VŠB TECHNICKÁ UNIVERZITA OSTRAVA | FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Řídicí systémy s počítači

Aplikace pro komunikaci s PIR senzory

Bc. David Racl
Bc. Lucie Zámečníková
Bc. Boris Pustějovský

2022

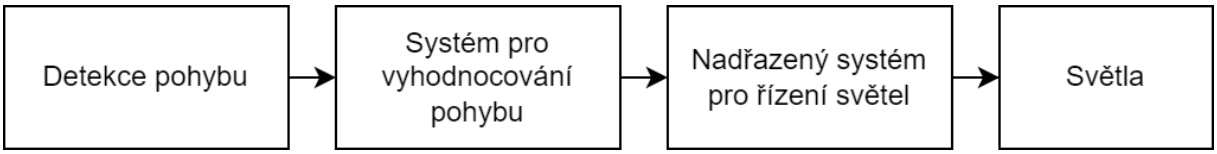
Obsah

Analýza technologického řešení

Úvod

Námi analyzované parkoviště obsahuje celkem 18 svítidel různých typů (BOOS Naica a Thorn R2L2). Pro řešení zadané problematiky byl zvolen centralizovaný systém. Na každý sloup – viz rozmístěný na obr. 1. bude umístěn jeden, případně dva moduly, který každý bude obsahovat pasivně infračervený senzor (PIR), který bude sloužit pro detekci pohybu v dané zóně na parkovišti.

Každý modul bude komunikovat s centralizovanou službou, která bude spuštěna lokálně na školním serveru. Ta bude k dispozici ze školní sítě. Komunikace bude probíhat za pomocí kabelového ethernetu oboustranně – služba bude posílat data do modulů a moduly zpět do služby. K napájení jednotek (modulů se senzory PIR) bude využito rozhraní ethernet pro komunikaci za pomoci technologie Power over Ethernet (PoE). Využije standard 802.3af, který umožňuje napájet zařízení po vedení dlouhém až 100 metrů, při napájecím napětí 48 V.



Obrázek 1: Diagram popusující problematiku zadání

Obrázek 2: Mapa rozmístění PIR senzorů

Vybraná technologie hardwaru:

Variant, jak realizovat modul v návrhu je několik (pořadí je dáno preferencí):

1. Raspberry Pico + PIR senzor

Jako řídící jednotka bude využita vývojová deska Raspberry Pi RP2040, kterou lze naprogramovat v jazyce C, případně Python. Pro podporu komunikace TCP/IP byl vybrán rozšiřující modul PoE-FeatherWing, který bude zároveň řešit i napájení pro RPi. Všechny součástky by byly umístěny do průmyslové krabice se stupněm krytí IP66 a UV ochrannou. PIR senzor je zvolen tak, aby odolal vnějším jevům.

Název	**Cena**
Raspberry Pi RP2040[1]	415 Kč
PoE-FeatherWing [2]	584 Kč
PIR senzor[3]	3000 Kč
Krabice[4]	414 Kč
Celkem	**4413 Kč**

Z hlediska senzorů byly zvoleny PIR senzory, které jsou propojeny s MCU Raspberry RP2040. Raspberry RP2040 je připojeno k ethernetu pomocí síťového kabelu, který slouží pro komunikaci se systémem a zároveň i pro napájení modulu.

2. IP kamera

Druhou variantou je možnost využívat IP kameru. Ta lze nakonfigurovat na danou oblast, lze jí přímo napájet pomocí PoE a po detekci je možné poslat packet o pohybu do služby na serveru. Výhodou je, že kamera zabere mnohem větší úhel záběru, a proto jich bude potřeba méně (pravděpodobně 4).

Název	**Cena**
IP Kamera[5]	**10 749 Kč**

3. Řídící jednotka PLC + PIR senzor

Další variantou je využít PLC, které bude číst digitální vstupy. Komunikace by probíhala podobně, ale využilo opět rozhraní TCP/IP, kterým je zařízení vybaveno.

Název	**Cena**
PLC[6]	2777 Kč
PIR senzor	3000 Kč
Krabice	414 Kč
Celkem	**6191 Kč**

[1] https://www.soselectronic.cz/products/wiznet/w5100s-evb-pico-359020?vat=1&gclid=Cj0KCQjwnvOaBhDTARIsAJf8eVNxCjY6XNpdnPJuUxhMBYIBB4EFIkubHrS9es9tiNUL6kCBEwcNVDMaAjvmEALw_wcB

[2] https://www.tindie.com/products/silicognition/poe-featherwing/

[3] https://www.123securityproducts.com/di601.html

[4] https://www.distrelec.cz/cs/vodotesne-pouzdro-stylove-viko-1555-90x160x60-5mm-svetla-seda-abs-ip66-hammond-1555jgy/p/30011062

[5] https://www.falanzo.cz/ip-kamery/ip-kamera-mobotix-vd-2-ir-720-p-bila/

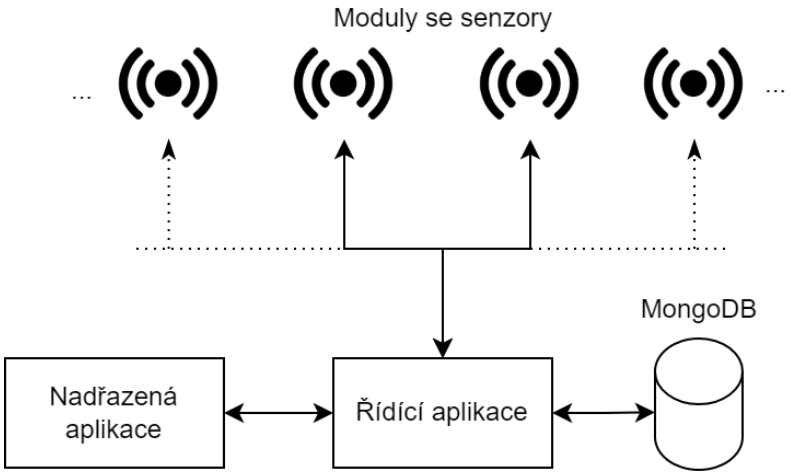
[6] https://www.jork.shop/produkt/automatizacni-systemy/logicky-modul-logo/ridici-jednotky-bez-displeje/6ed1052-2cc08-0ba1-94175.htm

Uživatelské rozhraní

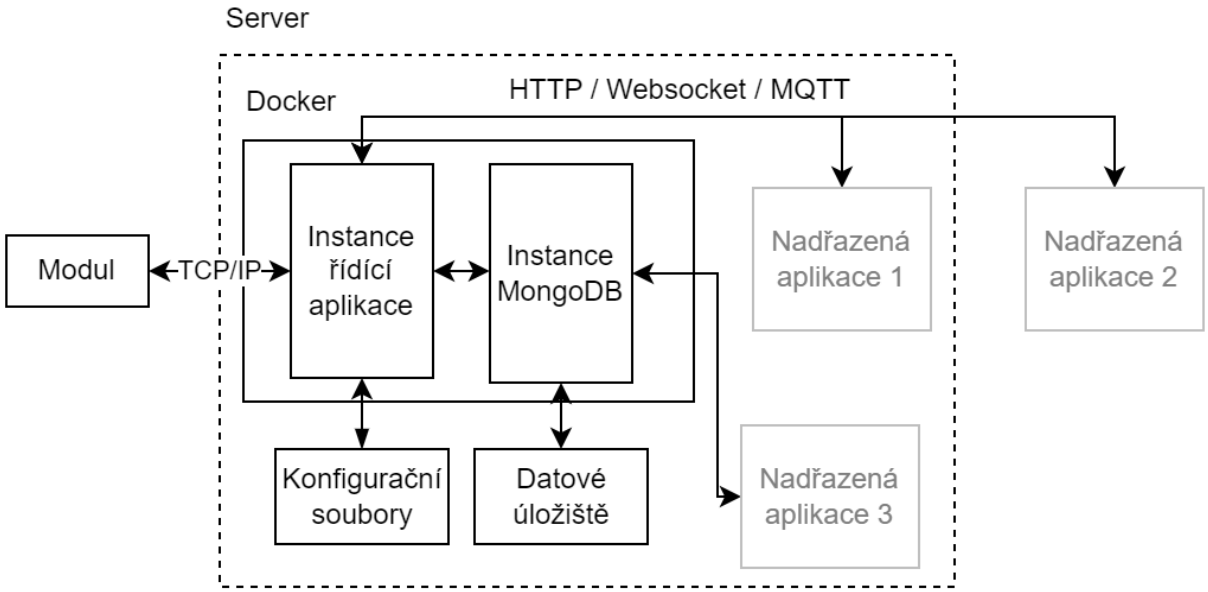
Shodli jsme se, že uživatelské rozhraní je nadbytečné. Důvodů je hned několik: konfigurace systému se provede pouze jednou a pak zůstane neměnná, je zde bezpečnostní riziko a informace, které by systém svou vizualizací sděloval, by nikomu k ničemu nebyli (vizualizaci řeší nadřazený systém).

Historie všech událostí v systému se budou ukládat do nerelační databáze Mongo DB, odkud by následně mohla být data exportována pomocí jednoduchého dotazu.

Dokumentace technologie



Obrázek 3: Obecný popis jednotlivých částí navržené technologie



Obrázek 4: Implementace části softwarového řešení na server

SW analýza

Obecná analýza

Aplikace bude přijímat pakety od PIR senzorů. Výsledky na základě vstupních informací bude vyhodnocovat a požadavky zasílat nadřazenému systému o zapnutí světel. Aplikace bude v pravidelném intervalu zasílat dotazy do modulů, jestli je senzor funkční. V případě, že modul do požadované doby neodpoví, bude systémem zaslán požadavek o přepnutí osvětlení do automatického režimu.

Analýza struktury vnějšího prostředí

Se systémem nebude pravidelně pracovat žádná obsluha. Systém bude řídit osvětlení parkoviště plně automaticky. Bude však možné stáhnout historická data z databáze pro analytické účely.

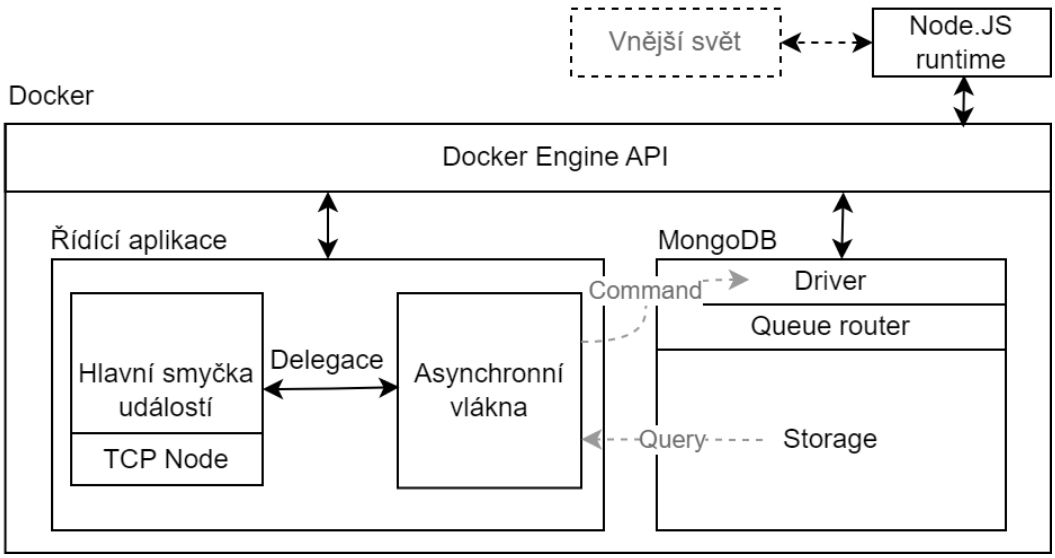
Analýza funkcí

Aplikace bude umožňovat následující funkce:

1. Odeslání dotazu na modul, zda je dosažitelný – kontrola stavu
    - každých 15 sekund
2. Přijímat odpovědi od modulů – změna stavu
    - kdykoliv. Moduly však budou odesílat zprávu pouze při změně stavu
3. Odesílaní výsledků, která světla mají být rozsvícena pro nadřazený systém
    - pouze při změna stavu, pokud jej proces řízení vyhodnotí jinak, než je aktuálně nastaveno
4. Přijímat dotazy, které světla mají být rozsvícena
    - kdykoliv

Analýza komunikace

Interně bude aplikace komunikovat pomocí Docker Engine API. Ta zajistí komunikaci jak s vnějším světem tak i interně mezi dalšími aplikacemi.



Obrázek 5: Interní komunikace řídící aplikace, databáze a vnějšího světa

Aplikace bude přijímat od modul pakety v 3. bytové formátu, kde první byte reprezentuje identifikátor modulu, *druhý* nám říká, který senzor na modulu mění stav a poslední je pro vyjádření aktuálního stavu

senzoru. Celkově tak může v systému být 256 zařízení, každé může mít teoreticky až 256 senzorů a každý senzor může nabývat až 256 stavů.

Příklady zaslaného paketu (0xD0103):

3. byte (MSB)	00001101 (0xD)
2. byte	00000001 (0x1)
1. byte (LSB)	00000011 (0x3)

Definované stavy:

Hodnota	Popis
0x00	Není detekován pohyb
0x01	Je detekován pohyb
0x02	Chyba senzoru
0x03	Rezervovaný stav
0x04	Rezervovaný stav

Kontrola přenosu a správnosti dat není třeba, dle specifik TCP packet vždy dorazí korektně.

Služba bude předávat informace o výsledném stavu světel nadřazenému systému pomocí API. Zpráva bude předávána pomocí HTTP požadavku s návratovým typem objektu application/json.

Struktura JSON je popsána podle validačního schématu OpenAPI v3, viz příloha *openapi.json*.

Analýza obsahu a struktury informací

Aplikace bude zpracovávat obdržené pakety. Podle obsahu paketů bude vykonávat svůj interní program a skládat paket, který bude zasílat ID světel, které zapnout, nadřazenému systému.

Obdržená a zpracovaná data budou uloženy v nerelační databázi. Historie se nebude mazat. Z historie tak bude možné kdykoliv provést rekonstrukci stavů. Samotná databáze se bude zálohovat s periodou jeden den.

Analýza toku informací

Aplikace obdrží paket od senzorů, zpracuje paket, vyhodnotí situaci a zašle požadavek na rozsvícení světla nadřazenému systému, případně nadřazený systém si sám vyžádá aktuálně požadovaný světla. Nezávisle na tomto procesu bude aplikace zasílat dotaz senzorům pro ověření činnost jednotlivých modulů.

Odesílaná data nebudou nijak šifrována, bude však omezen přístup ke komunikační síti – dle pravidel školy.

Analýza slabých míst

Aplikace zasílá nadřazené aplikaci informace o tom, jaké světlo má být zapnuto. Zároveň kontroluje funkčnost senzorů, ale nekontroluje funkčnost nadřazené aplikace. Nadřazené aplikaci jsou data pouze odesílaná, ale není přijímaná zpětná vazba. Systém není redundantní, to znamená, že pokud nebude k dispozici řídící aplikace, nebude možné na změny stavů modulů reagovat.

Systémová specifikace

Software je určen pro automatické rozsvícení a zhasínání světel na parkovišti při detekci pohybu člověka, nebo dopravního prostředku.

Pro provoz je potřeba aktivní komunikace Ethernetu pro přenos dat a napájení Raspberry PR2040. Z hlediska vnějších dat je systém určen pro detekci pohybu, takže se očekává pohybová aktivita, která bude PIR senzorem detekována.

Z hlediska funkčních požadavků funguje aplikace i celý systém zcela automaticky, proto není potřeba od uživatele či správce žádná pravidelná interakce. Senzor očekává, že systém bude se senzorem komunikovat a budou si navzájem vyměňovat data. Nadřazená aplikace očekává, že bude od aplikace dostávat informace o tom, jaká světla má zapnout.

Z hlediska nefunkčních požadavků se očekává, že systém bude odolný vůči rušení i výpadkům. Dostupnost systému by měla být maximální možná – 99,9 %.

Chování za chybových situací

Ve všech případech, kdy je detekována chyba je vždy tato informace delegována nadřazenému systémů, který na tuhle situaci může reagovat jinak.

Modul detekuje chybu senzoru / Modul přestane komunikovat

Modul zašle zprávu o chybě na senzoru. V takovém případě řídící systém bere senzor jako by byl detekován pohyb.

Výpadek řídící aplikace

Běží zde externí služba monitoringu zdraví, která tento výpadek detekuje. V tomto případě se služba pokusí kontejner s řídící aplikací znovu zapnout – v případě 3 neúspěšných pokusů je informován administrátor formou notifikace na mobilní zařízení.

Výpadek databáze

Řídící aplikace po startu všechny aktuální stavy ukládá do své mezipaměti, pokud dojde k výpadku databáze, jsou všechny změny řazeny do fronty, kde čekají na své uložení. Jakmile je databáze k dispozici všechna data se zpětně ukládají. V případě výpadku aplikace však dojde ke ztrátě neuložených dat.

Výpadek nadřazené aplikace

Řídící aplikace nebude nijak ovlivněna.

Detekována špatná konfigurace

V případě špatné konfigurace se aplikace během inicializace vypne s chybovou hláškou.

Požadavky na dokumentaci

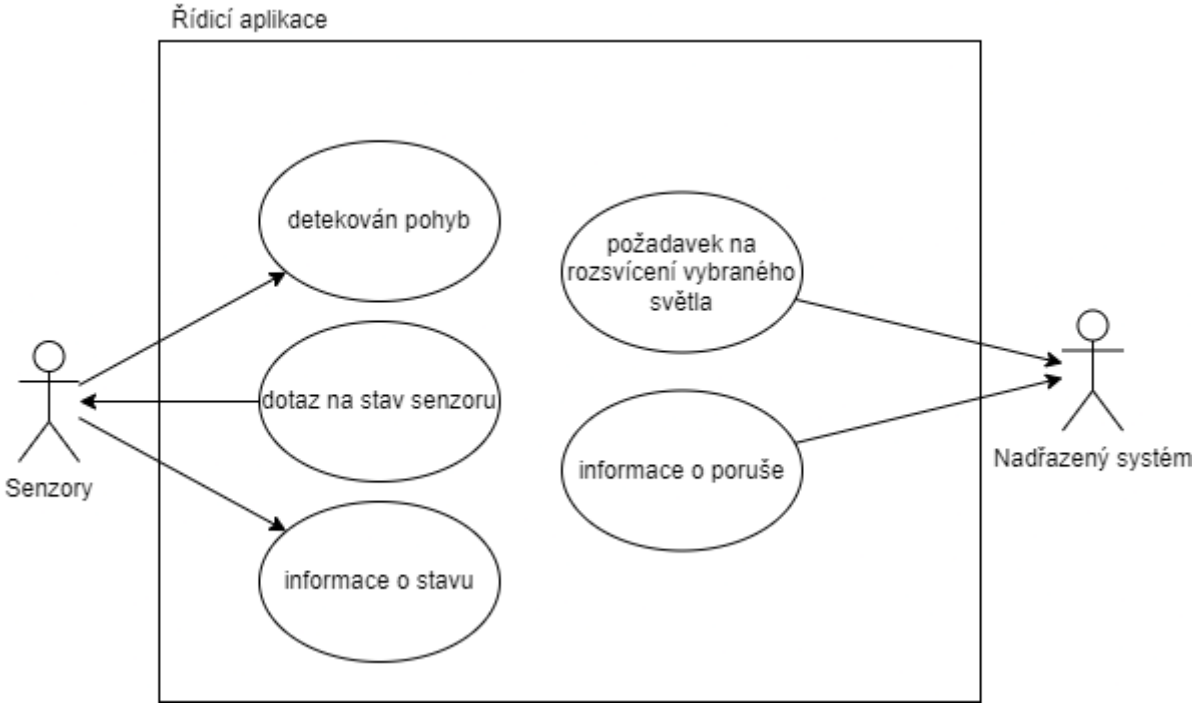
Dokumentace technologie je ve složce datasheets na GitHubu. Zde se nachází popis všech hardwarových komponent, které byly v projektu zmíněny.

