. P. F.: *Raspberry Pi Software Guide*. [online], [cit. 1.5.2018]. Dostupné z: https://www.raspberrypi.org/learning/software-guide/quickstart/

[24] Foundation, P. S.: *Python2orPython3*. [online], [cit. 2.5.2018]. Dostupné z: https://wiki.python.org/moin/Python2orPython3

[25] Beazley, D. M.: *Python: referenční programátorská příručka*. 2002, [cit. 4.5.2018]. Dostupné z: http://cvut.summon.serialssolutions.com/#!/search?bookMark=ePnHCXMw42JgAfZbU5kZuCyArWVj0GnoRpwMbAGVoK3wPEBJYM-Km0HWzTXE2UM3uay0JB46ThEPrv4MLI1A99jilwcA3HoaVQ

[26] Foundation, P. S.: *pySerial API*. [online], [cit. 5.5.2018]. Dostupné z: https://pythonhosted.org/pyserial/pyserial_api.html

[27] Engelfriet, A.: *Choosing an Open Source License*. 2010, [cit. 5.5.2018].

[28] Znamenáček, J.: *Výjimky - Programování v Pythonu*. 2011, [cit. 8.5.2018]. Dostupné z: https://edux.fit.cvut.cz/oppa/BI-PYT/prednasky/bi-pyt_p-06_vyjimky.pdf

[29] Balík, M.; Vagner, L.; Vogel, J.: *Proměnné, základní vstup a výstup*. 2011, [cit. 8.5.2018]. Dostupné z: https://edux.fit.cvut.cz/courses/BI-PA1/_media/lectures/l02-var-cz.pdf

[30] Croston, B.: *RPi.GPIO*. [online], [cit. 5.5.2018]. Dostupné z: https://pypi.org/project/RPi.GPIO/

[31] Foundation, P. S.: *threading - Thread-based parallelism*. [online], [cit. 8.5.2018]. Dostupné z: https://docs.python.org/3/library/threading.html

[32] Reitz, K.: *Requests: HTTP for Humans*. [online], [cit. 9.5.2018]. Dostupné z: http://docs.python-requests.org/en/master/

[33] Electronics, S.: *AT Commands Reference Guide*. 2006, [cit. 9.5.2018]. Dostupné z: https://www.sparkfun.com/datasheets/Cellular%20Modules/AT_Commands_Reference_Guide_r0.pdf

[34] [GB], R. C. L.: *RasPiComm+ main board for Raspberry Pi*. [online], [cit. 13.5.2018]. Dostupné z: https://cz.rs-online.com/web/p/vyvojove-sady-pro-procesory-a-mikrokontrolery/8474894/

[35] a.s., J.: *Jablotron*. [online], [cit. 13.5.2018]. Dostupné z: https://www.jablotron.com/cz/

[36] s.r.o., S. E.: *Selax Electronics s.r.o.* [online], [cit. 13.5.2018]. Dostupné z: http://www.selax.cz/

[37] s.r.o., P. S.: *Zabezpečovací systémy Plzeň, Praha - PCO Service s.r.o.* [online], [cit. 13.5.2018]. Dostupné z: `http://www.pcoservice.cz/`

[38] Váverka, P.: *PV elektronické systémy - Pavel Váverka.* [online], [cit. 13.5.2018]. Dostupné z: `http://www.pavelvaverka.cz/`

[39] Stallings, W.: *Operating systems: internals and design principles.* 2009, [cit. 11.5.2018].

Příloha A

Seznam použitých zkratek

AI Artificial Intelligence

API Application Interface

GPIO General Purpose Input/Output

GSM Global System for Mobile Communications

JEE Java Enterprise Edition

OOP Object-Oriented Programming

OS Operating System

RESTful Representational State Transfer

RPi Raspberry PI

SMS Short Message Service

Příloha **B**

Obsah přiložené SD karty

- `readme.txt` ..... stručný popis obsahu SD karty
- `src`
  - `design` ..... zdrojové soubory návrhu pro Enterprise Architect
  - `impl` ..... zdrojové kódy implementace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF