VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ
DEPARTMENT OF INFORMATION SYSTEMS

MODUL PRO ZPRACOVÁNÍ DAT IIOT

IIOT DATA PROCESSING MODULE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **TOBIÁŠ FRAJKA**
AUTHOR

VEDOUCÍ PRÁCE **Ing. JIŘÍ HYNEK, Ph.D.**
SUPERVISOR

BRNO 2025

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Zadání bakalářské práce

165114

Ústav: Ústav informačních systémů (UIFS)
Student: **Frajka Tobiáš**
Program: Informační technologie
Název: **Modul pro zpracování dat IIoT**
Kategorie: Informační systémy
Akademický rok: 2024/25

Zadání:

1. Prostudujte oblast internetu věcí v průmyslu (IIoT – *Industry Internet of Things*) Průmysl 4.0.
2. Seznamte se s problematikou sběru, ukládání a přenosu dat v prostředí IIoT, včetně existujících řešení a standardů (OPC UA, MQTT atd.).
3. Analyzujte systémové požadavky na sběr a přenos dat ve vybraném průmyslovém prostředí IIoT (typ dat, přenosová rychlost, zabezpečení, škálovatelnost).
4. Dle výsledků analýzy navrhněte architekturu řešení pro zpracování dat ve vybraném prostředí IIoT (včetně sběru dat z různých zdrojů, ukládání do vhodné databáze a integrace s dalšími platformami pro zpracování).
5. Implementujte navrhované řešení s důrazem na modularitu a schopnost integrace do stávajících systémů.
6. Otestujte funkčnost a výkonnost výsledného řešení na vybraných systémech. Vyhodnoťte výsledky a navrhněte možná vylepšení nebo rozšíření.

Literatura:

- Alabadi, M., Habbal, A., & Wei, X. (2022). Industrial internet of things: Requirements, architecture, challenges, and future research directions. *IEEE Access*, *10*, 66374-66400.
- Greengard, S. (2015). *The Internet of Things*. MIT Press.
- Lakhwani, K., Gianey, H. K., Wireko, J. K., & Hiran, K. K. (2020). *Internet of things (IoT): Principles, paradigms and applications of IoT*. BPB Publications. ISBN 978-93-89423-365.
- Lv, Z., & Fersman, E. (2022). *Digital twins: Basics and applications*. Springer International Publishing. https://doi.org/10.1007/978-3-031-11401-4
- Tran, D. L., Yu, T., & Riedl, M. (2020, October). Integration of IIoT communication protocols in distributed control applications. In *IECON 2020 The 46th Annual Conference of the IEEE Industrial Electronics Society* (pp. 2201-2206). IEEE.

Při obhajobě semestrální části projektu je požadováno:
Body 1 - 4.

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

Vedoucí práce: **Hynek Jiří, Ing., Ph.D.**
Vedoucí ústavu: Kolář Dušan, doc. Dr. Ing.
Datum zadání: 1.11.2024
Termín pro odevzdání: 14.5.2025
Datum schválení: 22.10.2024

Abstrakt

Práca sa venuje návrhu, implementácii a overeniu funkčnosti modulárneho systému na zber a spracovanie dát v rôznorodom prostredí priemyselného internetu vecí (IIoT) s ohľadom na koncepty myšlienky Industry 4.0. Cieľom práce je navrhnúť riešenie vhodné do produkčného prostredia a otestovať jeho funkčnosť. V úvodných kapitolách sú analyzované dostupné možnosti a využitie systémov na spracovanie dát a spôsoby komunikácie so zariadeniami v priemysle, vrátane štandardov ako OPC-UA a MQTT. Na základe analýzy požiadaviek bol navrhnutý modulárny systém postavený na architektúre mikroslužieb, kde každá služba beží v samostatnom Docker kontajneri. Implementácia systému zahŕňa vstupný modul pre zber dát, výstupný modul pre komunikáciu s externými službami, časovú databázu TimescaleDB pre efektívne ukladanie dát a webovú aplikáciu pre správu a monitorovanie. Technologicky je systém postavený na moderných nástrojoch – .NET 8 pre backend, Next.js s TypeScriptom pre frontend a gRPC pre komunikáciu medzi modulmi. Funkčnosť systému bola overená v spolupráci so spoločnosťou AGEsoft s.r.o., ktorá poskytla možnosti testovania v simulovanom prostredí. Výsledkom práce je plne funkčný, modulárny a škálovateľný systém pripravený na nasadenie v reálnych priemyselných prevádzkach.

Abstract

This thesis focuses on the design, implementation, and validation of a modular system for data collection and processing in the diverse environment of the Industrial Internet of Things (IIoT), taking into account Industry 4.0 concepts. The aim of the work is to design a solution suitable for a production environment and test its functionality. The introductory chapters analyze available options for data processing systems and communication methods with industrial devices, including standards such as OPC-UA and MQTT. Based on requirements analysis, a modular system built on microservices architecture was designed, with each service running in a separate Docker container. The system implementation includes an input module for data collection, an output module for communication with external services, a TimescaleDB time-series database for efficient data storage, and a web application for management and monitoring. Technologically, the system is built on modern tools – .NET 8 for the backend, Next.js with TypeScript for the frontend, and gRPC for inter-module communication. The functionality of the system was verified in cooperation with AGEsoft s.r.o., which provided testing opportunities in a simulated environment. The result is a fully functional, modular, and scalable system ready for deployment in real industrial operations.

Kľúčové slová

Internet vecí, Priemyselný internet vecí, Priemysel 4.0, MQTT, OPC-UA, Modulárnosť, Mikroslužby, IoT

Keywords

Internet of Things, Industrial Internet of Things, Industry 4.0, MQTT, OPC-UA, Modularity, Microservices, IoT

Citácia

FRAJKA, Tobiáš. *Modul pro zpracování dat IIoT*. Brno, 2025. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Jiří Hynek, Ph.D.

Modul pro zpracování dat IIoT

Prehlásenie

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne pod vedením Ing. Jiřího Hyneka, Ph.D. Ďalšie informácie mi poskytla spoločnosť AGEsoft s.r.o., s ktorou som na tejto práci spolupracoval. Uviedol som všetky literárne pramene, publikácie a ďalšie zdroje, z ktorých som čerpal.

.....
Tobiáš Frajka
12. mája 2025

Poďakovanie

Rád by som poďakoval vedúcemu mojej bakalárskej práce za odborné vedenie, cenné rady a pripomienky počas vypracovania tejto práce. Osobitné poďakovanie patrí spoločnosti AGEsoft s.r.o. za poskytnutie príležitosti na spoluprácu, odbornú pomoc a možnosť otestovať navrhnutý systém v reálnych podmienkach. Taktiež ďakujem Ing. Rastislavovi Frajkovi zo spoločnosti AGEsoft s.r.o. za odborné diskusie a myšlienky, ktoré pomohli formovať výsledok tejto práce.

Obsah

Zoznam obrázkov

Kapitola 1

Úvod

Priemysel sa v posledných rokoch výrazne mení – prechádza procesom modernizácie, automatizácie a robotizácie. Sme svedkami prebiehajúcej priemyselnej revolúcie, označovanej ako ***Priemysel 4.0***. Tento pojem zahŕňa množstvo oblastí a prístupov, kde každá popisuje iné možnosti a spôsoby, ako napredovať, digitalizovať a najmä zefektívňovať výrobné procesy. Jedným z podstatných pilierov tohto posunu je Priemyselný internet vecí – ***Industrial Internet of Things (IIoT)*** – spolu s ďalšími technológiami, ako sú *kyber-fyzikálne systémy* (CPS), *big data*, *umelá inteligencia* (AI), *cloud computing* a ďalšie, ktoré umožňujú transformáciu priemyselných podnikov v duchu Priemyslu 4.0. V rámci IIoT sú zariadenia, senzory a stroje prepojené prostredníctvom siete, čo umožňuje vymieňať, zhromažďovať a spracovávať obrovské množstvo dát v reálnom čase. Tieto dáta sú kľúčové pre optimalizáciu výrobných procesov, prediktívnu údržbu, zvyšovanie efektivity a znižovanie nákladov. Avšak zber, spracovanie a analýza takýchto dát predstavujú veľké výzvy, najmä pokiaľ ide o škálovateľnosť, bezpečnosť a integráciu s existujúcimi systémami.

Jedným z hlavných problémov je rôznorodosť zariadení a komunikačných protokolov používaných v priemyselnom prostredí. Výrobcovia využívajú odlišné štandardy a technológie, čo komplikuje interoperabilitu medzi systémami a spomaľuje adopciu paradigmy Priemyslu 4.0. Preto je nevyhnutné navrhovať riešenia, ktoré sú modulárne a flexibilné, aby dokázali komunikovať s rôznymi zariadeniami a adaptovať sa na meniace sa požiadavky. Okrem technických výziev prináša Priemysel 4.0 a IIoT aj významné príležitosti pre podniky, ktoré chcú zvýšiť svoju konkurencieschopnosť a inovatívnosť. Digitalizácia výrobných procesov umožňuje lepšie využitie zdrojov, rýchlejšiu reakciu na meniace sa trhové podmienky a otvára dvere k novým obchodným modelom.

Moja motivácia venovať sa tejto téme vyplýva z osobného záujmu o moderné technológie a ich praktické uplatnenie v priemysle. Práca bola vyvíjaná v spolupráci s firmou AGEsoft s.r.o.[1], ktorá poskytla príležitosti na otestovanie a nasadenie systému v produkčnom prostredí. Táto spolupráca umožnila lepšie porozumieť praktickým požiadavkám priemyselných podnikov a overiť funkčnosť navrhovaného riešenia v praxi.

Cieľom tejto práce je navrhnúť, implementovať a overiť funkčnosť modulárneho systému na zber a spracovanie dát v prostredí priemyselného internetu vecí. Systém je koncipovaný tak, aby bol vysoko znovupoužiteľný, modulárny a škálovateľný, čo umožňuje jeho bezproblémovú integráciu do existujúcich priemyselných riešení a zároveň flexibilné prispôsobenie sa špecifickým požiadavkám rôznych výrobných a monitorovacích scenárov. V rámci práce je vyvinutý referenčný prototyp založený na architektúre mikroslužieb, pričom každá služba

[1] Stránky firmy: `www.agesoft.sk`

beží v samostatnom Docker kontajneri. Tento prístup zaručuje nielen rýchle nasadenie a jednoduché horizontálne škálovanie, ale aj nezávislú aktualizáciu jednotlivých komponentov bez narušenia celkovej funkčnosti systému.

Navrhovaný systém podporuje integráciu s viacerými komunikačnými protokolmi, ako sú MQTT, OPC UA a ďalšie, čo umožňuje pripojenie rôznych senzorov a zariadení, typických pre priemyselné prostredie. Komunikácia medzi mikroslužbami je zabezpečená pomocou technológie gRPC, čím sa dosahuje nízka latencia, synchrónnosť a vysoká priepustnosť. Získané dáta sú následne ukladané do špecializovanej time-series databázy, čo umožňuje ich efektívne spracovanie, archiváciu a neskoršiu analýzu.

V 2. kapitole sa venujem prehľadu konceptu Priemyslu 4.0 a jeho kľúčových technológií, vrátane IIoT, kyber-fyzikálnych systémov, big data, cloud computingu, fog computingu a edge computingu. Predstavím existujúce technológie a spôsoby, ako zariadenia medzi sebou komunikujú, vrátane protokolov ako MQTT, OPC UA, Modbus a AMQP. Nasledujúca kapitola 3 sa zameriava na zber, ukladanie a prenos dát v prostredí IIoT, vrátane analýzy existujúcich riešení a štandardov. Budem sa venovať aj otázkam bezpečnosti a škálovateľnosti pri spracovaní veľkých objemov dát. V ďalšej časti práce, kapitole 4, analyzujem požiadavky na systém, pričom sa sústredím na typy dát, s ktorými bude systém pracovať, rýchlosť prenosu, bezpečnosť a škálovateľnosť. Na základe tejto analýzy definujem špecifikáciu pre navrhovaný systém.

Nasleduje návrh architektúry riešenia, kde detailne popisujem spôsob, ako budem zberať dáta z rôznych zdrojov, ich ukladanie do vhodnej databázy a integráciu s platformami pre ďalšie spracovanie. Implementácia navrhovaného riešenia kladie dôraz na modulárnosť a možnosť integrácie do existujúcich systémov. Tento návrh berie ohľad aj na potreby budúcich používateľov systému. V záverečných kapitolách testujem funkčnosť a výkonnosť výsledného riešenia na vybraných systémoch, vyhodnocujem výsledky a navrhujem možné vylepšenia alebo rozšírenia. Testovanie prebiehalo v spolupráci s AGEsoft s.r.o., čo umožnilo overiť praktickú využiteľnosť systému v priemyselnom prostredí.

Zámerom tejto práce je teda realizovať a overiť navrhovaný modulárny systém na zber a spracovanie dát, ale aj vytvoriť referenčný základ pre budúcich tvorcov aplikácií, platforiem a nástrojov v oblasti priemyselnej automatizácie. Ďalším cieľom je reflektovať na získané poznatky z testovacieho a produkčného nasadenia a prevádzky a definovať body pre zlepšenie či rozšírenie funkčnosti systému v kontexte zefektívnenia celého procesu získavania a spracovávania dát. Výsledkom by mala byť infraštruktúra schopná efektívne premeniť získané surové dáta na hodnotné informácie, čo prispeje k optimalizácii výrobných procesov, posilneniu konkurencieschopnosti a strategickému rozvoju podnikov v duchu koncepcie Priemyslu 4.0.

Kapitola 2

Priemysel 4.0 a priemyselný internet vecí

Táto kapitola uvádza čitateľa do základných myšlienok a princípov Priemyslu 4.0 a jeho úzkeho prepojenia s priemyselným internetom vecí (IIoT). Najprv vysvetľuje, prečo je Priemysel 4.0 považovaný za štvrtú priemyselnú revolúciu a aké kľúčové princípy formujú jeho podobu. Následne sa sústreďuje na predstavenie hlavných technológií, od kyber-fyzikálnych systémov a analýzy big data až po cloudové, fog a edge riešenia. Tieto informácie vytvárajú ucelený rámec, ktorý pomôže lepšie pochopiť vzájomné prepojenie teoretických základov, technologických možností a praktických aplikácií Priemyslu 4.0 v modernom priemyselnom prostredí.

2.1 Úvod do Priemyslu 4.0

Priemysel 4.0 (P4.0) predstavuje novú úroveň organizácie a riadenia celého hodnotového reťazca životného cyklu produktov. Je založený na inteligentných digitálnych sieťach, ktoré sú schopné samostatne kontrolovať a monitorovať výrobné procesy. Nemecká vláda definuje Priemysel 4.0 ako „technickú integráciu kyber-fyzikálnych systémov do výrobných a logistických procesov a využitie internetu vecí a služieb v priemyselných procesoch“ [15]. Pojem Priemysel 4.0 nabral na kredibilite v roku 2011, kedy pod týmto názvom (Industrie 4.0) navrhla spomínaná nemecká vláda projekt pre podniky, politikov a vedeckú obec [10] a v nasledujúcich rokoch na tom začala budovať a rozvíjať svoj priemysel. Predstavuje podľa nich štvrtú priemyselnú revolúciu, ktorá je charakterizovaná zavedením Internetu vecí a služieb (Internet of Things and Services) do výrobného prostredia. V tomto kontexte vznikajú kyber-fyzikálne systémy (CPS), ktoré integrujú fyzické výrobné zariadenia s digitálnymi technológiami. Ide o systémy a technológie, ktoré umožňujú spravovať a riadiť vzájomne prepojené sústavy fyzických prostriedkov a ich výpočtových schopností [19, 6]. CPS predstavujú integráciu fyzických procesov s digitálnymi technológiami, kde senzory a akčné členy umožňujú zbierať dáta z prostredia, ktoré sú následne spracované a analyzované v reálnom čase. Tým sa vytvárajú inteligentné systémy, ktoré dokážu autonómne rozhodovať a optimalizovať výrobné procesy [15]. Podľa nemeckej vlády by mali byť podniky prepojené týmito spôsobmi:

- horizontálne – prepojenie jednotlivých podnikov pre efektívnejšiu koordináciu,
- vertikálne – integrácia zariadení a systémov do podnikateľských procesov,

- digitálne – digitálna integrácia procesov naprieč celým hodnotovým reťazcom.

Pre lepšie pochopenie významu Priemyslu 4.0, je dôležité pozrieť sa na predchádzajúce priemyselné revolúcie, ktoré formovali spôsob, akým vyrábame a konzumujeme produkty:

- Prvá priemyselná revolúcia (koniec 18. storočia) bola charakterizovaná mechanizáciou výroby pomocou vodnej a parnej energie. Táto revolúcia priniesla výrazné zvýšenie produktivity a umožnila masovú výrobu.
- Druhá priemyselná revolúcia (koniec 19. storočia) zaviedla elektrickú energiu do výrobných procesov, čo umožnilo zavedenie montážnych liniek a ďalšie zvýšenie produktivity.
- Tretia priemyselná revolúcia (70. roky 20. storočia) priniesla automatizáciu výroby pomocou elektroniky a informačných technológií. Zavedenie programovateľných logických automatov (PLC) a robotov zmenilo spôsob, akým sa produkty vyrábajú.
- Štvrtá priemyselná revolúcia (Priemysel 4.0) nadväzuje na predchádzajúce a integruje digitálne technológie do všetkých aspektov výroby. Ide o prepojenie fyzických a digitálnych systémov, čo umožňuje vytváranie inteligentných tovární [15].

Môžeme vidieť, že predchádzajúce revolúcie boli špecifikované pohľadom do minulosti - avšak tá 4. bola popísaná a očakávaná s výhľadom do budúcnosti, preto sa mnohé spoločnosti a výskumné ústavy chopili možnosti aktívne sa podieľať na dotváraní tejto myšlienky [18, 11]. Preto nemecká vláda nebola jediná, ktorá sa tohto chytila, ale podobne v USA spoločnosť General Electric podporuje myšlienku Priemyselného internetu (Industrial Internet). Je definovaný ako integrácia fyzických strojov a zariadení so sieťovými senzormi a softvérom, ktoré sa používajú na predpovedanie, kontrolovanie a plánovanie lepších obchodných a spoločenských výsledkov [11].

Aj keď P4.0 predstavuje významnú transformačnú víziu pre moderný priemysel, je dôležité zdôrazniť, že nejde o jednotný súbor nástrojov. Namiesto toho ide o paradigmu, ktorá zahŕňa široké spektrum technológií, princípov a postupov. Tento koncept slúži ako rámec pre integráciu rôznych inovácií s cieľom vytvoriť inteligentnejšie, efektívnejšie a flexibilnejšie výrobné systémy. P4.0 teda nie je definovaný jedinou konkrétnou technológiou, protokolom ani zariadením, ale skôr kombináciou rôznych pokročilých technológií, ako sú kyber-fyzikálne systémy (CPS), internet vecí (IoT), big data a analytika, umelá inteligencia (AI), cloud computing, rozšírená realita (AR) a mnohé ďalšie. Tieto technológie môžu byť implementované rôznymi spôsobmi v závislosti od potrieb a možností konkrétnych podnikov.

2.2 Základné princípy Priemyslu 4.0

Kľúčovou charakteristikou Priemyslu 4.0 je jeho flexibilita a adaptabilita. Tento koncept umožňuje podnikom prispôsobiť implementáciu Priemyslu 4.0 svojim špecifickým požiadavkám a cieľom. Niektoré podniky môžu klásť dôraz na automatizáciu a robotizáciu, zatiaľ čo iné sa môžu zamerať na digitalizáciu procesov a integráciu dátových tokov. Flexibilita konceptu Priemyslu 4.0 tak umožňuje rôznym odvetviam a organizáciám využiť jeho princípy na zlepšenie svojich operácií bez nutnosti dodržiavať pevne stanovený súbor technológií alebo postupov.

Preto je úlohou podnikov a výskumných inštitúcií interpretovať a aplikovať princípy Priemyslu 4.0 spôsobom, ktorý najlepšie vyhovuje ich potrebám. To zahŕňa výber vhodných technológií, prispôsobenie existujúcich procesov a investovanie do vzdelávania zamestnancov. Takýto prístup podporuje inováciu a umožňuje podnikom aktívne sa podieľať na formovaní budúcnosti priemyslu [18, 11]. Keďže sa nejedná o nijak štandardizovaný pojem, chýba jednotná definícia, ktorá by popisovala, čo konkrétne P4.0 je a ktoré oblasti a technológie zahŕňa, často to je na našej interpretácii znalostí z danej oblasti. Preto sa M. Hermann, T. Pentek a B. Otto vo svojom výskume [11] venovali kvantitatívnemu prieskumu, zisteniu centrálnych aspektov P4.0 a odvodení princípov, ktoré sa budú dať použiť pri vytváraní systému s ohľadom na myšlienky P4.0. Identifikovali 4 najdôležitejšie oblasti: prepojenosť (Interconnection), informačná transparentnosť (Information transparency), decentralizované rozhodovanie (Decentralized decisions) a technická asistencia (Technical assistance).

Prepojenosť zahŕňa prepojenie strojov, zariadení, senzorov a ľudí prostredníctvom Internetu vecí (IoT) a Internetu ľudí (IoP), čo umožňuje komunikáciu a spoluprácu v reálnom čase, čím vzniká Internet všetkého (IoE). IoE napríklad podľa spoločnosti Cisco označuje ľudí, veci a miesta, ktoré môžu zviditeľniť svoje služby iným entitám v okolí [16].

Informačná transparentnosť popisuje vytvorenie digitálneho modelu alebo virtuálnej kópie reálneho sveta *(digital twin)* vďaka zvyšujúcemu sa počtu dátových bodov, ako napríklad senzorov a strojov [23]. Vďaka tomu vieme robiť vhodné rozhodnutia na základe poskytnutých informácií a systém vie sám vykonávať úlohy vďaka vstupom z reálneho aj digitálneho prostredia.

Vďaka prepojenosti systémov a dostupnosti informácií je potom možné prerozdeliť rozhodovacie procesy na najnižšie úrovne a vďaka tomu znížiť čas potrebný na vykonanie rozhodnutia a zvýšiť efektivitu a flexibilitu. **Decentralizované rozhodovanie** tak umožňuje systémom a pracovníkom konať autonómne na základe aktuálnych informácií. Tým, že sú objekty a ľudia prepojení a informácie sú transparentné, môžeme efektívne kombinovať lokálne a globálne dáta pre lepšie rozhodovanie a zvýšenie celkovej produktivity. V dôsledku stále sa zvyšujúcej zložitosti priemyselného prostredia sa úloha človeka posúva od priameho vykonávania úloh k pozícii strategického rozhodovateľa a koordinátora.

Aby sa však daný pracovník dokázal efektívne orientovať v množstve dostupných informácií, potrebuje podporu vo forme **asistenčných systémov**. Tieto systémy mu umožňujú prehľadne zobrazovať veľké objemy dát v správnom kontexte, čo mu napomáha pri prijímaní správnych rozhodnutí a rýchlom riešení vzniknutých problémov. Moderné technológie, ako sú inteligentné zariadenia, nositeľná elektronika a pokročilé softvérové nástroje, zohrávajú kľúčovú úlohu v poskytovaní tejto podpory a umožňujú pracovníkom efektívne zvládať výzvy súčasného priemyslu. Tieto štyri základné princípy Priemyslu 4.0 predstavujú rámec pre navrhovanie a implementáciu moderných priemyselných systémov. Ich implementácia je kľúčová pre úspešné zvládnutie výziev súčasného priemyslu a prechod k inteligentnejšej a efektívnejšej výrobe.

2.3 Technológie Priemyslu 4.0

Ako bolo spomenuté, P4.0 nie je jedna technológia, ktorá určuje, ako má byť systém implementovaný, ale ide o skupinu technológií s rozsiahlym potenciálom na inovácie a širokými možnosťami implementácií. Ide napríklad o spomínané CPS systémy, big data, umelú inteligenciu, augmentovanú a virtuálnu realitu, cloud, fog a edge computing, či IoT a IIoT.

2.3.1 Kyber-fyzikálne systémy (CPS)

Kyber-fyzikálne systémy (Cyber-Physical Systems, CPS) predstavujú integráciu fyzických procesov s digitálnymi výpočtovými a komunikačnými technológiami. Tieto systémy umožňujú úzke prepojenie medzi fyzickým svetom a kybernetickým priestorom, pričom spolupracujú v reálnom čase na dosiahnutie spoločných cieľov. Podľa Y. Liu a kol. [21] môžeme identifikovať štyri hlavné charakteristiky CPS:

- Fyzický systém: zahŕňa akčné členy *(aktuátory)* a elektromechanické zariadenia, ktoré interagujú s fyzickým prostredím a vykonávajú fyzické úlohy.
- Kybernetický a informačný systém: obsahuje riadiacu logiku, senzorové jednotky, siete, úložiská a správu pamäte. Tieto komponenty spracúvajú a prenášajú informácie potrebné na riadenie fyzických procesov.
- Integrácia heterogénnych systémov: predstavuje spojenie rôznorodých kybernetických a fyzických komponentov do jedného celku. Táto integrácia umožňuje vzájomnú interakciu a koordináciu medzi rôznymi systémami.
- Požiadavky na bezpečnosť, prácu v reálnom čase a predvídateľnosť: CPS musia byť navrhnuté tak, aby boli odolné voči hrozbám, schopné pracovať v reálnom čase a zabezpečovali predvídateľné správanie systému.

V rámci CPS sú senzory a akčné členy prepojené prostredníctvom vysokoúrovňových riadiacich jednotiek. Tieto systémy často komunikujú s vonkajším prostredím, čo zvyšuje riziko kybernetických útokov na vnútorné časti systému. Oneskorenie alebo narušenie jedinej úlohy v dôsledku takýchto útokov môže viesť k nepredvídateľnému správaniu celého systému. Implementácia CPS pomáha riešiť otázky bezpečnosti, schopnosti práce v reálnom čase a spoľahlivosti pri monitorovaní a riadení procesov. Tieto systémy musia byť schopné zvládať problémy v systéme, byť škálovateľné a odolné voči hrozbám [21, 27, 14].

2.3.2 Big data

Big data predstavuje spracovanie, analýzu a využitie obrovských množstiev dát, ktoré pochádzajú z rôznorodých zdrojov a rýchlo rastú svojím objemom aj komplexnosťou. Tieto dáta, ktoré zahŕňajú štruktúrované (napr. databázy), semi-štruktúrované (napr. XML) a neštruktúrované formy (napr. texty, obrázky, videá), vyžadujú špecializované technológie a prístupy na spracovanie a využitie [34, 42]. V priemyselnom kontexte je big data kľúčovým prvkom Priemyslu 4.0 a Priemyselného internetu vecí (IIoT), keďže sa vzťahuje na spracovanie a analýzu obrovských objemov dát, ktoré sú generované rôznymi zdrojmi v reálnom čase. Tieto dáta môžu pochádzať zo senzorov, strojov, výrobných liniek, systémov riadenia výroby a ďalších zariadení prepojených v rámci IIoT [24].

Big data je podľa [34] možné definovať pomocou 3V:

- Objem *(volume)* – vysoký objem dát na zber, ktoré sú vyprodukované veľkým množstvom zdrojov
- Rýchlosť *(velocity)* – rýchlosť, akou sú dáta generované, spracovávané a analyzované
- Rôznorodosť *(variety)* – rôzne typy a formáty dát (štruktúrované, neštruktúrované, semi-štruktúrované).

V rámci Priemyslu 4.0 umožňuje big data podnikom využiť tieto obrovské množstvá dát na zlepšenie rozhodovacích procesov, optimalizáciu výrobných operácií a vývoj nových stratégií.

2.3.3 Cloud Computing, Fog Computing a Edge Computing

Efektívne spracovanie a využitie obrovských objemov dát v Priemysle 4.0 zabezpečujú technológie ako cloud computing, fog computing a edge computing. Tieto prístupy umožňujú podnikom optimalizovať procesy, zlepšovať rozhodovanie a vytvárať nové obchodné príležitosti.

Cloud computing poskytuje prístup k zdieľaným výpočtovým zdrojom cez internet, čo umožňuje ukladať a spracovávať dáta bez potreby investovať do vlastnej infraštruktúry [26]. Medzi výhody patrí škálovateľnosť a nákladová efektívnosť, avšak výzvou môže byť latencia, bezpečnosť a závislosť na pripojení. **Fog computing** prináša výpočtové a úložné zdroje bližšie k miestu generovania dát, čím znižuje latenciu a zlepšuje reakčný čas [8]. Slúži ako medzivrstva medzi edge a cloudom, umožňujúc rýchlejšie spracovanie dát. Výzvami sú komplexnejšia správa a potreba zabezpečenia distribuovaných zdrojov. **Edge computing** posúva spracovanie dát priamo na zariadenia alebo senzory na „okraji“ siete [36]. To minimalizuje latenciu a znižuje potrebu prenosu dát do centrálneho systému. Ideálny pre aplikácie vyžadujúce okamžitú odozvu, no je potrebné zohľadniť obmedzenia plynúce z nižšieho potenciálneho výkonu.

Kombinácia týchto technológií umožňuje vytvoriť flexibilnú a efektívnu infraštruktúru pre IIoT aplikácie. Edge zariadenia zbierajú a predbežne spracovávajú dáta, fog uzly poskytujú lokálne služby a cloudové platformy zabezpečujú globálnu analytiku a ukladanie.

2.4 Priemyselný internet vecí

Priemyselný internet vecí (Industrial Internet of Things, IIoT) predstavuje jednu z kľúčových technológií, ktorá umožňuje transformáciu podnikov pod ideou P4.0. IIoT je špecifickou podmnožinou širšieho konceptu Internetu vecí (IoT), ktorá sa zameriava na prepojenie priemyselných zariadení, strojov, senzorov a systémov s cieľom zlepšiť efektivitu, produktivitu a bezpečnosť výrobných procesov. Na rozdiel od „spotrebiteľského“ internetu vecí, ktorý H. Boyes vo všeobecnosti popisuje ako skupinu infraštruktúr, ktoré prepájajú pripojené objekty a umožňujú ich manažment, získavanie údajov a prístup k dátam, ktoré vygenerujú, pričom pripojené objekty sú senzory a/alebo akčné členy vykonávajúce špecifickú funkciu, ktoré sú schopné komunikovať s inými zariadeniami [9], tak IIoT definuje ako systém, ktorý prepája inteligentné zariadenia a kyber-fyzikálne systémy s informačnými technológiami, a často aj s cloudovými alebo edge computing platformami. Tento prepojený systém umožňuje v priemyselnom prostredí inteligentný a autonómny prístup k informáciám, ich zber, analýzu, komunikáciu a výmenu v reálnom čase.

Napríklad autori [16] uviedli, že Industrial Internet of Things (IIoT) (známy aj ako Industrial Internet), predstavuje špecifickú kategóriu IoT aplikácií, ktorú preferujú veľké big-tech[1] spoločnosti. Presnosť, s akou stroje dokážu vykonávať úlohy ako zber dát a komunikácia, prevyšuje ľudské schopnosti, čo výrazne podporilo adopciu IIoT. Podľa nich medzi hlavné stavebné kamene IIoT patria komunikácia stroj-stroj (M2M), analýza veľkých dát a techniky strojového učenia. Tieto dáta umožňujú firmám rýchlejšie identifikovať a riešiť problémy, čo vedie k úsporám času a finančných prostriedkov.

[1] veľké technologické spoločnosti, ako Google, Amazon, Microsoft atď.

V práci [13] sa autori pozreli na používané protokoly v prostredí priemyselného internetu a porovnali ich výkonnosť. Skúmali možnosti viacerých protokolov založených na 2 architektúrach – synchrónne request-response (žiadosť-odpoveď) a asynchrónne publish-subscribe (publikovať-odoberať). Podľa nich je v súčasnosti najväčším problémom IIoT aplikácií využitie množstva proprietárnych protokolov na komunikáciu medzi jednotlivými zariadeniami. Preto sa pozreli na možnosti a výhody použitia MODBUS TCP a/alebo MQTT v týchto systémoch.

M. Alabadi a spol. [4] nadväzujú na Priemysel 4.0 a uvádzajú, že IIoT je technologický pokrok, ktorý vedie k implementáciám princípov Priemyslu 4.0 prostredníctvom zvýšenia produktivity a hospodárskeho vplyvu výrobného sektora. To je umožnené globálnou konektivitou medzi systémami v rôznych lokalitách. Popisujú aj rozdiely medzi IoT a IIoT:

- IoT sa zameriava na optimalizáciu spotreby, zvýšenie osobného komfortu a kontrolu nákladov. Využíva sa na automatizáciu každodenných domácich procesov.
- IIoT sa usiluje o maximálnu efektivitu a bezproblémový beh na akomkoľvek zariadení. Používa sa najmä na monitorovanie výroby a environmentálnych aspektov v podnikoch.
- IIoT vyžaduje nízku latenciu a systémy by mali podporovať automatické presmerovanie do záložného systému pri poruchách.
- IIoT zariadenia musia odolávať extrémnym vplyvom prostredia a musia byť škálovateľné pre rozsiahle siete s tisíckami senzorov a zariadení.

2.4.1 Príklady použitia IIoT

Podľa viacerých zdrojov [16, 13, 4, 9] IIoT nachádza široké uplatnenie v rôznych oblastiach priemyselných a podnikových procesov. Medzi hlavné použitia IIoT patria:

- **Inteligentné budovy, továrne a výroba:**
  - **Zlepšenie efektivity a flexibility výrobných procesov:** Optimalizácia výrobných procesov, znižovanie prestojov a adaptácia na meniace sa požiadavky trhu.
  - **Vzdialené monitorovanie výroby:** Sledovanie výrobných operácií v reálnom čase, umožňujúce okamžité zásahy pri identifikácii problémov. Monitorovanie funkčnosti a stavu zariadení.
  - **Využitie rozšírenej reality, simulácií a virtuálnych prototypov:** Používanie AR/VR technológií pre školenie pracovníkov, simuláciu výrobných procesov a testovanie nových výrobkov bez potreby fyzických prototypov.
  - **Zber dát z výrobných zariadení pre analýzu a optimalizáciu:** Zhromažďovanie a analýza dát z rôznych výrobných zariadení na identifikáciu trendov, prediktívnu údržbu a optimalizáciu výrobných procesov.
  - **Zlepšenie riadenia spotreby energie vo výrobe:** Monitorovanie a optimalizácia energetickej spotreby pomocou inteligentných senzorov a riadiacich systémov.
  - **Prediktívna údržba:** Využívanie analytických nástrojov na predikciu porúch a plánovanie údržby predtým, než dôjde k zlyhaniu zariadení.

- **Inteligentné mestá:**
  - **Riadenie dopravy a verejnej dopravy:** Implementácia inteligentných systémov na riadenie dopravy a optimalizáciu verejnej dopravy.
  - **Inteligentné osvetlenie:** Používanie IoT technológií na automatizáciu a optimalizáciu mestského osvetlenia.
  - **Monitorovanie životného prostredia:** Sledovanie environmentálnych parametrov pre zlepšenie kvality života v mestách.
  - **Zber dát o mestskej infraštruktúre:** Zhromažďovanie a analýza dát o mestskej infraštruktúre pre jej efektívnejšie riadenie a údržbu.

Zdroje sa zhodujú v tom, že IIoT sa využíva predovšetkým v priemyselnom prostredí na automatizáciu, monitorovanie a optimalizáciu procesov. Jeho použitie je široké a zahŕňa inteligentné továrne, dopravu, logistiku či správu budov. Spoločnými cieľmi sú zvýšenie efektivity, zníženie nákladov, zlepšenie kvality a bezpečnosti a umožnenie diaľkového riadenia a monitorovania.

2.4.2 Problémy implementácie IIoT a budúcnosť

Jednou z hlavných výziev v oblasti IIoT je zaistenie spoľahlivej, efektívnej a škálovateľnej komunikácie vo veľkých priemyselných prostrediach [12, 4]. Toto si vyžaduje riešenie problémov s latenciou a nedostatočnou interoperabilitou. V literatúre sa zdôrazňuje, že IIoT infraštruktúra často zahŕňa heterogénne zariadenia a rôznorodé protokoly, čo motivuje vývoj pokročilých integračných techník a architektúr pre výmenu dát. Rozsiahle priemyselné systémy navyše generujú enormné množstvá dát, ktoré je nevyhnutné spoľahlivo zbierať, ukladať a analyzovať v reálnom čase s cieľom zabezpečiť presné a včasné rozhodovacie procesy [3]. Kľúčovou požiadavkou je optimalizácia prerozdelenia zdrojov (výpočtový výkon, pamäť, energia), aby zariadenia v sieti dokázali fungovať nepretržite s dostatočnou kapacitou a bez výpadkov.

Nemenej dôležitým problémom je ochrana dát a infraštruktúry IIoT pred kybernetickými hrozbami, pričom sa kladie dôraz na integritu a dostupnosť údajov v rozsiahlych sieťach. IIoT zariadenia sú často charakterizované obmedzenými zdrojmi, čo sťažuje implementáciu robustných bezpečnostných mechanizmov [9]. Útoky na IIoT môžu mať vážne dôsledky, vrátane narušenia výrobných procesov, finančných strát, a dokonca aj ohrozenia ľudského zdravia. Zdroje tiež poukazujú na potrebu implementovať bezpečnostné opatrenia už v počiatočnej fáze návrhu IIoT systémov a pravidelne monitorovať siete pre detekciu a prevenciu hrozieb [23]. Zariadenia v IIoT prostredí môžu byť statické alebo aj mobilné, čo zvyšuje nároky na spoľahlivosť a nepretržitú dostupnosť systému. Zdroje preto zdôrazňujú potrebu škálovateľných architektúr, schopných efektívne reagovať na rast a meniace sa požiadavky. Budúce IIoT riešenia by mali byť flexibilné a adaptabilné, aby sa dokázali dynamicky prispôsobovať novým podmienkam a vznikajúcim potrebám [3].

Autori sa zhodujú, že budúci vývoj IIoT bude charakterizovaný implementáciou pokročilých technológií, cieľom efektívnejšieho využívania zdrojov a hlbšou integráciou s konceptom Priemyslu 4.0. Očakáva sa, že umelá inteligencia (predovšetkým deep learning), edge computing a big data budú kľúčovými prvkami spracovania údajov takmer v reálnom čase. Dôraz sa bude klásť na vývoj škálovateľných a flexibilných architektúr, schopných zvládať rýchlo rastúci počet zariadení a obrovské množstvá produkovaných dát. Ďalším trendom bude aj zvýšená pozornosť smerujúca k energetickej efektivite a optimalizácii prenosových

protokolov, aby sa minimalizovali náklady a zaistila kompatibilita medzi rôznorodými systémami. Taktiež sa očakáva širšia integrácia mobilných technológií a rozširovanie IIoT do nových odvetví, spolu s využívaním prediktívnej analýzy a údržby na znižovanie prestojov a prevádzkových nákladov [9, 12, 3].

Kapitola 3

Spracovanie dát v IIoT prostredí

V kontexte IIoT predstavuje zber dát jeden z ústredných problémov. Kvalitný a spoľahlivý zber dát zo strojov, výrobných liniek či senzorov je kľúčový pre zvyšovanie digitálnej úrovne priemyselných systémov, podporu systémovej integrácie, diagnostiku porúch a nasadzovanie digitálnych dvojičiek. Zároveň je však potrebné dáta nielen efektívne zhromažďovať, ale ich aj bezpečne prenášať a následne ukladať v takom formáte, ktorý umožní ich ďalšie spracovanie a analýzu v reálnom čase i spätne.

V priemyselnej praxi sú často kľúčovými prvkami riadiace systémy, ako napríklad PLC[1], ku ktorým sú pripojené rôzne senzory a akčné členy. Problém však nastáva v tom, že priemyselné zariadenia podporujú len úzky výber protokolov a technológií, ktoré sú navyše často proprietárne, čo komplikuje vzájomnú komunikáciu, prenos dát a následnú integráciu do dátovej infraštruktúry. Aby bolo možné dáta spracovávať efektívne a jednotným spôsobom, je potrebné zjednotiť rôzne priemyselné protokoly do jedného unifikovaného komunikačného kanála [5]. Tento unifikovaný prístup následne vytvorí predpoklady nielen pre spoľahlivý zber a prenos dát, ale aj pre ich vhodné ukladanie do úložísk a databáz, z ktorých môžu byť dáta ďalej spracované pokročilými analytickými nástrojmi.

3.1 Zber a prvotné spracovanie dát

Zber a spracovanie dát *(data ingestion)* predstavuje proces získavania dát z rôznych zdrojov a ich transport do cieľového systému (napr. databáz) s dôrazom na efektivitu, správnosť a aktuálnosť. Tradične boli tieto úlohy riešené konceptom známym ako ETL [37] (Extract, Transform, Load), kde sa dáta v dávkach extrahovali, transformovali a následne nahrávali do cieľových úložísk. Staršie prístupy však často využívali dávkové spracovanie, pričom dáta boli prenášané medzi jednotlivými krokmi prostredníctvom súborov. Tento prístup bol síce postačujúci pre dátové sklady a aplikácie, kde nebol vysoký tlak na aktuálnosť údajov, ale dnes naráža na svoje limity [25]. Moderné priemyselné aplikácie v kontexte IIoT prinášajú oveľa náročnejšie požiadavky na zber dát. Zariadenia, senzory a výrobné linky generujú kontinuálne veľké objemy údajov, ktoré sa môžu zmeniť v krátkych časových intervaloch. IoT a IIoT systémy si vyžadujú takmer okamžitú reakciu na zmeny vo fyzickom prostredí, čo znamená, že dáta musia byť spracované a sprístupnené v reálnom čase alebo s minimálnou latenciou. Preto aj v kontexte tejto práce je potrebné preskúmať možnosti, ako škálovateľne spracovávať dáta s takmer okamžitou odozvou.

[1] PLC – *programmable logic controller* (programovateľný logický automat) – relatívne malý priemyselný počítač, ktorý slúži na automatizáciu procesov v reálnom čase

3.1.1 ETL (Extract, Transform, Load)

Používa sa primárne pri zbere dát z existujúcich systémov alebo viacerých databáz za účelom ďalšieho agregovania do dátových skladov (*data warehouses*) a prípravy dát pre ďalšie spracovanie. Postup ETL sa podľa [37, 38] skladá z troch krokov:

- **Extract** – proces identifikácie a extrahovania správnych a relevantných dát z rôznorodých zdrojov a ďalších databáz. Cieľom je získať čo najviac hodnotných dát bez vplyvu na beh zdrojových systémov.
- **Transform** – komplexný krok zahŕňajúci operácie nad extrahovanými dátami. Najčastejšie ide o rôzne filtrovanie, spájanie s dátami z iných tabuliek a zdrojov či modifikácie na korektný formát pre uloženie.
- **Load** – prenesenie transformovaných a upravených dát do dátového skladu, kde sú usporiadané do faktových a dimenzionálnych (databázových) tabuliek pre efektívne ukladanie a ďalšie analýzy.

Procesy ETL sú efektívne pre dávkové spracovanie dát z tradičných a často starších systémov, kde sa dáta najprv extrahujú, následne transformujú a potom nahrávajú do dátového skladu. Avšak pri potrebe spracovávania dát v reálnom čase môže ETL viesť k oneskoreniam a preťaženiu systémov. Alternatívou je ELT (Extract, Load, Transform), kde sa dáta najprv načítajú do cieľového úložiska a transformácie prebiehajú priamo tam, čo zvyšuje efektivitu a umožňuje lepšiu škálovateľnosť pri spracovaní veľkých objemov dát [37, 38].

3.1.2 Streaming Data Pipelines

Streamingový prístup (tzv. *Streaming Data Pipelines* sa odlišuje od dávkového spracovania ETL a ELT tým, že dáta sú spracovávané nepretržite, hneď po ich príchode [25]. Používanie nástrojov a platforiem ako Apache Kafka, Apache Flink, Spark Structured Streaming, či cloudových služieb (napr. Amazon Kinesis, Azure Event Hubs, Google Pub/Sub) umožňuje:

- **Minimálnu latenciu**: dáta sú k dispozícii takmer okamžite po ich vytvorení, čo je kľúčové pre reaktívne systémy, prediktívnu údržbu a rozhodovanie v reálnom čase.
- **Škálovateľnosť a odolnosť voči nárazovým zmenám v objeme dát**: platformy pre spracovanie prúdových dát sú navrhnuté na horizontálne škálovanie a prácu s veľkými objemami údajov.
- **Jednoduchšia integrácia so špecializovanými analytickými nástrojmi**: prúdové dátové toky môžu byť napojené priamo na databázy typu time-series, BI nástroje alebo AI/ML platformy, pričom dáta sú aktuálne a pripravené na spracovanie bez zbytočných medzikrokov.

V širšom kontexte umožňujú Streaming Data Pipelines organizáciám efektívne spracovávať neustále prichádzajúce dáta, čím podporujú rýchlejšie a informovanejšie rozhodovanie [22]. Tieto systémy poskytujú robustné riešenia pre spracovanie veľkých objemov dát v reálnom čase, čo je neoceniteľné pre rôzne oblasti ako finančné služby, zdravotníctvo, e-commerce, telekomunikácie či priemysel. Streaming Data Pipelines umožňujú kontinuálne monitorovanie a analýzu dát, čo vedie k zvýšenej operačnej efektívnosti, lepšiemu zákazníckemu servisu

a schopnosti rýchlo reagovať na meniace sa podmienky trhu. Napriek ich výhodám predstavujú nároky na odborné znalosti a investície do infraštruktúry významné výzvy, ktoré si vyžadujú starostlivé plánovanie a implementáciu [16, 17].

Napriek tomu, že ETL a ELT postupy majú svoje využitie a sú dostatočné v prostrediach s nižšími nárokmi na aktuálnosť a objem dát, v kontexte IIoT sa výrazne zvyšuje potreba okamžitej dostupnosti informácií a schopnosti spracovávať veľké množstvá dát v reálnom čase. Priemyselné systémy produkujú dáta nepretržite, a to často v enormnom objeme, pričom je nevyhnutné okamžite reagovať na zmeny prostredia, plánovať údržbu či predchádzať poruchám. Streaming Data Pipelines, ktoré umožňujú plynulé, kontinuálne spracovanie dát a rýchlu odozvu, sa preto ukazujú ako kľúčové riešenie. Táto architektúra dokáže flexibilne reagovať na rýchlo sa meniace podmienky, spracovávať dáta z rôznorodých zdrojov a protokolov a efektívne integrovať tieto údaje s analytickými a vizualizačnými nástrojmi. Výsledkom je dátová infraštruktúra schopná podporiť moderné IIoT aplikácie, ktoré ťažia z nepretržite aktualizovaných dát a využívajú ich na optimalizáciu výrobných procesov, prediktívnu údržbu a implementáciu pokročilých modelov umelej inteligencie.

3.2 Prenos dát v systéme a mimo neho

Spoľahlivý prenos dát je kľúčovým prvkom pre správne fungovanie systému. Z pohľadu komunikácie je IoT a IIoT agregáciou a spojením rôznych protokolov a typov sietí [16, 30, 13]. V rámci Priemyselného internetu vecí (IIoT) existuje množstvo technológií a komunikačných protokolov, ktoré umožňujú prepojenie priemyselných zariadení, strojov, senzorov a systémov. Tieto technológie zabezpečujú spoľahlivú a efektívnu komunikáciu medzi zariadeniami a systémami na rôznych úrovniach priemyselnej automatizácie. V tejto sekcii budú predstavené MQTT, OPC UA, Modbus TCP a AMQP.

MQTT

MQTT (Message Queuing Telemetry Transport) je ľahký binárny protokol založený na modeli publikovanie/odber (Publish/Subscribe), navrhnutý pre prenos malých objemov dát v sieťach s obmedzenými zdrojmi a vysokou latenciou. Operuje nad transportným protokolom TCP/IP a je optimalizovaný pre nízku šírku pásma a nízku spotrebu energie, čo ho robí ideálnym pre IIoT aplikácie [7].

Hlavné charakteristiky MQTT:

- Mechanizmus Publish/Subscribe: Klienti publikujú správy na určité témy *(topics)* a odoberajú správy z tém, ktoré ich zaujímajú. Broker sprostredkováva komunikáciu medzi publikujúcimi a odoberajúcimi klientmi.
- Kvalita služieb (QoS): MQTT definuje tri úrovne spoľahlivosti doručenia správ. Pri úrovni QoS 0 je správa odoslaná najviac jedenkrát bez potvrdenia doručenia, čo poskytuje najrýchlejší, ale najmenej spoľahlivý prenos. QoS 1 zabezpečuje doručenie správy aspoň jedenkrát s potvrdením prijatia, ale môže dôjsť k viacnásobnému doručeniu. QoS 2 garantuje doručenie presne jedenkrát pomocou dvojfázového potvrdzovacieho mechanizmu, čím poskytuje najvyššiu spoľahlivosť za cenu vyššej réžie.
- Minimalistická hlavička: Hlavička správy má len 2 bajty, čo znižuje réžiu prenosu.

MQTT je štandardizovaný v norme ISO/IEC 20922:2016 [2], ktorá definuje jeho syntax a sémantiku.

3.2.1 OPC UA

OPC UA (Open Platform Communications Unified Architecture) je súčasný štandard vyvinutý organizáciou OPC Foundation s cieľom podporiť interoperabilitu v priemyselnej automatizácii a príbuzných oblastiach. Predstavuje významné zlepšenie oproti predchádzajúcim OPC štandardom tým, že zjednocuje rôzne typy dát do jedného adresného priestoru a poskytuje konzistentný mechanizmus na ich prístup a manipuláciu [32].

Dôvody vývoja OPC UA

- Zjednotenie prístupu k dátam: Predchádzajúce OPC štandardy, ako OPC DA (Data Access), OPC HDA (Historical Data Access) a OPC A&E (Alarms & Events), fungovali nezávisle od seba, bez vzájomného prepojenia dát. OPC UA integruje tieto rôzne typy dát do jednotného adresného priestoru, čo umožňuje prepojenie aktuálnych hodnôt, historických dát a udalostí.
- Nové požiadavky: S rastúcou komplexnosťou priemyselných procesov vznikla potreba podpory nových funkcií, ako je prístup k historickým udalostiam, podpora viacerých hierarchií a implementácia metód a programov (tzv. príkazov). OPC UA umožňuje vyššiu úroveň modelovania dát, kde položky môžu byť typované a nie sú obmedzené len na jednoduché dátové typy.
- Technologická migrácia: Staršie OPC štandardy boli založené na technológiách Microsoft COM/DCOM, ktoré sú viazané na platformu Windows. S cieľom zabezpečiť platformovú nezávislosť a využiť moderné technológie, OPC UA definuje abstraktnú sadu služieb, ktoré môžu byť mapované na rôzne komunikačné protokoly, vrátane webových služieb.
- Rozšírenie oblastí aplikácie: OPC UA je navrhnuté tak, aby pokrývalo široké spektrum aplikácií od zariadení na výrobnej linke až po podnikové informačné systémy. Umožňuje jednotný a interoperabilný prístup k dátam naprieč celým podnikom. [20]

Architektúra OPC UA

Architektúra OPC UA je založená na servisne orientovanej architektúre (SOA), ktorá definuje abstraktné služby pre interakciu s adresným priestorom servera. Tento prístup umožňuje platformovú nezávislosť, pretože abstraktné služby môžu byť implementované na rôznych hardvérových a softvérových platformách. Flexibilné mapovanie na rôzne technológie umožňuje využitie služieb OPC UA s rôznymi komunikačnými protokolmi, ako je napríklad SOAP pre webové služby alebo binárne protokoly pre vyšší výkon. Rozšíriteľnosť a škálovateľnosť sú dosiahnuté možnosťou pridávať nové dátové typy a funkcie bez narušenia existujúcich systémov.

Jednou z kľúčových vlastností OPC UA je jednotný adresný priestor, ktorý integruje aktuálne dáta, historické dáta a udalosti či alarmy. Adresný priestor je štruktúrovaný pomocou uzlov a vzťahov medzi nimi. Uzly môžu reprezentovať zariadenia, senzory, premenné alebo komplexné objekty. Každý uzol má definovaný typ a môže obsahovať atribúty, ako sú názov, identifikátor či dátový typ, a referencie na iné uzly, čo umožňuje vytvárať hierarchie a siete. Tento model umožňuje detailnú digitálnu reprezentáciu reálnych systémov, čo zjednodušuje integráciu a interoperabilitu medzi rôznymi zariadeniami a aplikáciami.

OPC UA kladie veľký dôraz na bezpečnosť, najmä pri prístupe k citlivým priemyselným dátam cez nezabezpečené siete, ako je internet. Bezpečnostný rámec zahŕňa autentifikáciu

pomocou certifikátov na overenie identity klientov a serverov, autorizáciu na základe rolí a povolení, šifrovanie komunikácie na ochranu dát počas prenosu a zabezpečenie integrity dát, aby sa zaručilo, že dáta neboli počas prenosu pozmenené. Tieto mechanizmy zabezpečujú dôveryhodnú a bezpečnú komunikáciu medzi klientmi a servermi v rôznych sieťových prostrediach.

Pre organizácie využívajúce staršie OPC štandardy založené na technológiách COM/DCOM existuje niekoľko stratégií migrácie. Jednou možnosťou je použitie tzv. *wrapperov*, čo sú softvérové vrstvy umožňujúce komunikáciu starších OPC klientov s OPC UA servermi a naopak. Ďalšou stratégiou je paralelná prevádzka, kde sa OPC UA systémy nasadzujú vedľa existujúcich systémov s postupným prechodom na novú technológiu. Nakoniec, úplná migrácia zahŕňa prebudovanie aplikácií a systémov na plnú podporu OPC UA, čo prináša výhody platformovej nezávislosti a moderných funkcií [20, 35, 32].

3.2.2 Modbus TCP a Modbus RTU

Modbus je široko používaný komunikačný protokol vyvinutý spoločnosťou Modicon (teraz Schneider Electric) v roku 1979 pre komunikáciu s programovateľnými logickými automatmi (PLC). Vďaka svojej jednoduchosti, spoľahlivosti a otvorenosti sa stal de facto štandardom v priemyselnej automatizácii. Existuje niekoľko variantov protokolu Modbus, z ktorých najrozšírenejšie sú Modbus RTU (Remote Terminal Unit) a Modbus TCP. Hoci oba varianty zdieľajú rovnakú aplikačnú vrstvu, líšia sa v spôsobe fyzického prenosu dát a v komunikačných mechanizmoch [40, 39].

Modbus RTU

Modbus RTU je navrhnutý pre sériovú komunikáciu cez rozhrania RS-232, RS-422 alebo RS-485. Využíva kompaktný binárny formát dát a je optimalizovaný pre spoľahlivý a efektívny prenos v priemyselnom prostredí [29].

Kľúčové vlastnosti Modbus RTU:

- Komunikačné médium: Sériové linky (RS-232, RS-422, RS-485).
- Štruktúra správ: Binárne rámce s pevnou štruktúrou; správy sú oddelené časovými medzerami (tichými intervalmi).
- Architektúra Master/Slave: Jeden hlavný zariadenie (master) komunikuje s jedným alebo viacerými podriadenými zariadeniami (slave). Master iniciuje všetku komunikáciu.
- Adresovanie: Každé slave zariadenie má jedinečnú adresu v rozsahu od 1 do 247.
- Kontrola chýb: Využíva kontrolný súčet pomocou cyklickej redundančnej kontroly (CRC) na detekciu chýb.
- Modbus RTU je vhodný pre aplikácie, kde sú zariadenia umiestnené blízko seba a môžu byť prepojené sériovou komunikáciou. Často sa používa v priemyselných systémoch na komunikáciu medzi PLC a rôznymi senzormi či akčnými členmi.

Modbus TCP

Modbus TCP, tiež známy ako Modbus TCP/IP, je variant protokolu Modbus, ktorý využíva siete Ethernet a protokol TCP/IP pre komunikáciu. Umožňuje integráciu Modbus zariadení do moderných sieťových infraštruktúr a komunikáciu na väčšie vzdialenosti [28].

Kľúčové vlastnosti Modbus TCP:

- Komunikačné médium: Ethernetové siete využívajúce protokol TCP/IP.
- Štruktúra správ: Modbus správy sú zapuzdrené v TCP rámci; nie je potrebné riešiť časové medzery alebo štart/stop bity.
- Architektúra Client/Server: terminológia sa mení na klient-server, kde klient iniciuje komunikáciu a server odpovedá.
- Adresovanie: zariadenia sú identifikované pomocou IP adries. Pole Unit Identifier môže byť použité pri komunikácii cez brány (gateways).
- Kontrola chýb: spolieha sa na kontrolu chýb zabudovanú v TCP/IP protokole.
- Modbus TCP umožňuje jednoduchú integráciu s existujúcimi sieťami Ethernet a je ideálny pre distribuované systémy a aplikácie, kde je potrebná komunikácia na veľké vzdialenosti alebo integrácia s IT infraštruktúrou.

3.2.3 AMQP (Advanced Message Queuing Protocol)

AMQP je otvorený aplikačný protokol pre podnikové zasielanie správ, navrhnutý pre spoľahlivú, bezpečnú a interoperabilnú komunikáciu [31]. Operuje nad TCP/IP a podporuje modely výmeny správ typu Publish/Subscribe a Request/Response.

AMQP zahŕňa tieto vlastnosti:

- Zaručené doručenie správ: AMQP poskytuje spoľahlivé doručenie správ s rôznymi úrovňami potvrdení, čo zaisťuje, že správy sú doručené presne podľa požiadaviek aplikácie.
- Flexibilita komunikácie: umožňuje rôzne vzory zasielania správ a sieťové topológie, čo umožňuje prispôsobiť komunikáciu špecifickým potrebám rôznych aplikácií a systémov.
- Bezpečnostné mechanizmy: má zabudovanú podporu pre autentifikáciu a šifrovanie, čím zabezpečuje ochranu prenášaných dát a integritu komunikácie medzi klientmi a servermi.

V prostredí IIoT je AMQP využívaný tam, kde je potrebná vysoká spoľahlivosť a bezpečnosť pri prenose správ medzi zariadeniami a systémami. Jeho schopnosť zaručiť doručenie a flexibilne prispôsobiť komunikáciu ho robí vhodným pre kritické priemyselné aplikácie, ktoré vyžadujú robustnú a škálovateľnú infraštruktúru pre zasielanie správ. [30]

AMQP je štandardizovaný v norme ISO/IEC 19464:2014 [1], ktorá definuje jeho syntaktické a sémantické pravidlá, čím zabezpečuje interoperabilitu medzi rôznymi implementáciami protokolu. Ide o prepis pôvodnej špecifikácie od OASIS pod štandardom ISO/IEC.

3.3 Ukladanie a správa dát

Vzhľadom na charakter IIoT a na rýchlosť a objem generovaných údajov je potrebné zvoliť také dátové úložisko a architektúru, ktoré dokážu pracovať s veľkými množstvami dát v reálnom čase, prípadne takmer v reálnom čase. Dôraz je tiež kladený na škálovateľnosť, vysokú dostupnosť a zabezpečenie dát. Podľa [33] vieme popísať správu dát ako architektúry, spôsoby a procesy, ktoré sú nasadené za účelom správneho ukladania a správy dát v danom systéme. Konkrétne pre IoT a IIoT to musí byť ako medzivrstva medzi vecami, ktoré generujú dáta (ako senzory a iné zariadenia) a aplikáciami, ktoré tieto dáta používajú kvôli množstvu rôznorodých typov dát.

3.3.1 Požiadavky na úložisko dát v IIoT

Dáta môžu pochádzať z rôznych zdrojov, majú rôzne formáty a väčšinou sú generované v reálnom čase. Autori [33, 17, 42] sa zhodujú na niekoľkých oblastiach, ktoré je podľa nich dôležité brať na vedomie pri navrhovaní IoT/IIoT systému:

- **Výkonnosť a nízka latencia:** generované dáta v IIoT systémoch sú často spracovávané takmer okamžite, čo si vyžaduje rýchle vkladanie (write) aj čítanie (read) bez výraznej odozvy.
- **Škálovateľnosť:** s rastúcim počtom senzorov a zariadení narastá aj objem dát, ktoré je potrebné zvládnuť bez zníženia výkonnosti. Úložisko musí podporovať horizontálne alebo vertikálne škálovanie.
- **Spoľahlivosť a odolnosť voči výpadkom:** v priemyselnom prostredí je výpadok alebo strata dát potenciálne veľmi nákladná. Je potrebné zaistiť mechanizmy replikácie a vysokú dostupnosť [41].
- **Bezpečnosť:** pri ukladaní dát je nutné zabezpečiť ich dôvernosť, integritu a ochranu pred neoprávneným prístupom. To zahŕňa šifrovanie, autentifikáciu a autorizáciu.
- **Dátová štruktúra a formát:** v IIoT sa často vyžaduje uchovávanie historických časových radov, ako aj rôznorodých formátov (logy, metriky, binárne dáta z PLC a pod.).

3.3.2 Relačné databázy vs. NoSQL prístupy

V tradičnom priemyselnom prostredí sa na ukladanie dát zvyknú využívať relačné databázy (napríklad MySQL, PostgreSQL či MS SQL Server). V prostredí IIoT však tieto riešenia môžu narážať na limity a obmedzenia týchto relačných databáz, preto ich bolo potrebné adaptovať [22]. Preto sa čoraz častejšie presadzuje využívanie NoSQL databáz, ktoré ponúkajú:

- **Lepšiu horizontálnu škálovateľnosť:** dáta je možné rozdeľovať (shardovať) medzi viaceré uzly, čím sa zvyšuje spracovateľská kapacita [33].
- **Flexibilnejší dátový model:** v IIoT systémoch môžu byť dáta rôznorodé a nemusí existovať jednotná schéma. NoSQL riešenia umožňujú schema-less alebo len čiastočne definovanú štruktúru.

- **Vysokú priepustnosť (throughput):** často sa vyžaduje nepretržité a rýchle zapisovanie veľkého objemu údajov zo senzorov.

Napriek tomu relačné databázy nie sú v IIoT úplne vylúčené. Pri niektorých úlohách (napríklad integrácia s existujúcimi podnikovo-informačnými systémami alebo komplexné transakčné operácie) môže byť výhodné využívať tradičné SQL databázy, prípadne kombinovať oba prístupy (tzv. polyglot persistence).

3.3.3 Databázy časových radov

Keďže v IIoT systémoch dominujú dáta s časovou pečiatkou (napríklad merania, senzory, logy), špeciálnu kategóriu databáz predstavujú *time-series* databázy. Tieto databázy sú optimalizované na ukladanie a spracovanie veľkého množstva metrík, udalostí a meraní, ktoré sú časovo označené (tzv. **time series**) a poskytujú rýchle dotazy na agregácie a analýzu historických dát. Podľa [5] patrí medzi vlastnosti časovo orientovaných databáz kompresia dát, správa životného cyklu dát, zhrnutie dát, podpora rozsiahlych časových prehľadov a dopytov apod. Medzi populárne riešenia patria napríklad:

- **InfluxDB:** špecializovaná databáza určená na časové rady, ktorá ponúka jednoduchý dotazovací jazyk (Flux) a podporu pre kontinuálne dotazy a alerting.
- **TimescaleDB:** postavená na PostgreSQL, čo umožňuje využívať ekosystém SQL a zároveň ponúka optimalizované úložisko pre časové záznamy.

Výhodou time-series databáz je efektívne indexovanie a ukladanie veľkého počtu záznamov s minimálnou latenciou pri zápise. Okrem toho disponujú funkciami na špecifické dotazy (napr. downsampling, group by time interval), ktoré sú pre analytiku IIoT dát vhodné.

3.3.4 Architektonické vzory pre ukladanie dát v IIoT

Ukladanie dát v prostredí IIoT sa neobmedzuje len na jedno konkrétne technologické riešenie či databázu. V praxi sa čoraz častejšie uplatňuje kombinácia viacerých prístupov, pričom sa zohľadňuje potreba spracovať dáta v reálnom čase a zároveň archivovať a analyzovať historické záznamy. Pri navrhovaní architektúry úložiska je dôležité, aby sa vzal do úvahy spôsob, akým dochádza k spracovaniu dát (v cloude, na edge alebo vo vrstvách fog computingu) – časť predspracovania môže prebiehať na okraji siete, kde sa dáta filtrujú a dočasne ukladajú ešte pred ich presunom do centrálneho dátového úložiska [4].

Ďalším dôležitým faktorom je požiadavka na okamžitú odozvu spracovaných dát. Aplikácie, ktoré vyžadujú takmer okamžité rozhodovanie (napríklad pri sledovaní bezpečnostných hrozieb alebo v kontexte prediktívnej údržby), často využívajú priamu komunikáciu s *time-series* databázou alebo NoSQL riešením, aby dáta boli dostupné bez zbytočnej latencie [8]. Na druhej strane, systémy zamerané na dlhodobú analýzu a reportovanie môžu siahať po centralizovaných dátových skladištiach (data warehouses) či data lake platformách, kde sa dajú spracovať obsiahle historické záznamy a využívať pokročilé analytické nástroje [33].

Diverzita protokolov a formátov, ktoré sa v IIoT prostredí vyskytujú, je tiež limitujúcim faktorom pri návrhu architektúry uloženia dát. Keďže každé zariadenie môže komunikovať iným protokolom (OPC UA, Modbus, MQTT a pod.) a generovať rozličné typy dát (textové logy, binárne dáta, merania s časovou pečiatkou), architektúra musí zahŕňať integračné

a konverzné mechanizmy. Tie zabezpečujú, že dáta sú transformované do požadovaného formátu a ukladajú sa buď priamo do *time-series* databáz, do centrálnych skladov, alebo do iných typov úložísk [13, 17].

V niektorých prípadoch sa využíva priamy zápis dát do *time-series* databázy, čo umožňuje veľmi rýchle analyzovanie a vizualizáciu dát s časovou značkou. Tento prístup minimalizuje latenciu pri čítaní a zápise a je vhodný pre aplikácie, v ktorých je rozhodujúca okamžitá dostupnosť aktuálnych údajov. Naopak, scenáre, ktoré sa spoliehajú na dlhodobé uchovávanie rozsiahleho množstva historických dát a rozsiahlu analytiku, častejšie siahajú po centrálnych dátových skladoch či data lake platformách, kde sa dáta spracovávajú v dávkach a sú ľahko dostupné pre pokročilé analytické nástroje. Napríklad v [4] autori popisovali použitie databázy Apache Cassandra[2] na spracovanie veľkého množstva dát s čo najnižšou latenciou.

V praxi býva nemálo implementácií, ktoré kombinujú oba prístupy. Takýto hybridný model je známy aj ako *polyglot persistence*, pri ktorom sa rôzne typy dát ukladajú do databáz špecificky prispôsobených danému formátu a charakteru. Senzorické dáta a telemetria s vysokou frekvenciou zápisu sú tak často smerované do NoSQL alebo *time-series* systémov, zatiaľ čo štruktúrované údaje a metadáta nachádzajú svoje miesto v relačných SQL databázach. Týmto spôsobom sa síce využívajú výhody viacerých databáz, no zároveň sa zvyšuje komplexita a nároky na správu a koordináciu medzi nimi [33].

Z pohľadu IIoT je zrejmé, že zber, prenos a ukladanie dát tvoria ucelený reťazec, v ktorom každá zložka musí fungovať spoľahlivo a koordinovane. Vysoká frekvencia generovania údajov, potreba ich spracovania v reálnom čase či rôznorodosť protokolov a formátov zvyšujú nároky na technológiu aj architektúru riešenia. V praxi sa preto kombinuje viacero prístupov – niektoré systémy vyžadujú dávkové procesy typu ETL, inde je nutná nepretržitá podpora streamingovej komunikácie, a pri ukladaní sa často využívajú špecializované time-series databázy alebo hybridná *polyglot persistence*. Takýto prístup reflektuje rozmanitosť požiadaviek IIoT, kde je potrebné na jednej strane zaistiť minimálnu latenciu a vysokú dostupnosť, a na druhej strane zvládnuť dlhodobú archiváciu a analytiku veľkých objemov dát.

[2]Apache Cassandra je distribuovaná NoSQL databáza typu wide-column store, ktorá sa často využíva aj na ukladanie časových radov (time-series) v kontexte IIoT a spracovania dát vo veľkom objeme

Kapitola 4

Analýza požiadaviek na systém pre zber a odosielanie dát

Ako bolo uvedené v úvode 1, táto práca je robená v spolupráci so spoločnosťou AGEsoft s.r.o., ktorá prejavila záujem o riešenie. Pri definovaní požiadaviek bolo preto nevyhnutné zohľadniť rôznorodosť projektov v ich portfóliu, ako aj očakávania koncových používateľov. Títo používatelia očakávajú spoľahlivé a používateľsky komfortné riešenie, ktoré zabezpečí bezproblémový zber a spracovanie dát zo strojov, senzorov a iných periférií s dôrazom na nízku latenciu a jednoduchú integráciu do existujúcich výrobných procesov.

4.1 Analýza používateľov a ich potrieb

Analýzou a konzultáciami, ktoré prebiehali v spolupráci s AGEsoft s.r.o., bolo zistené, že cieľovou skupinou používateľov systému budú primárne programátori alebo integrátori služieb, ktorí majú skúsenosti s vývojom softvéru a prácou so sieťovými protokolmi, databázami a automatizačnými nástrojmi. Ich úlohou je navrhovať a implementovať systémy a riešenia priemyselnej automatizácie a monitorovania výrobných procesov – to zahŕňa návrh, konfiguráciu zariadení, testovanie a integráciu s ďalšími IT systémami zákazníka.

Na základe série konzultácií a systematickej analýzy procesov boli zozbierané požiadavky na navrhovaný systém. Konzultácie zahŕňali predovšetkým stretnutia s programátormi a systémovými integrátormi spoločnosti AGEsoft s.r.o., ako aj priame rozhovory s vybranými koncovými klientmi. Súčasťou analýzy bolo tiež skúmanie existujúcich dokumentácií technických riešení, sledovanie pracovných postupov v reálnych podmienkach a revízia architektúr aktuálne používaných systémov. Boli identifikované tieto základné potreby používateľov, ktoré je potrebné riešiť:

- **Podpora pre rôzne komunikačné protokoly:** systém musí umožniť zber a odosielanie dát prostredníctvom protokolov ako MQTT, OPC UA, REST API, Modbus a ďalších. To zabezpečí kompatibilitu s existujúcimi zariadeniami a budúce rozšírenia.
- **API na integráciu s externými nástrojmi:** rozhrania na integráciu s cloudovými službami, analytickými nástrojmi (napr. Grafana[1]) a vizualizačnými platformami.
- **Monitorovanie a diagnostika:** nástroje na monitorovanie stavu systému, sledovanie logov a diagnostiku chýb sú kľúčové pre zaistenie spoľahlivej prevádzky.

[1] Oficiálne stránky Grafan-y: `https://grafana.com`

- **Jednoduché nasadenie a správa:** riešenie by malo byť ľahko nasaditeľné na rôznych platformách (lokálne servery, edge zariadenia, cloud).
- **Škálovateľnosť:** systém musí byť schopný zvládnuť rastúci počet pripojených zariadení a objem spracovávaných dát bez výrazného dopadu na výkon.
- **Bezpečnosť:** zabezpečenie prenosu dát, autentifikácia používateľov a ochrana pred neoprávneným prístupom.

Navrhovaný systém bude používaný predovšetkým na zber, spracovanie a odosielanie dát zo zariadení v priemyselnom prostredí. Používatelia budú systém nasadzovať ako základnú vrstvu na integráciu rôznych senzorov, riadiacich jednotiek a výrobných liniek do centrálneho dátového systému. Používatelia najprv nakonfigurujú konektory a moduly podľa špecifikácií konkrétneho projektu. Po úvodnom nastavení bude systém kontinuálne prijímať dáta, vykonávať ich validáciu a filtrovanie a následne ich spracovávať podľa definovaných pravidiel. Výsledné údaje budú uložené v databázach alebo odoslané do externých nástrojov na vizualizáciu a analýzu.

4.2 Analýza súčasného prostredia a existujúcich riešení

Keďže sa jedná o veľmi aktuálne a kompetitívne prostredie, je dôležité pozrieť sa aj na už existujúce riešenia. Prieskum sa zameral na to, či nájdené riešenia spĺňajú vyššie definované požiadavky a navyše tieto body:

- **Nasadenie na edge prostredí / lokálne na mieste:** z aktuálneho konceptu vyplýva, že riešenie musí byť nasadené lokálne v prevádzke / továrni, kde je potrebné pracovať s IIoT zariadeniami.
- **Možnosť integrovať s lokálnymi službami:** napríklad *HMI* (rozhranie človek-stroj).

Z prieskumu vyplynulo, že nástroje na trhu často pokrývajú len časť týchto definovaných potrieb – niektoré umožňujú rýchle cloudové nasadenie, iné zasa poskytujú prepracovanú lokálnu integráciu. Väčšina existujúcich riešení je však zväčša viazaná na konkrétnu platformu (cloud, prípadne proprietárne riešenie daného výrobcu riadiacich systémov). Z týchto dôvodov môže byť integrácia do rôznych projektov a systémov zložitá a vyžaduje úpravy na mieru.

Výsledky prieskumu možno rozdeliť do dvoch väčších kategórií:

- **Cloudové služby**, ktoré poskytujú široké spektrum nástrojov a služieb na zber a spracovanie dát, avšak nasadenie v plne lokálnom prostredí je často obmedzené alebo dostupné len cez hybridné riešenia.
- **Korporátne / proprietárne riešenia**, ktoré sú určené na prevádzku v priemyselnom prostredí a ponúkajú robustnú, avšak uzavretú platformu s často veľmi nekompromisným licenčným modelom.

Existuje však aj jedna výnimka, kde ide o open-source nástroj s možnosťou fungovať lokálne aj v cloude a ľubovoľne ho rozširovať – *Node-RED*.

4.2.1 Cloudové riešenia

Azure IoT

Platforma **Microsoft Azure IoT**[2] poskytuje komplexný ekosystém služieb pre IoT a IIoT, vrátane Azure IoT Hub, IoT Edge a ďalších komponentov (napr. Azure Event Grid či Azure Stream Analytics). Umožňuje zber dát z rôznych zariadení pomocou protokolov ako MQTT, AMQP či HTTPS. Pre lokálne spracovanie je k dispozícii riešenie *IoT Edge*, ktoré dokáže bežať na edge zariadeniach a ponúka modularitu (tzv. edge moduly) na základné spracovanie dát priamo v továrni. Napriek týmto možnostiam je hlavným cieľom Azure IoT smerovanie dát do cloudu (Azure služby). Pre spoločnosti, ktoré potrebujú menej závislosti na verejných cloudoch alebo majú striktné bezpečnostné obmedzenia, môže byť integrácia s Azure IoT výzvou a často si vyžaduje hybridný prístup alebo privátny cloud (napr. Azure Stack).

AWS IoT Hub

AWS IoT Hub[3] (respektíve AWS IoT Core) predstavuje podobnú sadu služieb ako Azure. Umožňuje zber, spracovanie a spravovanie dát z IoT zariadení, pričom takisto ponúka nástroje na spracovanie priamo na edge zariadeniach cez *AWS IoT Greengrass*. Celé riešenie plynule zapadá do ekosystému AWS, najmä ak je potrebné rýchlo vyvíjať cloudové aplikácie pre analýzu či vizualizáciu. Podobne ako pri Azure, aj tu existuje možnosť nasadenia na edge v rámci *Greengrass*, avšak pre úplné lokálne fungovanie (bez cloudu) je často nutné doplniť alebo manuálne upraviť nástroje pre nasadenie. Pri projektoch, kde je už AWS prostredie bežne využívané, môžu byť výhody rýchleho prepojenia a zdieľania dát obrovské, no pre iné scenáre to môže byť príliš úzko viazané na AWS infraštruktúru.

4.2.2 Korporátne / proprietárne riešenia

Ignition by Inductive Automation

Pod označením **Ignition**[4] od spoločnosti Inductive Automation nájdeme jednu z najznámejších *SCADA*[5] platforiem, ktorá poskytuje flexibilné a modulárne prostredie na priemyselnú automatizáciu, zber a spracovanie dát. Ignition môže bežať buď na lokálnom serveri, vo virtuálnom prostredí, alebo na edge zariadeniach v rámci rozsiahlejších inštalácií. Riešenie ponúka možnosť prepojovať rôzne protokoly (napr. OPC UA, Modbus, MQTT), poskytuje nástroje na vývoj HMI a SCADA obrazoviek a disponuje funkciami na správu veľkého množstva zariadení.

- **Modulárny dizajn:** umožňuje pridať len tie moduly, ktoré sú potrebné, čo prispieva k optimalizácii nákladov a zjednodušeniu riešenia.
- **Ľahká škálovateľnosť:** je možné nasadzovať Ignition od menších projektov až po rozsiahle výrobné závody.
- **Možnosti integrácie:** rôzne komunikačné protokoly, databázy a externé služby.

[2] Oficiálne stránky Microsoft Azure IoT `https://azure.microsoft.com/en-us/solutions/iot`

[3] Oficiálne stránky AWS IoT `https://aws.amazon.com/iot/?nc2=h_ql_prod_it_iot`

[4] Oficiálne stránky Ignition `https://inductiveautomation.com/ignition/`

[5] SCADA (Supervisory Control and Data Acquisition) je systém umožňujúci diaľkové monitorovanie, zber a riadenie priemyselných procesov a infraštruktúr v reálnom čase.

Nevýhodou je, že Ignition je proprietárny softvér s licenčným modelom, vysokou cenovkou a niektoré pokročilé funkcionality či špecifické konektory môžu byť dostupné len prostredníctvom platených rozšírení.

Siemens Insights Hub / MindSphere

Siemens Insights Hub[6] (bývalá platforma **MindSphere**) predstavujú ďalší príklad proprietárneho ekosystému. Tieto riešenia umožňujú zber a spracovanie dát z inteligentných zariadení a výrobných liniek, pričom sú navrhnuté na prevádzku v priemyselných podnikoch s vysokými nárokmi na stabilitu. Siemens zároveň poskytuje kompletné portfólio cloudových služieb a analytických nástrojov, do ktorých je možné smerovať výrobné a prevádzkové dáta. Lokálne nasadenie je možné, ale spravidla vyžaduje nákup licencií a infraštruktúry priamo od Siemensu. Pre spoločnosti s už existujúcou inštaláciou Siemens riadiacich systémov, PLC a senzorov je Insights Hub lákavý, no integrácia do heterogénneho prostredia od rôznych dodávateľov môže byť náročná kvôli neochote spoločnosti Siemens podporovať protokoly tretích strán.

4.2.3 Výnimka: Node-RED

Node-RED[7] je open-source nástroj pôvodne vyvinutý v IBM, založený na jazyku JavaScript a platforme Node.js. Umožňuje jednoduché vytváranie tokov (tzv. *flows*) na zber, spracovanie a odosielanie dát pomocou širokej škály predpripravených uzlov (*node-ov*). Keďže je Node-RED open-source a dá sa nainštalovať na široké spektrum zariadení (vrátane edge platforiem, mikropočítačov a serverov), predstavuje flexibilnú alternatívu k vyššie spomenutým uzavretým systémom.

- Ponúka množstvo konektorov a uzlov pre rôzne priemyselné protokoly (Modbus, OPC UA, MQTT, REST API).
- Je ľahko rozšíriteľný – používatelia si môžu vytvárať vlastné uzly alebo využívať už vytvorené komunitou.
- Umožňuje prispôsobenie nasadenia na mieru, či už do lokálnej siete alebo do cloudu.

Nevýhodou býva chýbajúca vstavaná podpora pre pokročilú správu väčšieho počtu zariadení a robustné zabezpečenie na podnikovej úrovni (tieto funkcie je nutné doimplementovať alebo využiť externé nástroje).

Z prieskumu vyplýva, že na trhu je viacero riešení, ktoré sa venujú spracovaniu dát z IIoT zariadení. Pre úplné pokrytie požiadaviek však často chýba komplexné riešenie schopné fungovať úplne lokálne a zároveň pozostávať z modulárnych nástrojov a otvorených štandardov. Mnoho priemyselných podnikov tak stojí pred dilemou, či investovať do proprietárnych platforiem (ktoré poskytujú stabilný ekosystém s dlhodobou podporou), alebo zvoliť flexibilnejšie, open-source riešenia vyžadujúce viac interného know-how a vývoja. V závislosti od konkrétnych projektových požiadaviek a existujúcich investícií sa potom rozhodujú pre konkrétnu stratégiu nasadenia. Presne toto vytvára **„dieru na trhu“** – systémy, ktoré by dokázali ľahko obsluhovať rôzne protokoly, boli modulárne a zároveň bezpečné, no bez nutnosti viazanosti sa na konkrétneho dodávateľa či cloud.

[6] Oficiálne stránky Siemens Insights Hub `https://plm.sw.siemens.com/en-US/insights-hub/`
[7] Oficiálne stránky projektu Node-RED `https://nodered.org/`

4.2.4 Identifikovaná medzera na trhu a východisko pre návrh riešenia

Na základe:

- prieskumu trhu,
- konzultácií so spoločnosťou AGEsoft,
- doterajších skúseností s existujúcimi nástrojmi,
- a rozhovorov s cieľovou skupinou používateľov,

sme dospeli k záveru, že na trhu **chýba** ľahko prispôsobiteľné riešenie, ktoré by:

1. **Podporovalo širokú škálu protokolov** (MQTT, OPC UA, Modbus, REST atď.) a zároveň bolo jednoducho rozšíriteľné o ďalšie.
2. **Umožňovalo nasadenie v „edge“ prostredí** priamo vo výrobe (lokálny server alebo kompaktné priemyselné PC) a následné prepájanie s cloudom či inými externými službami len podľa potreby.
3. **Bolo modulárne a otvorené**, aby sa dalo ľahko integrovať s už existujúcimi softvérovými systémami a umožňovalo prispôsobovať internú logiku.
4. **Poskytovalo podnikové funkcie v oblasti bezpečnosti**, monitorovania a diagnostiky systému, ale bez nutnosti platiť veľké licenčné poplatky.
5. **Nezabúdalo na jednoduchú inštaláciu a používateľskú prívetivosť**, aby bolo vhodné aj pre menšie tímy a rýchle PoC (Proof of Concept) projekty.

Táto identifikovaná medzera na trhu zároveň formuje základné východisko pre návrh nového systému, ktorému sa ďalej venuje kapitola 5. Vychádzame pri tom z poznatkov o architektúrach z kapitoly 2 (Priemysel 4.0 a IIoT) a z princípov zberu, prenosu a ukladania dát z kapitoly 3. Doplnkovou motiváciou je aj dopyt po open-source riešeniach, ktoré umožňujú rýchle prototypovanie a dlhodobú udržateľnosť bez rizika „vendor lock-in“[8].

Napokon, rozhovory s odborníkmi z praxe potvrdili, že **kompatibilita a škálovateľnosť** riešenia predstavuje kľúčový atribút – ak raz systém “dorastie” do rozsiahlych inštalácií s tisíckami pripojených senzorov, musí byť schopný v reálnom čase spracúvať prichádzajúce dáta a bezpečne ich ukladať (viď aj kapitolu 3). Tieto požiadavky sa preto premietli do detailných špecifikácií, ktoré budú východiskom pre návrh a implementáciu výsledného systému.

[8] Vendor lock-in činí zákazníka závislým na službách a produktoch konkrétneho dodávateľa tým, že vytvára značné náklady a prekážky k prechodu k riešeniam iných dodávateľov.

Kapitola 5

Návrh architektúry riešenia

Táto kapitola predstavuje prechod od analýzy požiadaviek a existujúcich riešení k vlastnému návrhu systému, ktorý má adresovať hlavné zistené problémy a potreby (uvedené v kapitole 4). Z prieskumu trhu, rozhovorov s odborníkmi a interných diskusií v spoločnosti AGEsoft s.r.o. vyplynulo, že na trhu chýba modulárne a ľahko nasaditeľné riešenie schopné fungovať lokálne (tzv. edge nasadenie) i v hybridnom režime, pričom musí podporovať rôzne komunikačné protokoly (MQTT, OPC UA, Modbus a ďalšie), ponúkať rozhranie na integráciu s externými nástrojmi a mať dostatočnú škálovateľnosť a bezpečnosť.

Cieľom návrhu je teda vytvoriť systém, ktorý umožní zber a odosielanie dát z priemyselných zariadení a senzorov, ich spoľahlivú správu, prepojenie so systémami pre vizualizáciu a analytiku a zároveň bude použiteľný pre viaceré scenáre (od malých pilotných projektov až po robustnejšie nasadenia). Vzhľadom na rôznorodosť potrieb koncových používateľov a na základe analýz sme sa rozhodli pre modulárnu architektúru, ktorú je možné prispôsobovať rôznym komunikačným a prevádzkovým požiadavkám.

5.1 Koncept riešenia

Východiskom pri navrhovaní systému bolo vytvoriť modulárnu a flexibilnú architektúru, ktorá umožní ľahkú integráciu s rôznymi druhmi priemyselných a senzorových zdrojov dát a zároveň poskytne možnosť jednoduchého napojenia na externé analytické nástroje či cloudové služby. V kontexte požiadaviek na škálovateľnosť a nezávislosť jednotlivých častí sme sa rozhodli využiť koncept mikroslužieb (microservices), ktoré budú:

1. **Bežať v Docker kontajneroch** – zjednoduší sa tak nasadzovanie, verzovanie a prípadné horizontálne škálovanie[1].
2. **Implementované v modernom, staticky typovanom jazyku s multiplatformovou podporou** – vďaka čomu získame rozsiahlu podporu knižníc pre prácu so senzormi, priemyselnými protokolmi a databázami a možnosť využívať robustné nástroje ekosystému zvolenej platformy.
3. **Využívať gRPC** na vzájomnú komunikáciu medzi mikroslužbami – gRPC poskytuje vysokú priepustnosť a nízku latenciu pri volaniach medzi službami, čo je kľúčové pre systém spracúvajúci dáta v takmer reálnom čase[2].

[1] Web Docker-u: `https://www.docker.com`

[2] Google Remote Procedure Call: `https://grpc.io`

Navrhovaný koncept riešenia vychádza z potreby spojiť zber a spracovanie dát z priemyselných zariadení a senzorov so schopnosťou jednoduchej integrácie do existujúcich systémov a nástrojov pre analýzu či vizualizáciu. Keďže riešenie je určené pre výrobné prevádzky (tzv. *edge* prostredie) i hybridný model (kombinácia lokálnej inštalácie a vzdialených služieb), naším cieľom je, aby si používatelia mohli:

- nasadiť základný systém *on-premise* (napr. priamo vo výrobnej hale), kde zabezpečí zber dát s minimálnou latenciou,
- integrovať s externou analytickou platformou (v cloude alebo v korporátnej sieti) podľa potreby a kapacít,
- ľahko rozširovať funkcionalitu vďaka tomu, že každý nový prvok (napr. modul pre spracovanie obrazu, monitorovanie alarmov alebo HMI) pripoja len ako samostatnú mikroslužbu.

Každá mikroslužba beží v Docker kontajneri, čo umožňuje nasadzovať celý systém buď na jednom edge serveri (napríklad vo výrobnej hale) alebo distribuovane na viacerých fyzických či virtuálnych uzloch. To umožňuje aj modularitu (možnosť rozšíriť systém o ďalšie mikroslužby, napríklad pre spracovanie v reálnom čase alebo AI) a zlepšuje odolnosť systému ako takého (ak niektorý modul zlyhá, dá sa samostatne reštartovať bez prerušenia prevádzky). Princíp mikroslužieb reflektuje požiadavku na nízku latenciu, jednoduchú údržbu a nezávislé nasadzovanie, pričom každý tím môže spravovať vlastnú časť logiky oddelene. Takto navrhnuté riešenie je:

- flexibilné s novými zdrojmi dát (stačí vytvoriť alebo pridať nový adaptér do vstupného modulu),
- rozšíriteľné o ďalšie analytické mikroslužby – napr. pridať streamové spracovanie, AI modul alebo pridanie vlastného HMI,
- jednoducho prenositeľné medzi rôznymi prostrediami (test, výroba, cloud) vďaka Docker kontajnerom.

Pre spájanie jednotlivých častí (modulov) využívame gRPC na rýchlu a efektívnu komunikáciu typu request/response, prípadne pre streamovanie dát a udalostí. Počas vývoja a preskúmania potrieb systému sa ukázalo, že pre potreby komunikácie medzi jednotlivými modulmi v rámci navrhovanej architektúry postačuje robustný gRPC protokol a nie je potrebné zavádzať ďalšie komplexné riešenia ako RabbitMQ / AMQP. Tieto prístupy (primárne gRPC a interakcie s databázou) umožňujú zvládať *real-time* zmeny, taktiež asynchrónne spracovanie udalostí a v neposlednom rade zaručujú, že každá mikroslužba môže byť naprogramovaná s dôrazom na svoj špecifický účel bez toho, aby musela ‘vedieť všetko o systéme‘.

Voľba konkrétneho programovacieho jazyka a vývojovej platformy bude upresnená v implementačnej časti práce. Pre účely architektúry je však potrebné definovať, že zvolená technológia musí podporovať multiplatformové nasadenie, bohatý ekosystém knižníc pre prácu s IoT, priemyselnými protokolmi a databázami a dlhodobú podporu v enterprise prostredí.

5.2 Návrh architektúry

V návrhu architektúry sú aktuálne zadefinované tri kľúčové a povinné moduly (mikroslužby) plus webová aplikácia, pričom každá služba je navrhnutá pre fungovanie v kontajneri:

1. **Vstupný modul**

   - Úlohou je pripojiť sa k rôznym zdrojom dát (napr. PLC, senzory, Modbus, OPC UA, MQTT, prípadne priamo k SQL databáze) a získané dáta ukladať do vnútornej time-series databázy (TimescaleDB).
   - Obsahuje konektory alebo adaptéry pre dané protokoly a zabezpečuje transformáciu dát do jednotného formátu.
   - Komunikuje s ostatnými mikroslužbami (napr. s výstupným modulom) cez gRPC pre vykonávanie požiadavkov ako „zdroj pravdy“ pre systém.

2. **Databáza**

   - Systémové jadro pre ukladanie dátových záznamov s časovou pečiatkou zbieraných vstupným modulom a všetkých potrebných dát pre fungovanie systému.
   - Použitá bude TimescaleDB[3] – objektovo-relačná databáza založená na PostgreSQL databázi s rošírením a optimalizáciou práce s časovými značkami.
   - Umožňuje vysokú rýchlosť vkladania a zároveň efektívne dotazovanie veľkého množstva časových dát (napr. agregácie, downsampling).
   - Existovať môže aj vonkajšia databáza, ktorá sa bude cez výstupný modul synchronizovať pre prístup k dátam mimo siete prevádzky.

3. **Výstupný modul**

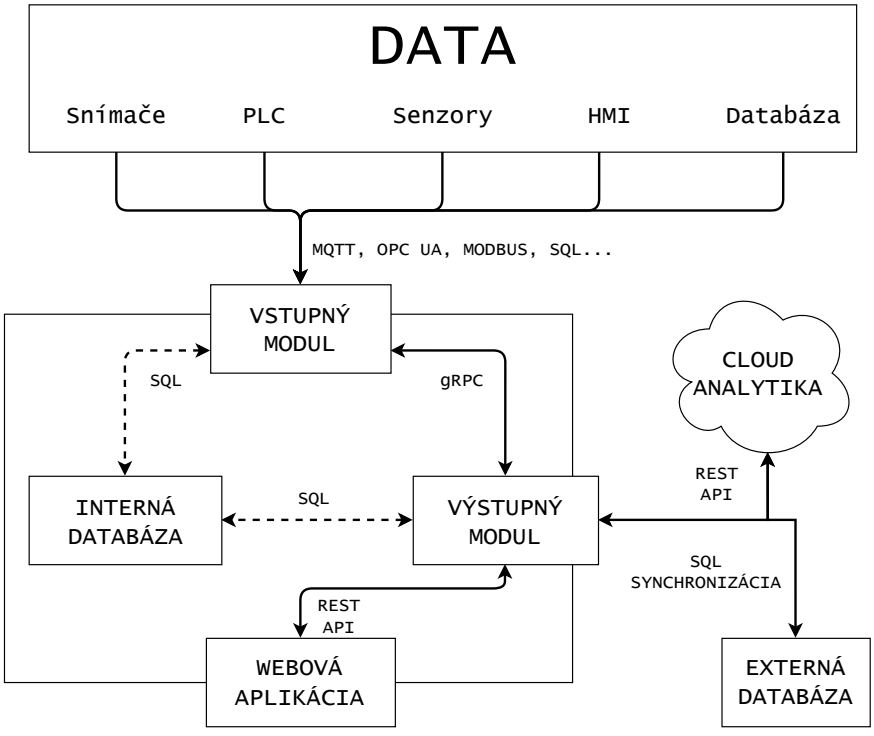
   - Zodpovedá za komunikáciu s externými službami, či už ide o cloudovú analytickú platformu, vzdialenú REST API službu alebo synchronizáciu s vonkajšou databázou (napr. korporátny dátový sklad).
   - Stará sa o správu modulov pomocou integrovanej webovej aplikácie, ktorá zobrazuje aktuálny stav systému a umožňuje jednotlivé moduly konfigurovať.
   - Prijíma dáta buď priamo z TimescaleDB (typicky cez SQL dotazy) alebo pomocou gRPC požiadavkov na vstupný modul.
   - Vie realizovať synchronizáciu – napríklad v pravidelných intervaloch skontrolovať stav vonkajšej databázy a aktualizovať vonkajšie záznamy.
   - Implementuje aj REST API pre podporu konfigurácie mimo siete prevádzky.

[3]Web databázy: `https://www.timescale.com`

4. **Webová aplikácia**

   - Postavená na Next.js 15 s TypeScriptom a Reactom pre moderné webové rozhranie[4].
   - Využíva SSR/ISR pre optimalizáciu výkonu.
   - UI komponenty budované s Tailwind CSS a shadcn/ui.
   - Komunikuje s výstupným modulom prostredníctvom REST API na načítanie aj odosielanie dát.
   - Obsahuje dashboard pre monitorovanie stavu adaptérov, vizualizáciu dát cez grafy (Recharts), konfiguráciu systému a správu používateľov.

Navrhnutá architektúra, zobrazená na 5.1, predstavuje komplexný, no zároveň modulárny systém, kde jednotlivé komponenty plnia jasne definované úlohy a komunikujú prostredníctvom štandardizovaných rozhraní. Vstupný modul zabezpečuje zber dát z rôznorodých zdrojov pomocou špecializovaných adaptérov, ktoré transformujú údaje do jednotného formátu a ukladajú ich do internej databázy. Výstupný modul slúži ako komunikačný most medzi vnútorným prostredím a externými systémami, poskytuje REST API pre webovú aplikáciu a umožňuje synchronizáciu s externými databázami. Webová aplikácia tvorí používateľské rozhranie celého systému. Táto architektúra umožňuje flexibilné nasadenie v rôznych prostrediach, jednoduchú údržbu a rozšíriteľnosť o nové zdroje dát či analytické funkcionality.



Obr. 5.1: Diagram konceptu architektúry systému

[4]Web Next.js: `https://nextjs.org`

5.3 Dátové štruktúry a entity

V rámci návrhu architektúry je dôležité definovať aj kľúčové dátové štruktúry, entity a ich vzájomné vzťahy. Dobre navrhnutý dátový model predstavuje základ celého riešenia a priamo ovplyvňuje nielen funkcionalitu, ale aj škálovateľnosť, rozšíriteľnosť a udržiavateľnosť systému v dlhodobom horizonte. Pri návrhu entít bol kladený dôraz na abstrakciu spoločných vlastností a flexibilitu pre budúce rozšírenia.

5.3.1 Základné entity

Základné entity predstavujú fundamentálne prvky dátového modelu, ktoré tvoria jadro informačnej štruktúry systému. Tieto entity sú navrhnuté tak, aby pokrývali všetky kľúčové aspekty zberu, spracovania a monitorovania dát. Dôležitým aspektom návrhu je správna granularita entít — dostatočne všeobecná pre flexibilitu, no zároveň dostatočne špecifická pre efektívnu implementáciu.

Senzorové dáta

Základnou entitou systému je model pre jednotné uchovávanie údajov zo senzorov. Tento model musí poskytovať:

- jednoznačnú identifikáciu senzora alebo meraného bodu,
- časovú značku merania,
- nameranú hodnotu v univerzálnom formáte,
- identifikáciu zdroja údajov,
- voliteľný doplňujúci popis.

Táto entita slúži ako spoločný formát pre údaje z rôznych zdrojov (MQTT, OPC UA, atď.), čo umožňuje ich jednotné spracovanie a ukladanie bez ohľadu na pôvodný protokol alebo formát. Jednotná reprezentácia senzorových dát je kľúčová pre efektívnu agregáciu a analýzu údajov z heterogénnych zdrojov. Navrhnutý model poskytuje štandardizovaný spôsob organizácie dát pre ďalšie spracovanie, čím výrazne zjednodušuje implementáciu analytických a vizualizačných nástrojov. Hodnoty sú transformované do konzistentného formátu hneď pri ich prijatí, čo eliminuje potrebu viacnásobných konverzií v neskorších fázach spracovania.

Systémové záznamy

Pre účely monitorovania a diagnostiky systém potrebuje entity na uchovávanie systémových udalostí a chybových hlásení. Tieto entity budú obsahovať:

- časovú značku udalosti,
- úroveň dôležitosti záznamu,
- textový popis udalosti,
- identifikáciu zdroja (komponenty systému),

- detaily o prípadnej chybe,
- doplňujúce diagnostické informácie.

V distribuovanom systéme, akým IIoT platforma bezpochyby je, predstavuje efektívne logovanie a monitorovanie kľúčový aspekt prevádzkovej stability. Navrhnutá entita poskytuje štruktúrovaný formát pre zaznamenávanie systémových udalostí s rôznymi úrovňami dôležitosti, od bežných informačných správ až po kritické chyby. Tento prístup umožňuje nielen retrospektívnu diagnostiku problémov, ale aj proaktívne monitorovanie stavu systému a včasnú detekciu potenciálnych problémov.

Synchronizačný proces

Pre zabezpečenie spoľahlivej synchronizácie údajov medzi internou a externou databázou je potrebná entita na sledovanie priebehu synchronizácie. Táto entita bude obsahovať:

- identifikátor synchronizačnej úlohy,
- typ synchronizovaných údajov,
- časové rozmedzie vykonania synchronizácie,
- aktuálny stav úlohy,
- štatistiky o počte synchronizovaných záznamov,
- informácie o prípadných chybách.

Synchronizácia dát medzi lokálnymi a vzdialenými systémami predstavuje komplexnú výzvu, najmä v podmienkach nestabilného pripojenia alebo obmedzenej šírky pásma. Navrhnutá entita poskytuje robustný mechanizmus pre monitorovanie a riadenie tohto procesu. Umožňuje nielen sledovať aktuálny stav synchronizácie, ale aj uchovávať historické záznamy o predchádzajúcich synchronizáciách, čo je kľúčové pre auditovateľnosť a diagnostiku. Kompletný záznam o každej synchronizačnej úlohe poskytuje transparentnosť a kontrolu nad procesom prenosu dát a pomáha identifikovať a riešiť potenciálne problémy, ako sú výpadky pripojenia alebo konflikty v dátach. Systém vďaka tomuto prístupu dokáže efektívne obnoviť synchronizáciu z bodu prerušenia bez nutnosti opätovného prenosu už synchronizovaných údajov.

5.3.2 Konfiguračný model

Systém bude používať hierarchický konfiguračný model, ktorý poskytuje:

- základnú konfiguráciu spoločnú pre všetky adaptéry,
- špecializované konfigurácie pre jednotlivé typy adaptérov s vlastnými špecifickými nastaveniami,
- možnosť dynamickej konfigurácie za behu systému.

Tento prístup umožňuje flexibilnú rozšíriteľnosť systému o nové typy adaptérov bez nutnosti zmeny existujúcich komponentov. Pre každý adaptér budú definované špecifické konfiguračné parametre potrebné pre jeho správne fungovanie, napríklad pripojovací reťazec, autentifikačné údaje alebo špecifické parametre protokolu.

Hierarchický konfiguračný model reprezentuje sofistikovaný prístup k správe nastavení v heterogénnom prostredí. Základná konfigurácia definuje spoločné parametre ako stav (povolený/zakázaný) a identifikáciu zdroja dát, zatiaľ čo špecializované konfigurácie rozširujú tento základ o parametre špecifické pre konkrétne protokoly.

Napríklad, MQTT adaptér vyžaduje konfiguráciu tém a možností spracovania JSON údajov, zatiaľ čo OPC UA adaptér potrebuje definíciu identifikátorov uzlov a intervalov publikovania a vzorkovania. Tento model nielen zjednodušuje vývoj nových adaptérov vďaka dedičnosti základných konfiguračných vlastností, ale zabezpečuje aj konzistentný prístup k správe konfigurácií naprieč celým systémom. Dynamická konfigurácia potom umožňuje zmenu parametrov počas behu systému bez nutnosti reštartovania komponentov, čo je kľúčové pre systémy s požiadavkou na vysokú dostupnosť.

Navrhnutý konfiguračný model úzko súvisí s architektúrou adaptérov, kde každý typ adaptéra implementuje rovnaké základné rozhranie, ale rozširuje ho o špecifickú funkcionalitu. Vďaka tomu možno jednoducho pridávať nové typy adaptérov bez nutnosti zmeny existujúcich komponentov. Tento prístup maximalizuje znovupoužiteľnosť kódu a minimalizuje duplicitu, čo významne prispieva k udržiavateľnosti a rozšíriteľnosti celého systému.

5.3.3 Tok dát v systéme

Dátový tok v systéme je navrhnutý v logických vrstvách, ktoré zodpovedajú hlavným fázam spracovania informácií:

Zber dát: Prvá vrstva zabezpečuje zber údajov z rôznych zdrojov prostredníctvom špecializovaných adaptérov. Adaptéry sa pripájajú k externým systémom (napr. MQTT broker, OPC UA server) a transformujú prijaté údaje do jednotného formátu pre ďalšie spracovanie.

Spracovanie a uloženie dát: V tejto vrstve sú zhromaždené údaje ďalej spracované, validované a uložené v časovej databáze. Systém bude podporovať transformáciu a normalizáciu údajov, filtrovanie a validáciu, efektívne ukladanie s optimalizáciou pre časové rady a automatickú správu životného cyklu údajov podľa konfigurovateľných politík.

Prístup k dátam: Táto vrstva poskytuje rozhranie pre prístup k uloženým údajom prostredníctvom API, ktoré umožňuje filtrovanie podľa rôznych kritérií (čas, senzor, zdroj), stránkovanie pre efektívny prístup k veľkému množstvu dát, získavanie štatistických údajov a agregácií a autorizovaný prístup k údajom z externých systémov.

Synchronizácia a integrácia: Posledná vrstva zabezpečuje prepojenie s externými systémami, zahŕňajúc synchronizáciu dát s externými databázami, integráciu s analytickými nástrojmi, výmenu konfiguračných nastavení medzi modulmi systému a poskytovanie dát pre vizualizáciu v používateľskom rozhraní.

Takto navrhnutý dátový tok zabezpečuje efektívne spracovanie údajov od ich získania zo senzorov až po ich prezentáciu používateľovi alebo ďalšie spracovanie v externých systémoch, s dôrazom na škálovateľnosť a spoľahlivosť celého procesu.

Kapitola 6

Implementácia navrhovaného riešenia

V tejto kapitole je popísaný spôsob implementácie navrhnutého systému so zameraním na realizáciu jednotlivých častí architektúry, použité technológie a riešenie špecifických technických výziev. Systém bol implementovaný v súlade s návrhom architektúry prezentovaným v predchádzajúcej kapitole, s dôrazom na modularitu, znovupoužiteľnosť a škálovateľnosť.

6.1 Prístup k implementácii a použité technológie

Pri implementácii bol zvolený modulárny prístup, kde jednotlivé časti systému majú jasne definované hranice a komunikačné rozhrania. Takýto prístup umožňuje nezávislý vývoj a jednoduché rozširovanie systému o nové funkcionality.

Pre implementáciu boli vybrané nasledujúce kľúčové technológie:

- **.NET 8** – moderná, multiplatformová implementácia frameworku .NET poskytujúca robustné základy pre vývoj backendových služieb
- **Entity Framework Core** – ORM (Object-Relational Mapping) framework pre prístup k databáze
- **TimescaleDB** – rozšírenie PostgreSQL optimalizované pre časové rady dát, ponúkajúce efektívne ukladanie a dotazovanie senzorových dát
- **gRPC** – vysokovýkonný framework pre volanie vzdialených procedúr, použitý pre komunikáciu medzi modulmi
- **Next.js 15** – React framework pre vývoj webových aplikácií s podporou renderingu na strane servera
- **TypeScript** – nadstavba JavaScriptu pridávajúca statickú typovú kontrolu
- **Tailwind CSS** – utility-first CSS framework pre rýchle vytváranie používateľského rozhrania
- **shadcn/ui** – kolekcia komponentov založených na Radix UI, poskytujúca konzistentný a prístupný dizajnový systém
- **React Query** – knižnica pre správu stavu a vykonávanie dotazov na strane klienta

- **Docker** – platforma pre kontajnerizáciu aplikácií umožňujúca jednoduché nasadenie a správu
- **Github Actions a Github Container Registry** – služby pre automatizovanie vytvárania, publikácie a ukladania jednotlivých obrazov kontajnerov pre rýchle nasadenie

Implementácia dátových štruktúr a entít navrhnutých v predchádzajúcej kapitole bola realizovaná s využitím Entity Framework Core ako ORM frameworku a TimescaleDB pre efektívne ukladanie časových radov. Senzorové dáta boli implementované prostredníctvom triedy `SensorData` pre prenos údajov medzi adaptérmi a rozšíriteľnej generickej triedy `TimeSerieEntity<TValue>`, ktorá umožňuje flexibilné ukladanie rôznych typov hodnôt. Systémové záznamy boli realizované triedou `LogEntity` s využitím TimescaleDB hypertabuliek pre optimálne vyhľadávanie a archiváciu. Pre synchronizačný proces bola vytvorená entita `SyncJobEntity`, ktorá sleduje priebeh, stav a výsledky synchronizačných úloh. Hierarchický konfiguračný model bol implementovaný pomocou dedičnosti, kde všetky špecifické konfigurácie adaptérov rozširujú základnú triedu `AdapterConfiguration`. Dátový tok bol realizovaný prostredníctvom asynchrónnych operácií s využitím vzoru udalostí (event pattern) pre komunikáciu medzi komponentmi a repozitárový vzor pre prístup k databáze.

6.2 Štruktúra projektu

Projekt je rozdelený do niekoľkých hlavných modulov (ich funkcie a tok dát sú viditeľné na obrázku 6.1), ktoré spolu vytvárajú kompletný systém:

6.2.1 Základná knižnica

Knižnica BP-IIoT_Core predstavuje základný stavebný kameň celého systému. Definuje spoločné koncepty, rozhrania, modely a služby, ktoré sú využívané ostatnými komponentmi. Táto knižnica zabezpečuje konzistenciu naprieč celým systémom a zjednodušuje vývoj nových funkcionalít. Hlavné súčasti zahŕňajú:

- **Dátové modely a entity** – definície tried ako `SensorData`, `SensorDataEntity`, `LogEntity`, ktoré tvoria základný dátový model systému.
- **Rozhrania** – definície služieb a rozhraní ako `IAdapter`, `IRepository`, `ILoggerService`, ktoré umožňujú modulárny návrh a jednoduchú výmenu implementácií.
- **Databázové kontexty** – triedy `TimeSerieDbContext` a `SystemDbContext` pre prácu s databázou prostredníctvom Entity Framework Core.
- **Základné služby** – implementácie služieb pre logovanie, správu konfigurácií, správu retenčných politík a synchronizáciu dát.
- **Protocol Buffers definície**[1] – kontrakty pre gRPC služby zabezpečujúce komunikáciu medzi modulmi.

V rámci Core knižnice bol implementovaný generický prístup s využitím funkcií jazyka C#, čo umožňuje flexibilnú a rozšíriteľnú implementáciu. Napríklad, pre entity časových

[1] Dokumentácia k Protocol Buffers: `https://protobuf.dev/`

radov bola vytvorená generická trieda, ktorá môže pracovať s rôznymi typmi hodnôt, poskytujúc tak flexibilitu pre budúce rozšírenia.

Pozornosť si zaslúži napríklad implementácia rozšírenej logovacej služby postavenej na knižnici Serilog, ktorá využíva koncept dávkového (batch) spracovania na optimalizáciu výkonu. Táto služba využíva `ConcurrentQueue` pre dočasné uchovávanie logov v pamäti, čím minimalizuje počet prístupov do databázy. Zápis do databázy nastáva automaticky pri dosiahnutí nakonfigurovanej veľkosti dávky alebo po uplynutí časového intervalu (pomocou `Timer`). Tento prístup umožňuje systému efektívne spracovávať veľké množstvo logov bez zbytočnej záťaže na databázu. Služba taktiež poskytuje rozšírené možnosti filtrovania a vyhľadávania logov podľa rôznych kritérií (časový rozsah, úroveň logovania, text správy), čo je kľúčové pre diagnostiku a monitorovanie systému. Implementácia kombinuje výhody poskyované knižnicou Serilog pre logovanie do súborov a konzoly s vlastným databázovým logovaním optimalizovaným pre časové rady.

Dôležitou súčasťou je aj vytvorenie migračných skriptov, ktoré zabezpečujú automatické vytváranie a aktualizáciu databázovej schémy. Špeciálnu pozornosť si zaslúžia migrácie pre TimescaleDB, ktoré transformujú štandardné PostgreSQL tabuľky na hypertabuľky a konfigurujú kompresiu a retenčné politiky, ktoré sa neskôr využívajú.

6.2.2 Adaptéry

Projekt BP-IIoT_Adapters implementuje konkrétne adaptéry pre rôzne komunikačné protokoly. Aktuálne zahŕňa implementácie pre MQTT a OPC UA protokoly (ktoré boli uprednostnené firmou AGEsoft s.r.o. kvôli aktuálnym projektom), pričom architektúra umožňuje jednoduché pridávanie ďalších adaptérov v budúcnosti.

Adaptér MQTT využíva knižnicu MQTTnet pre pripojenie k MQTT brokerom, odoberanie správ z definovaných tém a ich transformáciu do jednotného formátu. Implementácia umožňuje spracovanie správ dvoma základnými spôsobmi – buď priame prijímanie a ukladanie prijatých dát v surovom formáte, alebo konfigurovateľné parsovanie JSON údajov. Adaptér poskytuje mechanizmus pre základné spracovanie chýb, ktorý v prípade problémov s JSON parsovaním zabezpečí logovanie a fallback na surové dáta. Adaptér podporuje:

- pripojenie k MQTT brokerom pomocou TCP,
- odoberanie správ z viacerých tém definovaných v konfigurácii a oddelených bodkočiarkou,
- voliteľné spracovanie JSON payloadov s konfigurovateľným mapovaním polí,
- automatickú detekciu a konverziu základných dátových typov (Integer, Float, Date),
- nastavenie QoS (Quality of Service) úrovne AtLeastOnce pre doručovanie správ.

Adaptér OPC UA je implementovaný pomocou oficiálnej knižnice OPC Foundation a poskytuje implementáciu pre komunikáciu s priemyselnými zariadeniami podporujúcimi tento štandard. Adaptér umožňuje vytvorenie session a subscription pre jeden OPC UA server a monitorovanie definovaných uzlov. Pri detekovaní zmien hodnôt adaptér konvertuje prijaté údaje do jednotného formátu a propaguje ich pomocou udalostí. Adaptér poskytuje funkcionalitu pre:

- pripojenie k OPC UA serveru s vytvorením konfigurácie aplikácie klienta,

- monitorovanie definovaných uzlov špecifikovaných v konfigurácii,
- konfiguráciu publikačných intervalov (PublishingInterval) pre subscription,
- nastavenie vzorkovacích intervalov (SamplingInterval) pre monitorované položky,
- korektné ukončenie session a subscription pri zastavení adaptéra.

Jednotný mechanizmus udalostí (event model) je kľúčovou súčasťou implementácie adaptérov. Oba adaptéry využívajú event `DataReceived`, pomocou ktorého propagujú získané dáta do vstupného modulu. Tento event prenáša inštancie triedy `SensorData`, ktorá zapuzdruje všetky potrebné informácie o senzorových dátach bez ohľadu na ich pôvod. Tento mechanizmus umožňuje jednotné spracovanie dát z rôznych zdrojov a izoluje adaptéry od zvyšku systému.

Konfigurácia adaptérov bola implementovaná pomocou hierarchie tried, kde špecifické konfigurácie pre jednotlivé adaptéry dedia od základnej triedy `AdapterConfiguration`. Táto hierarchia umožňuje polymorfné správanie, kde systém môže pracovať s konfiguráciami bez znalosti ich konkrétneho typu. Systém automaticky serializuje a deserializuje konfigurácie do/z JSON formátu, čo umožňuje ich perzistentné ukladanie a načítavanie.

Oba adaptéry implementujú rovnaké základné rozhranie `IAdapter`, čo umožňuje ich jednotné spracovanie v systéme. Táto abstrakcia umožňuje systému pracovať s adaptérmi bez znalosti ich konkrétnej implementácie, čo prispieva k modulárnosti a rozšíriteľnosti riešenia. Architektúra adaptérov je navrhnutá podľa princípov SOLID[2], čo umožňuje jednoduché pridávanie nových typov adaptérov (napr. Modbus, REST API a SQL s ktorými sa v budúcnosti počíta) bez nutnosti úpravy ostatných častí systému.

6.2.3 Vstupný modul

Vstupný modul je implementovaný ako samostatná ASP.NET Core služba, ktorá je zodpovedná za zber dát z rôznych zdrojov prostredníctvom adaptérov. Implementácia využíva moderný prístup s minimálnym hostingom (minimal hosting) na báze WebApplication API v ASP.NET Core.

Kľúčové aspekty implementácie vstupného modulu zahŕňajú:

- **Vkladanie závislostí** – rozsiahle využitie tzv. *dependency injection* pre správu závislostí a konfiguráciu služieb, vrátane registrácie singletonov pre dlhotrvajúce objekty ako adaptéry a scoped služieb pre databázové operácie.
- **gRPC služba** – implementácia gRPC služby `AdapterConfigurationService` pre vzdialenú konfiguráciu a riadenie adaptérov, ktorá umožňuje výstupnému modulu a klientom komunikovať so vstupným modulom.
- **Dynamická správa adaptérov** – načítavanie a správa adaptérov je založená na konfiguráciách uložených v externých JSON súboroch, ktoré sa načítavajú pri štarte a môžu byť aktualizované počas behu.
- **Spracovanie senzorových dát** – implementácia `SensorDataHandler`, ktorý slúži ako event handler pre udalosti generované adaptérmi, čo umožňuje jednotné spracovanie dát bez ohľadu na ich zdroj.

[2]SOLID princípy objektovo orientovaného programovania: Single responsibility, Open-closed, Liskov substitution, Interface segregation, Dependency inversion

- **Migrácie databázy** – automatické detekcia a aplikovanie migrácií pre databázovú schému pri štarte aplikácie, zabezpečujúce konzistentnú štruktúru databázy.
- **Správa životného cyklu** – implementácia štartovacích rutín pre inicializáciu adaptérov a registráciu callbackov pre čisté ukončenie aplikácie a korektné odpojenie adaptérov.

Vstupný modul automaticky vytvára predvolené konfigurácie pre adaptéry pri prvom spustení, pokiaľ neboli vopred definované. Tieto konfigurácie zahŕňajú predvolené hodnoty pre pripojenie k MQTT brokerom a OPC UA serverom. Modul automaticky detekuje zmeny v konfigurácii a môže dynamicky reštartovať adaptéry pri zmene parametrov bez potreby reštartovania celej aplikácie.

Pre spracovanie dát je implementovaná stratégia, ktorá využíva vzor *fire-and-forget* pre asynchrónne spracovanie správ. `SensorDataHandler` používa `Task.Run` na asynchrónne spracovanie dát zo senzorov, čo umožňuje efektívnu manipuláciu s veľkým množstvom správ bez blokovania hlavného vlákna adaptéra. Na mapovanie senzorových dát z modelu do databázových entít sa používa knižnica AutoMapper, ktorá zjednodušuje transformáciu dát.

Vstupný modul taktiež implementuje systém logovania založený na Serilog, ktorý umožňuje detailné sledovanie činnosti adaptérov a diagnostiku prípadných problémov. Logy sú smerované súčasne na konzolu a do súborov s denným rolovaním, čo umožňuje jednoduché monitorovanie a archiváciu histórie operácií.

Konfigurácia adaptérov je perzistentne uložená prostredníctvom služby z Core projektu`ConfigurationFileManager`, ktorý zabezpečuje serializáciu a deserializáciu konfigurácií do a z JSON formátu. Toto umožňuje zachovanie nastavení medzi reštartami aplikácie.

6.2.4 Výstupný modul

Výstupný modul poskytuje REST API pre externé aplikácie a implementuje synchronizáciu dát s externými databázami. Je implementovaný ako ASP.NET Core Web API aplikácia s nasledujúcimi kľúčovými komponentmi:

- **REST API kontroléry** – implementácia API endpointov pre prístup k senzorovým dátam, logom, konfiguráciám adaptérov a systémovým nastaveniam.
- **gRPC klient** – implementácia klienta `AdapterConfigurationClientService` pre komunikáciu so vstupným modulom prostredníctvom gRPC.
- **Služba pre synchronizáciu dát** – implementácia `DatabaseSyncService` pre asynchrónnu synchronizáciu senzorových dát a logov s externými databázami.
- **Synchronizačná služba** – `DatabaseSyncBackgroundService` implementuje plánovanú synchronizáciu dát podľa nastaveného časového harmonogramu, ktorá beží na pozadí.
- **Správa retenčných politík** – implementácia `DataRetentionService` pre správu životného cyklu dát v TimescaleDB vrátane automatického čistenia historických dát.
- **Synchronizácia nastavení** – implementácia mechanizmu pre synchronizáciu zdieľaných nastavení (`appsettings`) medzi vstupným a výstupným modulom cez dedikované kontroléry.

Modul pracuje s dvoma rôznymi databázovými kontextami – `TimeSerieDbContext` pre časové rady senzorových dát a `SystemDbContext` pre systémové záznamy ako logy a informácie o synchronizačných úlohách. Pre efektívnu prácu s databázou v asynchrónnych scenároch modul využíva nielen štandardné DbContext inštancie, ale aj `DbContextFactory` implementácie, ktoré umožňujú vytvorenie nových inštancií kontextu aj v rámci dlhotrvajúcich asynchrónnych operácií, najmä počas procesu synchronizácie databáz.

Dôležitou súčasťou modulu je automatická aplikácia databázových migrácií pri štarte aplikácie. Tento proces zabezpečuje, že schéma databázy je vždy aktuálna a konzistentná s aktuálnou verziou aplikácie. Migrácie sa aplikujú oddelene pre časové rady (TimeSerieDbContext) a systémové dáta (SystemDbContext).

Synchronizácia s externou databázou je implementovaná ako dvojvrstvový proces:

- manuálne spúšťaná synchronizácia cez API endpointy,
- automatická plánovaná synchronizácia prostredníctvom background služby s konfigurovateľnými intervalmi (hodinovo, denne, týždenne).

API implementuje rozšírené možnosti filtrovania, stránkovania a zoraďovania, čo umožňuje efektívny prístup k dátam z externých aplikácií. Konfigurácia modulu je perzistentne uložená a môže byť aktualizovaná dynamicky počas behu aplikácie, bez potreby reštartu.

Modul poskytuje možnosť synchronizácie konfiguračných nastavení (`appsettings.json`) so vstupným modulom, čo zabezpečuje konzistenciu nastavení ako sú pripojenia k databáze, logovanie a politiky retencie dát medzi oboma modulmi. Táto synchronizácia môže prebiehať automaticky alebo byť iniciovaná manuálne cez API, pričom je možné skontrolovať stav synchronizácie a verzie konfigurácií. Implementácia zahŕňa mechanizmy pre bezpečné zálohovanie aktuálnych nastavení pred synchronizáciou.

6.2.5 Webová aplikácia

Používateľské rozhranie pre celý systém je sprostredkované pomocou webovej aplikácie a umožňuje operátorovi monitorovať a konfigurovať všetky aspekty tejto platformy. Aplikácia je implementovaná s využitím moderného technologického stacku:

- **Next.js 15** – React framework s podporou renderingu na strane servera, ktorý poskytuje vysoký výkon a optimalizáciu
- **TypeScript** – pre typovú bezpečnosť, lepšiu dokumentáciu kódu a efektívnejšiu podporu vývojových nástrojov
- **Tailwind CSS** – utility-first CSS framework pre rýchly vývoj konzistentného používateľského rozhrania
- **shadcn/ui** – komponenty postavené na Radix UI pre prístupný a moderný dizajn systému
- **React Query** – pre deklaratívnu správu stavu, cachovanie dát a automatické aktualizácie UI
- **Recharts** – flexibilná knižnica pre vizualizáciu dát pomocou interaktívnych grafov

Architektúra aplikácie je organizovaná podľa princípov Next.js App Router, čo umožňuje modulárnu štruktúru, efektívne rozdelenie kódu a optimalizáciu výkonu. Kód je logicky členený do priečinkov `/app` obsahujúceho stránky a layouty podľa URL štruktúry,

`/components` so znovupoužiteľnými React komponentmi a `/lib` so zdieľanými utilitami, typmi, hook-mi a API klientmi. Hlavná funkcionalita webovej aplikácie je rozdelená do štyroch kľúčových sekcií. Dashboard poskytuje prehľadovú obrazovku zobrazujúcu stav adaptérov (MQTT, OPC UA) a systému synchronizácie databáz, s rýchlym prístupom k hlavným funkciám.

Sekcia *Sensor Data* slúži na vizualizáciu a analýzu dát zo senzorov prostredníctvom konfigurovateľných tabuliek a grafov. Užívateľ môže filtrovať dáta podľa senzorov a časových období, prepínať medzi tabuľkovým a grafickým zobrazením a nastaviť automatické obnovovanie dát v reálnom čase. Systém tiež podporuje export a import konfigurácií dashboardov pre jednoduchú prenositeľnosť nastavení.

Sekcia *Logs* predstavuje pokročilý prehliadač systémových logov s možnosťami filtrovania podľa úrovne závažnosti (Trace, Debug, Info, Warning, Error, Critical) a zdroja (Input Module, Output Module). Používateľ môže definovať presný časový rozsah s presnosťou na minúty, vyhľadávať v obsahu logov pomocou fulltextového vyhľadávania, a zobraziť detailné informácie o chybových hláseniach vrátane výpisu zásobníka pre efektívne riešenie problémov.

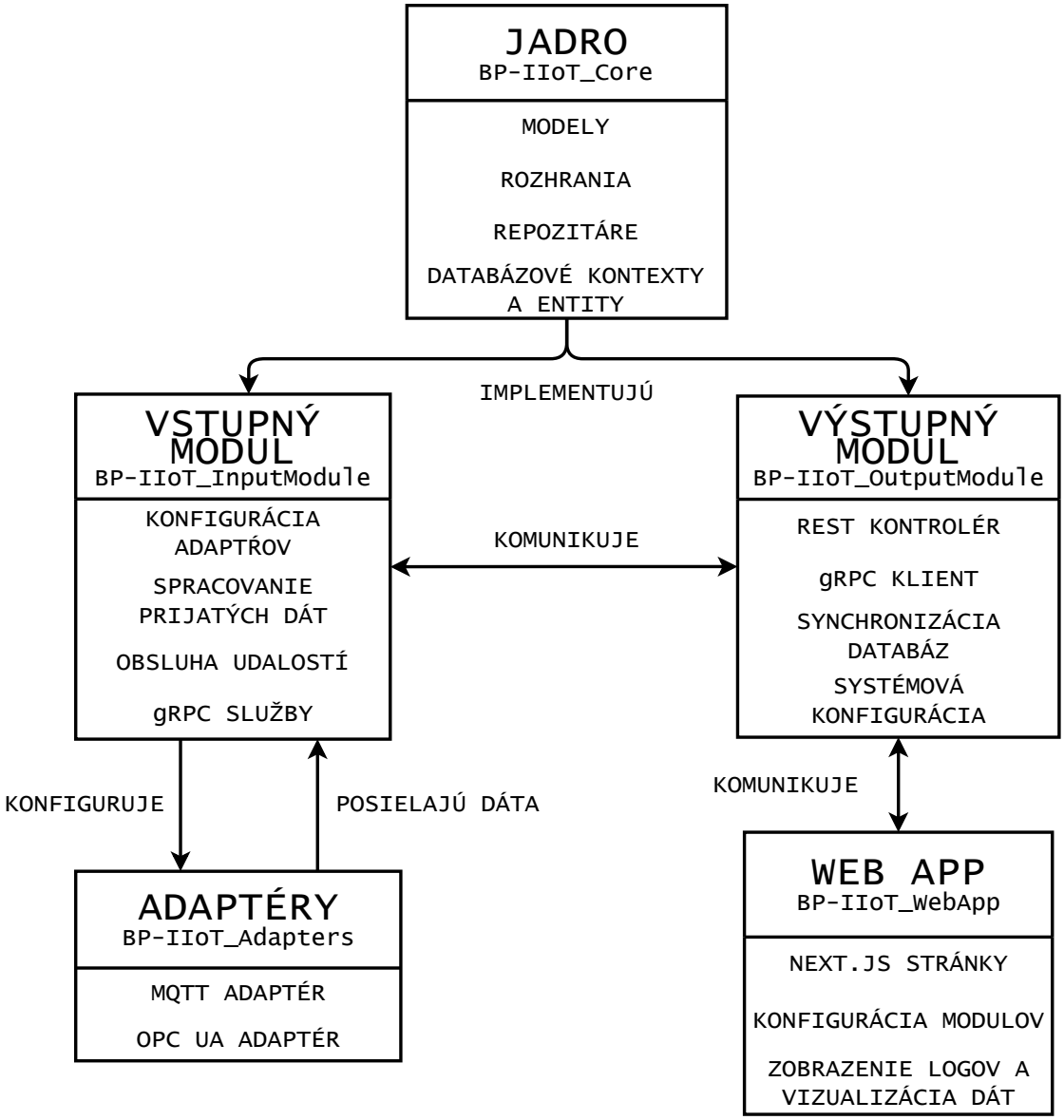
Konfiguračný systém pre všetky aspekty platformy je v časti *Settings*. Zahŕňa konfiguráciu adaptérov, zdieľané nastavenia pre databázu, logovanie a retenčné politiky, ako aj špecifické nastavenia pre vstupný a výstupný modul. Užívatelia môžu spravovať synchronizáciu konfigurácií medzi modulmi a nastavovať CORS a gRPC parametre pre sieťovú komunikáciu.

Pre efektívnu komunikáciu so serverovou časťou boli implementované typovo bezpeční API klienti organizovaní podľa doménových oblastí:

- `api/adapters.ts` – komunikácia s adaptérmi (MQTT, OPC UA),
- `api/logs.ts` – získavanie a filtrovanie systémových logov,
- `api/sensor-data.ts` – práca so senzorovými dátami vrátane ich transformácie,
- `api/settings.ts` – nastavenia systému,
- `api/database-sync.ts` – synchronizácia databáz.

Títo API klienti sú zabalení do React Query hooks, ktoré poskytujú automatické cachovanie dát pre zníženie záťaže API, indikátory načítania pre zlepšenie UX, správu chýb s notifikáciami používateľov a optimistické aktualizácie UI pre okamžitú spätnú väzbu.

Aplikácia je pripravená pre kontajnerizáciu pomocou Dockeru s multi-stage buildom, ktorý minimalizuje veľkosť výsledného obrazu. Produkčná konfigurácia je optimalizovaná pre výkon a bezpečnosť, vrátane behu pod neprivilegovaným používateľom, čo zvyšuje celkovú bezpečnosť nasadeného systému.



Obr. 6.1: Diagram jednotlivých častí systému

6.3 Kľúčové princípy implementácie

Pri celkovej analýze implementácie systému BP-IIoT možno identifikovať niekoľko vzájomne prepojených charakteristík a princípov, ktoré sa konzistentne uplatňujú naprieč všetkými modulmi. Tieto spoločné črty sú základom pre kvalitu a budúcu udržateľnosť celého riešenia.

6.3.1 Návrh systému a rozdelenie na moduly

Systém je charakteristický dôslednou aplikáciou princípu striktnej modularity, kde každý komponent má jasne definovanú zodpovednosť a presne špecifikované komunikačné rozhrania. Pri návrhu sme dbali na dodržiavanie osvedčených princípov (SOLID), čo znamená, že kód je lepšie organizovaný, ľahšie sa testuje a udržiava. Systém má vrstvenú štruktúru – oddelenú časť pre používateľské rozhranie, spracovanie dát a prácu s databázou. Vďaka tomu zmeny v jednej časti systému menej ovplyvnia tie ostatné.

6.3.2 Technologické princípy a postupy

Z technologického hľadiska je implementácia charakterizovaná využitím najnovších verzií frameworkov a knižníc (.NET 8, Next.js 15), čo zabezpečuje prístup k aktuálnym funkcionalitám a optimalizáciám. Systém konzistentne aplikuje asynchrónne programovacie vzory (async/await) naprieč všetkými komponentmi, čo umožňuje efektívne využitie výpočtových zdrojov a zabezpečuje responzívnosť aj pri spracovaní veľkého množstva súbežných operácií.

Jednotný prístup k správe dát sa prejavuje vo využití špecializovaných riešení pre časové rady a štandardizovaných postupov pre logovanie a diagnostiku. Pozoruhodným aspektom je zachovanie typovej bezpečnosti od backendu (C#) až po frontend (TypeScript), čo eliminuje celú kategóriu potenciálnych chýb a zjednodušuje refaktorovanie a údržbu kódu.

6.3.3 Vývojárske postupy

Z pohľadu vývojára implementácia aplikuje paradigmu vkladania závislostí (Dependency Injection) naprieč celým systémom, čo podporuje modulárny návrh a testovateľnosť. Pri vývoji sme najprv definovali rozhrania (čo má daná časť robiť) a až potom sme tvorili konkrétnu implementáciu (ako to má robiť). To umožňuje lepšie rozdeliť prácu a mať jasnejšie zodpovednosti.

Využitie generického programovania a návrhových vzorov ako Repository pattern predstavuje ďalší aspekt sofistikovaného prístupu k implementácii. Tieto postupy zabezpečujú vysokú mieru znovupoužiteľnosti kódu a zjednodušujú rozširovanie funkcionality systému.

6.3.4 Prevádzkové charakteristiky

Z prevádzkového hľadiska je systém implementovaný s dôrazom na jednoduchú nasaditeľnosť prostredníctvom kontajnerizácie všetkých komponentov. Všetky nastavenia systému sú na jednom mieste a dajú sa ľahko meniť, čo umožňuje prispôsobiť systém rôznym prostrediam bez nutnosti meniť a znovu kompilovať kód.

Automatizované databázové migrácie pri štarte aplikácie eliminujú potrebu manuálnych zásahov pri aktualizáciách schémy, čím sa redukuje riziko chýb a zjednodušuje proces nasadenia nových verzií. Konzistentný prístup k logovaniu a diagnostike naprieč celým systémom umožňuje efektívnu identifikáciu a riešenie prevádzkových problémov.

6.3.5 Komunikačné princípy

Komunikácia medzi komponentmi systému je postavená na jasne definovaných kontraktoch formalizovaných prostredníctvom Protocol Buffers, čo zabezpečuje typovú bezpečnosť a kompatibilitu. Vnútorná komunikácia využíva event-based prístup, ktorý minimalizuje synchronizačné závislosti medzi modulmi.

Implementácia uplatňuje API-First prístup s dôrazom na kvalitu a konzistentnosť aplikačných programových rozhraní, čo uľahčuje integráciu s externými systémami a prípadný budúci vývoj alternatívnych klientov.

6.4 Zhrnutie implementácie

Implementácia systému BP-IIoT predstavuje moderné a dobre premyslené riešenie pre priemyselný internet vecí (IIoT). Spája osvedčené postupy zo softvérového inžinierstva s aktuálnymi technológiami. Systém je navrhnutý ako súbor spolupracujúcich modulov, kde každý má svoju jasnú úlohu a spôsob komunikácie. To uľahčuje jeho postupný vývoj a pridávanie nových funkcií či adaptérov pre ďalšie zariadenia.

Silnou stránkou riešenia je jeho architektúra, ktorá dobre spája výkonnú serverovú časť (backend v .NET 8), optimalizovanú pre spracovanie dát, s moderným a rýchlo reagujúcim používateľským rozhraním (frontend v Next.js a React). Na komunikáciu medzi nimi sa používa rýchly protokol gRPC, čo zaisťuje nízke oneskorenie a dobrú priepustnosť dát. Vďaka efektívnemu využívaniu asynchrónnych operácií a optimalizovanej práci s databázou TimescaleDB dokáže systém spracovať aj veľké množstvá dát s rozumnými nárokmi na hardvér.

Unifikovaný prístup ku konfigurácii, logovaniu a riadeniu chybových stavov naprieč všetkými modulmi výrazne zjednodušuje údržbu a diagnostiku systému. Implementovaná podpora kontajnerizácie umožňuje jednoduché nasadenie v heterogénnych prostrediach – od lokálnych on-premise serverov až po distribuované cloudové infraštruktúry.

Vo výsledku, implementovaný systém BP-IIoT tvorí pevný základ pre zbieranie a analýzu dát z rôznych priemyselných zariadení a pre ich jednoduché prepojenie s firemnými informačnými systémami. Jeho modulárna stavba a dôraz na znovupoužiteľné komponenty ho robia prispôsobivým pre špecifické potreby rôznych priemyselných odvetví a technických prostredí.

Implementácia systému, ako bola popísaná v tejto kapitole, predstavuje konkrétne zhmotnenie navrhnutej architektúry a technologických rozhodnutí. Aby sme však mohli s istotou potvrdiť, že výsledné riešenie je nielen funkčné podľa požiadaviek, ale aj stabilné, efektívne a spoľahlivé v reálnych podmienkach, je nevyhnutné ho podrobiť dôkladnému overeniu. Práve tomuto kľúčovému kroku – testovaniu systému – sa bude venovať nasledujúca kapitola, ktorá popíše ako sme výsledný produkt testovali, použité nástroje a výsledky overovania jednotlivých častí systému BP-IIoT.

Kapitola 7

Testovanie systému

Testovanie je kritickou súčasťou vývoja softvéru, ktorá zabezpečuje, že výsledný produkt spĺňa požadované funkčné a nefunkčné požiadavky. V rámci vývoja systému BP-IIoT bol proces testovania naplánovaný a integrovaný do vývojového cyklu. Táto kapitola popisuje metodiku testovania a implementované testovacie scenáre.

Pre účely testovania počas vývoja bol v rámci riešenia vytvorený testovací projekt `BP-IIoT_Tests`, ktorý je integrovaný do hlavného projektu. Tento projekt poskytuje infraštruktúru pre rôzne typy testov a využíva moderné testovacie technológie a knižnice:

- **xUnit** – moderný testovací framework pre .NET
- **Moq** – knižnica pre mockovanie závislostí
- **FluentAssertions** – knižnica poskytujúca expresívnu syntax pre assertions
- **Microsoft.AspNetCore.Mvc.Testing** – nástroje pre testovanie ASP.NET Core aplikácií
- **Testcontainers** – knižnica umožňujúca automatické vytváranie Docker kontajnerov pre testovacie účely

Štruktúra testovacieho projektu je organizovaná do logických priečinkov pre jednotkové testy, integračné testy, testy API a celkové testy, reflektujúc modulárnu povahu samotného systému.

7.1 Implementované testy

7.1.1 Jednotkové testy

Jednotkové testy sa zameriavajú na overenie správnej funkcionality individuálnych komponentov systému v izolácii od ich závislostí. Pre systém BP-IIoT boli implementované základné unit testy konfiguračnej služby, ktoré overujú jej schopnosť korektne načítať a aktualizovať verziu konfigurácie zo súboru appsettings.json a schopnosť importovať zdieľané nastavenia. Tieto testy sú kľúčové, keďže verzia konfigurácie je základom pre synchronizáciu nastavení medzi modulmi.

7.1.2 Integračné testy

Integračné testy overujú spoluprácu medzi rôznymi komponentmi systému. Pre BP-IIoT boli implementované testy interakcie konfiguračnej služby so súborovým systémom a gRPC

komunikácie medzi modulmi. Test konfiguračnej služby overuje jej schopnosť manipulovať s fyzickými konfiguračnými súbormi v reálnom súborovom systéme. Test gRPC komunikácie verifikuje, že gRPC služba pre konfiguráciu adaptérov správne vracia nakonfigurované hodnoty pre jednotlivé adaptéry.

7.1.3 Testy API rozhrania

V rámci testov API boli implementované základné testy pre konfiguračné API, ktoré overujú získanie stavu synchronizácie a vykonanie synchronizácie. Tieto testy simulujú HTTP požiadavky a overujú, že odpovede obsahujú očakávané dáta a stavové kódy.

7.1.4 Infraštruktúra pre end-to-end testy

Pre komplexné testovanie systému ako celku bola vytvorená základná infraštruktúra pomocou Docker Compose, ktorá umožňuje vytvoriť testovacie prostredie obsahujúce všetky komponenty systému – PostgreSQL databázu, vstupný a výstupný modul.

7.2 Plánované testy

Z dôvodu prioritizácie implementácie funkčných častí systému neboli niektoré plánované testy zatiaľ realizované a budú súčasťou ďalšieho vývoja:

- **Rozšírené unit testy** – pre MQTT adaptér a spracovanie senzorových dát
- **Ďalšie integračné testy** – najmä pre synchronizáciu dát medzi databázami
- **Komplexné API testy** – pre adaptérové API a API senzorových dát, vrátane rôznych spôsobov filtrovania, stránkovania a zoraďovania
- **Komplexné end-to-end testy** – vrátane simulácie MQTT zariadení a overenia dátového toku

7.3 Spätná väzba zo spoločnosti AGEsoft

V priebehu vývoja a testovania systému BP-IIoT prebiehali pravidelné konzultácie so spoločnosťou AGEsoft s.r.o., ktorá poskytovala cennú spätnú väzbu. Implementácia bola hodnotená pozitívne, najmä z hľadiska modulárnej architektúry, ktorá umožňuje jednoduchú integráciu nových adaptérov.

Systém bol testovaný v simulovanom prostredí s použitím MQTT a OPC UA simulátorov, ktoré spoločnosť bežne využíva. Na testovacom projekte s niekoľkými zariadeniami systém fungoval spoľahlivo a podľa očakávaní. Spoločnosť AGEsoft realizovala vlastné výkonnostné testy, ktoré potvrdili schopnosť systému efektívne spracovávať údaje v reálnom čase. Ďalšie rozsiahlejšie výkonnostné testy budú prebiehať v nadchádzajúcich fázach projektu.

Konzultácie s vedúcim vývoja potvrdili, že systém spĺňa očakávania a je vhodný pre nasadenie v reálnych prevádzkach. Pozitívne hodnotená bola implementácia jednotného konfiguračného systému a možnosť dynamickej rekonfigurácie adaptérov bez nutnosti reštartu. Potvrdená bola aj kompatibilita systému s rôznymi hardvérovými platformami a operačnými systémami.

Spoločnosť ocenila implementáciu webového rozhrania, ktoré poskytuje intuitívne prostredie pre konfiguráciu a monitoring. Na základe pozitívnej spätnej väzby sa AGEsoft rozhodla nasadiť tento produkt do ostrej prevádzky v niekoľkých nadchádzajúcich projektoch.

7.4 Zhrnutie testovania systému

Testovanie systému BP-IIoT prebiehalo na viacerých úrovniach, so zameraním na kľúčové funkčné aspekty. Implementovaný testovací framework poskytuje dobrý základ pre ďalšie rozširovanie testovacích scenárov.

Základné unit testy potvrdili správnu implementáciu kľúčových komponentov, integračné testy overili ich spoluprácu a API testy verifikovali správnu implementáciu vybraných HTTP endpointov. Spätná väzba od spoločnosti AGEsoft potvrdzuje, že aj pri cielenom testovaní na vybraných scenároch systém plní požadované funkcie.

Na základe výsledkov testov a spätnej väzby možno konštatovať, že systém BP-IIoT spĺňa základné stanovené požiadavky a je vhodný pre nasadenie v testovacích prostrediach. Ďalšie testovanie bude prebiehať paralelne s vývojom dodatočných funkcionalít a prípravou systému na širšie nasadenie.

Kapitola 8

Návrh potenciálnych rozšírení

Ako každý softvérový systém, aj BP-IIoT poskytuje priestor pre ďalšie rozširovanie a vylepšovanie. V tejto kapitole identifikujeme a diskutujeme o potenciálnych smeroch budúceho vývoja, ktoré by mohli zvýšiť funkcionalitu, bezpečnosť, spoľahlivosť a použiteľnosť navrhnutého riešenia. Tieto návrhy vychádzajú z analýzy súčasného stavu systému, spätnej väzby od používateľov a identifikovaných limitácií počas implementácie a testovania.

8.1 Rozšírenie bezpečnostných mechanizmov

Súčasná implementácia systému sa zameriava primárne na funkčné aspekty a základnú integráciu komponentov. V produkčnom nasadení však bezpečnosť predstavuje kritický faktor, najmä pre systémy pracujúce s citlivými priemyselnými údajmi. Prvým významným rozšírením by preto malo byť zavedenie autentifikačného a autorizačného systému.

Implementácia používateľských účtov s rôznymi úrovňami oprávnení by umožnila granulárnu kontrolu prístupu k jednotlivým častiam systému. Administrátorské účty by mali plnú kontrolu nad konfiguráciou adaptérov a systémovými nastaveniami, zatiaľ čo účty s obmedzenými právomocami by mohli mať prístup iba k monitorovaniu a vizualizácii dát. Tento prístup by minimalizoval riziko neautorizovaných zásahov do systému, ktoré by mohli ohroziť jeho stabilitu alebo integritu dát.

8.2 Rozšírenie testovacej základne

Ako bolo uvedené v kapitole 7, súčasná implementácia zahŕňa základnú sadu testov, ktorá pokrýva kľúčové funkcie systému. Pre zaistenie vyššej kvality a stability v dlhodobom horizonte by však bolo vhodné výrazne rozšíriť testovaciu základňu. Toto rozšírenie by malo zahŕňať pokročilejšie unit testy pre všetky komponenty systému, integračné testy pre vzájomnú komunikáciu modulov a end-to-end testy simulujúce reálne používateľské scenáre.

Osobitná pozornosť by mala byť venovaná testovaniu adaptérov pre rôzne komunikačné protokoly, kde by sa mali simulovať rôzne chybové stavy a hraničné situácie, ako napríklad výpadky spojenia, neočakávané formáty správ alebo veľké objemy dát. Pre vstupný a výstupný modul by mali byť implementované záťažové testy overujúce škálovateľnosť a stabilitu pri spracovaní veľkého počtu požiadaviek.

Popri rozšírení manuálnych testov by bolo vhodné zaviesť automatizované testovanie v rámci CI/CD pipeline, ktoré by zabezpečilo kontinuálnu kontrolu kvality pri každej zmene

v kóde. Toto by nielen zlepšilo spoľahlivosť systému, ale aj urýchlilo vývojový cyklus a znížilo riziko zanesenia regresií pri implementácii nových funkcionalít.

8.3 Podpora pre dodatočné komunikačné protokoly

Súčasná implementácia systému sa zameriava na protokoly MQTT a OPC UA, ktoré pokrývajú významný segment trhu IIoT zariadení. Pre zvýšenie všestrannosti a použiteľnosti systému v širšom spektre priemyselných prostredí by však bolo vhodné rozšíriť podporu aj o ďalšie komunikačné protokoly.

Medzi prioritné rozšírenia by patrila implementácia adaptéra pre Modbus, ktorý je stále široko používaný v priemyselnej automatizácii, najmä v starších zariadeniach a systémoch. Ďalej by bolo vhodné pridať podporu pre protokol AMQP (Advanced Message Queuing Protocol), ktorý poskytuje pokročilé možnosti smerovania správ a je častou voľbou v enterprise prostredí. Pre zariadenia využívajúce webové technológie by bola prínosná implementácia adaptéra pre REST API alebo webhooky. Vhodné by bolo aj pridanie podpory rôznych proprietárnych protokolov, ako napr. S7 využívaný v zariadeniach Siemens, ktoré majú významné zastúpenie v priemyselnom sektore.

8.4 Vylepšenie používateľského rozhrania a dashboardu

Aktuálny dashboard poskytuje základnú funkcionalitu pre monitorovanie a konfiguráciu systému. Pre zlepšenie používateľskej skúsenosti a poskytnutie komplexnejšieho prehľadu o stave systému by bolo vhodné implementovať rozšírenia zamerané na vizualizáciu a analýzu dát. Prvým krokom by mohlo byť vytvorenie konfigurovateľných widgetov pre dashboard, ktoré by umožnili používateľom prispôsobiť rozhranie podľa vlastných potrieb a priorít. Tieto widgety by mohli zahŕňať grafy zobrazujúce trendy v senzorových dátach, prehľady o stave jednotlivých adaptérov alebo alarmy upozorňujúce na prekročenie definovaných prahových hodnôt.