Předávací podmínky

Z hlediska testování budou provedeny funkční testy, jednotkové testy, integrační testy a výkonnostní testy. Všechny testy budou prováděny a implementovány až během vývoje software.

UML analýza

Případy užití



Obrázek 6: Use Case Diagram

UC 1

1. senzor detekuje pohyb
2. senzor odešle systému informaci, že detekoval pohyb
3. systém obdrží informaci a zvolí ID světla, které má být spuštěno
4. systém odešle ID světla nadřazenému systému

UC 2

1. systém pošle dotaz senzoru
2. senzor odpoví že je v pořádku

Rozšíření scénáře

2.1. senzor neodpoví do požadované doby

2.1.1 systém pošle znovu dotaz senzoru
2.1.2 senzor neodpoví do požadované doby
2.1.3 systém pošle informaci o poruše senzoru nadřazenému systému

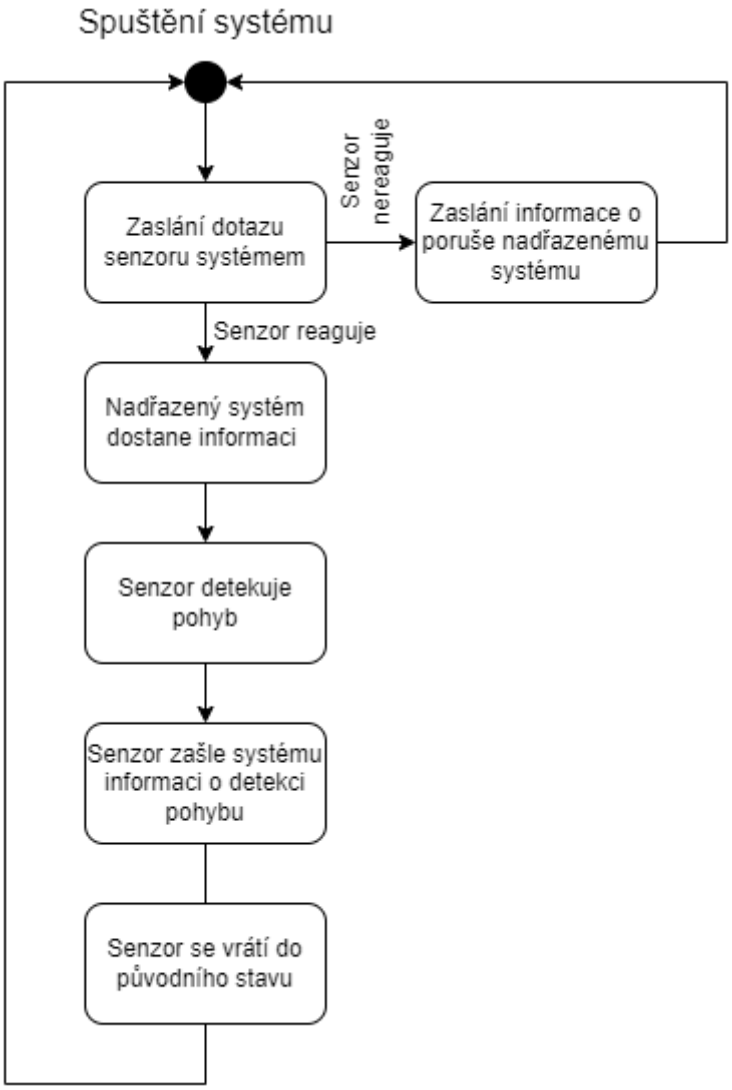
2.2 senzor odpoví, že není v pořádku

2.2.1 systém pošle informaci o poruše senzoru nadřazenému systému

Stavový diagram

Ve stavovém bloku je zobrazený postup funkce PIR senzoru. Nejprve proběhne spuštění nadřazeného systému, posléze se náš systém zeptá PIR senzorů zda komunikují. Pokud senzor komunikuje správně, náš systém zašle informaci nadřazenému systému o funkčnosti, v opačném případě zašle informaci o nefunkčnosti.

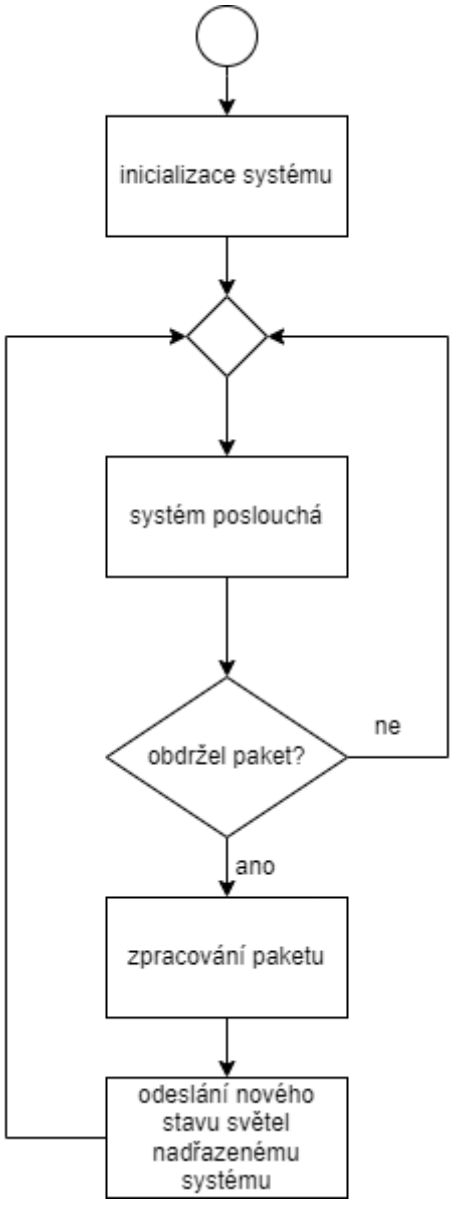
Jakmile senzor zaznamená pohyb zašle paket našemu systému. Systém paket zpracuje a zašle informaci nadřazenému systému, který rozsvítí požadované světlo. Senzor se následně vrátí do stavu, ve kterém neměl zaznamenaný pohyb. Tento cyklus se stále opakuje.



Obrázek 7: stavový diagram

Aktivitní diagram

Diagram popisuje funkci řídicího programu, který poběží na serveru.



Obrázek 8: aktivitní diagram

Bonus: self-assessment

Před záčátkem návrhu jsme se si rozdělili navzájem role. David je vedoucí projektu – stará se o rozdělení úkolů v projektu a pořádal meetingy, Lucie je designer – navrhuje architekturu systému, jak by mohla vypadat a Boris je programátor – navrhuje, jak by software mohl ve skutečnosti pracovat.

Celý projekt jsme navzájem konzultovali a úkoly vždy společně rekapitulovali a hodnotili a výsledky zpracovali společně do finální podoby.