Ďalším vylepšením by mohlo byť rozšírenie analytických možností s implementáciou základných štatistických funkcií priamo v používateľskom rozhraní. Toto by zahŕňalo výpočet korelácie medzi rôznymi senzormi, detekciu anomálií v dátach alebo prediktívnu analýzu založenú na historických trendoch.

Pre zlepšenie operatívneho prehľadu by bolo užitočné implementovať systém notifikácií, ktorý by mohol upozorňovať na dôležité udalosti ako výpadky komunikácie, prekročenie kapacitných limitov alebo kritické chyby v systéme. Tieto notifikácie by mohli byť doručované prostredníctvom webového rozhrania, e-mailov alebo integrácie s existujúcimi komunikačnými platformami.

8.5 Zhrnutie budúceho vývoja

Navrhnuté rozšírenia a vylepšenia predstavujú strategický plán pre ďalší vývoj systému BP-IIoT, ktorý by zabezpečil jeho dlhodobú udržateľnosť, konkurencieschopnosť a užitočnosť v dynamicky sa meniacom prostredí priemyselného internetu vecí. Prioritizácia týchto vylepšení by mala vychádzať z konkrétnych potrieb používateľov a spätnej väzby z reálnych nasadení, pričom implementácia by mala prebiehať inkrementálne s dôrazom na zachovanie stability a kompatibility s existujúcimi inštaláciami.

V súlade s modulárnou architektúrou systému by každé rozšírenie malo byť implementované ako samostatný, dobre definovaný komponent, ktorý možno integrovať do existujúceho ekosystému bez narušenia jeho funkcií. Tento prístup by nielen uľahčil postupný vývoj, ale aj umožnil zákazníkom adoptovať len tie vylepšenia, ktoré zodpovedajú ich konkrétnym potrebám a prioritám.

Kapitola 9

Záver

Cieľom tejto práce bolo navrhnúť, implementovať a overiť funkčnosť modulárneho systému na zber a spracovanie dát v prostredí priemyselného internetu vecí (IIoT). Zámerom bolo vytvoriť systém, ktorý je znovupoužiteľný, modulárny a škálovateľný, umožňujúci jednoduchú integráciu do existujúcich priemyselných riešení. Návrh bol koncipovaný s dôrazom na flexibilitu a prispôsobiteľnosť rôznym výrobným a monitorovacím scenárom.

Výsledkom práce je implementovaný systém BP-IIoT, ktorý pozostáva z niekoľkých kľúčových komponentov: vstupného modulu pre zber dát z rôznych zdrojov, výstupného modulu pre komunikáciu s externými službami a konfiguráciu systému, časovej databázy TimescaleDB pre efektívne ukladanie senzorových dát a webovej aplikácie pre monitorovanie a správu celého systému. Systém je postavený na architektúre mikroslužieb, kde každá služba beží v samostatnom Docker kontajneri, čo umožňuje jednoduché nasadenie, škálovanie a nezávislú aktualizáciu jednotlivých komponentov.

Z technologického hľadiska sa podarilo úspešne implementovať moderné prístupy a technológie. Backend služby sú vyvinuté na platforme .NET 8, ktorá poskytuje vynikajúcu výkonnosť a multiplatformovú podporu. Komunikácia medzi modulmi je zabezpečená prostredníctvom gRPC, čo umožňuje efektívnu a typovo bezpečnú výmenu dát. Pre ukladanie časových radov bola zvolená databáza TimescaleDB, ktorá poskytuje optimálny výkon pri práci s veľkým množstvom senzorových dát. Používateľské rozhranie je implementované pomocou frameworku Next.js 15 s TypeScriptom, čo zabezpečuje responzívnu a intuitívnu webovú aplikáciu.

V súčasnosti systém podporuje adaptéry pre protokoly MQTT a OPC UA, ktoré patria medzi najrozšírenejšie komunikačné štandardy v priemyselnom prostredí. Modulárna architektúra však umožňuje jednoduché pridávanie ďalších adaptérov pre rôzne protokoly a zdroje dát. Implementovaný je tiež mechanizmus na synchronizáciu dát medzi lokálnou a externou databázou, čo umožňuje flexibilnú integráciu s podnikovými informačnými systémami.

Funkčnosť systému bola overená v spolupráci so spoločnosťou AGEsoft s.r.o., ktorá poskytla cenné pripomienky a umožnila testovanie v simulovanom prostredí s použitím MQTT a OPC UA simulátorov. Na základe pozitívnej spätnej väzby sa spoločnosť rozhodla systém ďalej rozvíjať a nasadiť ho v reálnych prevádzkach. Aktuálne sú naplánované tri rôzne nasadenia, pričom jedno z nich bude v USA na linke na spracovanie oceľových rúr. Toto praktické využitie potvrdzuje, že navrhnutý systém spĺňa požiadavky priemyselného prostredia a má potenciál pre širšie nasadenie.

V prílohe tejto práce sú zahrnuté fotografie z reálnych nasadení implementovaného systému BP-IIoT, ktoré dokumentujú jeho praktické využitie v priemyselnom prostredí. Na

obrázkoch A.1 až A.4 je zachytené nasadenie systému v USA na linke na spracovanie oceľových rúr, kde je možné vidieť kompletnú výrobnú infraštruktúru, vrátane dopravníkového systému, ovládacieho panelu, B&R priemyselného počítača slúžiaceho ako server pre beh systému a elektrického rozvádzača s riadenými PLC. Obrázok A.5 zachytáva staršie priemyselné nasadenie, pre ktoré sa plánuje implementácia systému BP-IIoT v blízkej budúcnosti.

Táto práca bola tiež prezentovaná na študentskej vedeckej konferencii Excel@FIT 2025, kde bol predstavený návrh a implementácia systému BP-IIoT. Príspevok získal pozitívnu spätnú väzbu od účastníkov konferencie i od ľudí z profesionálnej oblasti, čo potvrdzuje relevantnosť a kvalitu navrhnutého riešenia v kontexte aktuálnych trendov v oblasti priemyselného internetu vecí.[1]

Pre budúci vývoj systému boli identifikované viaceré možnosti rozšírení a vylepšení. Medzi prioritné oblasti patrí posilnenie bezpečnostných mechanizmov implementáciou komplexného autentifikačného a autorizačného systému, rozšírenie testovacej základne o pokrytie všetkých komponentov systému, pridanie podpory pre ďalšie komunikačné protokoly (Modbus, AMQP, REST API) a vylepšenie používateľského rozhrania o konfigurovateľné widgety a rozšírené analytické možnosti. Tieto vylepšenia budú implementované v nadchádzajúcich verziách systému v súlade s potrebami konkrétnych nasadení a spätnou väzbou z reálnych prevádzkových prostredí.

Systém BP-IIoT predstavuje komplexné a flexibilné riešenie pre zber a spracovanie dát v prostredí priemyselného internetu vecí. Jeho modulárna architektúra, využitie moderných technológií a dôraz na škálovateľnosť a výkonnosť ho predurčujú na úspešné nasadenie v rôznych priemyselných sektoroch. Práca tak prináša nielen teoretický príspevok k oblasti IIoT, ale aj prakticky využiteľný produkt s reálnym potenciálom zlepšiť efektivitu a produktivitu v priemyselných procesoch.

[1] Odkaz na abstrakt príspevku: `https://excel.fit.vutbr.cz/submissions/2025/067/67.pdf`

Literatúra

[1] 1, I. J. *Information technology — Advanced Message Queuing Protocol (AMQP) v1.0 specification*. International Organization for Standardization, 2014. Dostupné z: https://www.iso.org/standard/64955.html.

[2] 1, I. J. *ISO/IEC 20922:2016, Information technology — Message Queuing Telemetry Transport (MQTT) v3.1.1*. International Organization for Standardization, 2016.

[3] Ahmed, E.; Yaqoob, I.; Gani, A.; Imran, M. a Guizani, M. Internet-of-things-based smart environments: state of the art, taxonomy, and open research challenges. *IEEE Wireless Communications*, 2016, zv. 23, č. 5, s. 10–16.

[4] Alabadi, M.; Habbal, A. a Wei, X. Industrial Internet of Things: Requirements, Architecture, Challenges, and Future Research Directions. *IEEE Access*, 2022, zv. 10, s. 66374–66400.

[5] An dong, S. a Fang, Z. Research on Open Source Solutions of Data Collection for Industrial Internet of Things. In: *2021 7th International Symposium on Mechatronics and Industrial Informatics (ISMII)*. 2021, s. 180–183.

[6] Baheti, R. a Gill, H. Cyber-physical systems. *The impact of control technology*, 2011, zv. 12, č. 1, s. 161–166.

[7] Banks, A. a Gupta, R. *MQTT Version 3.1.1* http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html. OASIS, October 2014. Dostupné z: http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html. OASIS Standard.

[8] Bonomi, F. a Milito, R. Fog Computing and its Role in the Internet of Things. *Proceedings of the MCC workshop on Mobile Cloud Computing*, August 2012.

[9] Boyes, H.; Hallaq, B.; Cunningham, J. a Watson, T. The industrial internet of things (IIoT): An analysis framework. *Computers in Industry*, 2018, zv. 101, s. 1–12. ISSN 0166-3615. Dostupné z: https://www.sciencedirect.com/science/article/pii/S0166361517307285.

[10] H. Kagermann, W.-D. L. a Wahlster, W. 'Industrie 4.0: Mit dem internet der Dinge auf dem Weg zur 4. Industriellen revolution. *VDI nachrichten, 13, 2-3* online. 2011. Dostupné z: https://www.ingenieur.de/technik/fachbereiche/produktion/industrie-40-mit-internet-dinge-weg-4-industriellen-revolution/. [cit. 2024-11-19].

[11] Hermann, M.; Pentek, T. a Otto, B. Design Principles for Industrie 4.0 Scenarios. In: *2016 49th Hawaii International Conference on System Sciences (HICSS)*. 2016, s. 3928–3937.

[12] J, J. Industrial Internet of Things (IIoT) – An IoT Integrated Services for Industry 4.0 : A Review. *International Journal of Applied Science Engineering*, Jún 2020, zv. 8.

[13] Jaloudi, S. Communication Protocols of an Industrial Internet of Things Environment: A Comparative Study. *Future Internet*, 2019, zv. 11, č. 3. ISSN 1999-5903. Dostupné z: https://www.mdpi.com/1999-5903/11/3/66.

[14] Jamaludin, J. a Rohani, J. M. Cyber-Physical System (CPS): State of the Art. In: *2018 International Conference on Computing, Electronic and Electrical Engineering (ICE Cube)*. 2018, s. 1–5.

[15] Kagermann, H.; Helbig, J.; Wahlster, W. a Hellinger, A. *Recommendations for Implementing the Strategic Initiative INDUSTRIE 4.0: Securing the Future of German Manufacturing Industry ; Final Report of the Industrie 4.0 Working Group.* Forschungsunion, 2013. Dostupné z: https://books.google.cz/books?id=AsfOoAEACAAJ.

[16] Khodadadi, F.; Dastjerdi, A. a Buyya, R. Chapter 1 - Internet of Things: an overview. In: Buyya, R. a Vahid Dastjerdi, A., ed. *Internet of Things*. Morgan Kaufmann, 2016, s. 3–27. ISBN 978-0-12-805395-9. Dostupné z: https://www.sciencedirect.com/science/article/pii/B9780128053959000010.

[17] Kuzlu, M.; Kalkavan, H.; Gueler, O.; Zohrabi, N.; Martin, P. J. et al. An End to End Data Collection Architecture for IoT Devices in Smart Cities. In: *2022 IEEE Power Energy Society Innovative Smart Grid Technologies Conference (ISGT)*. 2022, s. 1–5.

[18] Lasi, H.; Fettke, P.; Kemper, H.-G.; Feld, T. a Hoffmann, M. Industry 4.0. *Business & information systems engineering*. Springer, 2014, zv. 6, s. 239–242.

[19] Lee, J.; Bagheri, B. a Kao, H.-A. A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems. *Manufacturing Letters*, 2015, zv. 3, s. 18–23. ISSN 2213-8463. Dostupné z: https://www.sciencedirect.com/science/article/pii/S221384631400025X.

[20] Leitner, S.-H. a Mahnke, W. OPC UA – Service-oriented Architecture for Industrial Applications. *Softwaretechnik-Trends Band 26, Heft 4*. Bonn: Geselllschaft für Informatik e.V., 2006.

[21] Liu, Y.; Peng, Y.; Wang, B.; Yao, S. a Liu, Z. Review on cyber-physical systems. *IEEE/CAA Journal of Automatica Sinica*, 2017, zv. 4, č. 1, s. 27–40.

[22] Lou, Y.; Qin, Y.; Ye, F.; Zhang, P. a Chen, Y. Hydrological stream data pipeline framework based on IoTDB. *Service Oriented Computing and Applications*, 2019, zv. 13, s. 287 – 295. Dostupné z: https://api.semanticscholar.org/CorpusID:195804507.

[23] Lv, Z. a Fersman, E. *Digital Twins: Basics and Applications*. Springer International Publishing, 2022. ISBN 9783031114007. Dostupné z: https://books.google.cz/books?id=qq04zwEACAAJ.

[24] Manyika, J.; Chui, M.; Brown, B.; Bughin, J.; Dobbs, R. et al. Big Data: The Next Frontier for Innovation, Comptetition, and Productivity. *McKinsey Global Institute*, Jún 2011.

[25] Meehan, J.; Aslantas, C.; Zdonik, S.; Tatbul, N. a Du, J. Data Ingestion for the Connected World. In: *Cidr*. 2017, sv. 17, s. 8–11.

[26] Mell, P. a Grance, T. *The NIST Definition of Cloud Computing*. Special Publication (NIST SP), National Institute of Standards and Technology, Gaithersburg, MD, 2011-09-28 2011.

[27] Mishra, A.; Vidyakant, J.; Appasani, B.; Ray, A.; Gupta, D. et al. Emerging technologies and design aspects of next generation cyber physical system with a smart city application perspective. *International Journal of Systems Assurance Engineering and Management*, Január 2022.

[28] Modbus Organization. *Modbus Messaging Implementation Guide V1.0b*. October 2006. Dostupné z: https://modbus.org/docs/Modbus_Messaging_Implementation_Guide_V1_0b.pdf. [Zobrazené 7.12.2024].

[29] Modbus Organization. *Modbus Application Protocol Specification V1.1b3*. April 2012. Dostupné z: https://modbus.org/docs/Modbus_Application_Protocol_V1_1b3.pdf. [Zobrazené 7.12.2024].

[30] Naik, N. Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP. In: *2017 IEEE International Systems Engineering Symposium (ISSE)*. 2017, s. 1–7.

[31] OASIS Advanced Message Queuing Protocol (AMQP) Technical Committee. *OASIS Advanced Message Queuing Protocol (AMQP) Version 1.0*. October 2012. Dostupné z: https://docs.oasis-open.org/amqp/core/v1.0/amqp-core-complete-v1.0.pdf. [Zobrazené 7.12.2024].

[32] OPC Foundation. *Unified Architecture*. 2024. Dostupné z: https://opcfoundation.org/about/opc-technologies/opc-ua/. [Zobrazené 7.12.2024].

[33] Rathi, S. a Jeba, N. Effective data management and real-time analytics in internet of things. *International Journal of Cloud Computing*, Január 2021, zv. 10, s. 112.

[34] Sagiroglu, S. a Sinanc, D. Big data: A review. In: *2013 International Conference on Collaboration Technologies and Systems (CTS)*. 2013, s. 42–47.

[35] Schwarz, M. H. a Börcsök, J. A survey on OPC and OPC-UA: About the standard, developments and investigations. In: *2013 XXIV International Conference on Information, Communication and Automation Technologies (ICAT)*. 2013, s. 1–6.

[36] Shi, W.; Cao, J.; Zhang, Q.; Li, Y. a Xu, L. Edge Computing: Vision and Challenges. *IEEE Internet of Things Journal*, 2016, zv. 3, č. 5, s. 637–646.

[37] Simitsis, A.; Skiadopoulos, S. a Vassiliadis, P. The History, Present, and Future of ETL Technology. In: *DOLAP*. 2023, s. 3–12.

[38] Singhal, B. a Aggarwal, A. ETL, ELT and Reverse ETL: A business case Study. In: *2022 Second International Conference on Advanced Technologies in Intelligent Control, Environment, Computing Communication Engineering (ICATIECE)*. 2022, s. 1–4.

[39] Swales, A. et al. Open modbus/tcp specification. *Schneider Electric*, 1999, zv. 29, č. 3, s. 19.

[40] Thomas, G. Introduction to the modbus protocol. *The Extension*, 2008, zv. 9, č. 4, s. 1–4.

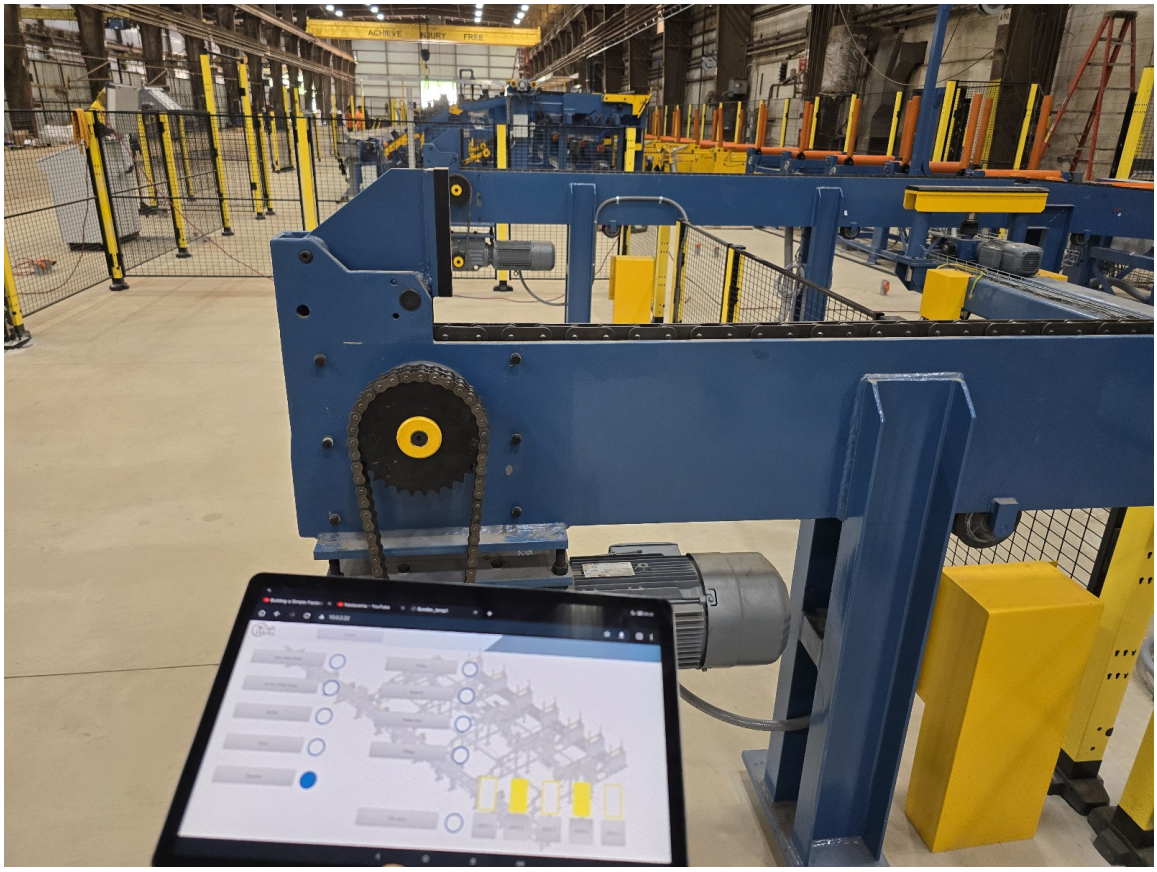
[41] Uckelmann, D.; Harrison, M. a Michahelles, F., ed. *Architecting the Internet of Things*. Springer, 2011. ISBN 978-3-642-19156-5. Dostupné z: http://dblp.uni-trier.de/db/books/daglib/0026698.html.

[42] Yaqoob, I.; Hashem, I. A. T.; Gani, A.; Mokhtar, S.; Ahmed, E. et al. Big data: From beginning to future. *International Journal of Information Management*, 2016, zv. 36, 6, Part B, s. 1231–1247. ISSN 0268-4012. Dostupné z: https://www.sciencedirect.com/science/article/pii/S0268401216304753.

Príloha A

Fotky reálnych nasadení systému

Obr. A.1: Linka na spracovanie ocelových rúr, USA



Obr. A.2: Ukážka HMI (human-machine interface) pre správu linky

Obr. A.3: B&R počítač, na ktorom beží aplikácia

Obr. A.4: Foto el. rozvádzača s riadenými PLC

Obr. A.5: Staršie nasadenie, pre ktoré sa bude systém implementovať