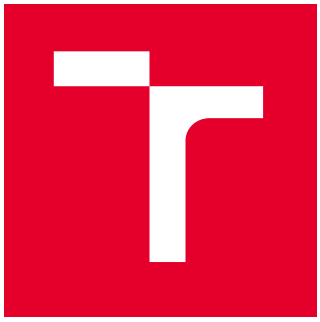



# VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

## FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

## ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

DEPARTMENT OF CONTROL AND INSTRUMENTATION


# MODUL PRO SBĚR DAT Z PRŮMYSLOVÝCH ZAŘÍZENÍ

MODULE FOR DATA COLLECTION FROM INDUSTRIAL EQUIPMENT

**DIPLOMOVÁ PRÁCE**
MASTER'S THESIS


**AUTOR PRÁCE**
AUTHOR

Bc. Petr Vávra

**VEDOUCÍ PRÁCE**
SUPERVISOR

doc. Ing. Petr Fiedler, Ph.D.



**BRNO 2022**

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

# Diplomová práce

magisterský navazující studijní program **Kybernetika, automatizace a měření**

Ústav automatizace a měřicí techniky

***Student:*** Bc. Petr Vávra ***ID:*** 191371

***Ročník:*** 2 ***Akademický rok:*** 2021/22

**NÁZEV TÉMATU:**

## Modul pro sběr dat z průmyslových zařízení

**POKYNY PRO VYPRACOVÁNÍ:**

Cílem práce je vytvoření aplikace pro sběr dat z průmyslových zařízení a vizualizaci dat klientovi.

1. Proveďte rešerši komerčně dostupných řešení a dostupných softwarových knihoven využitelných pro sběr dat.
2. Navrhněte architekturu a implementujte vlastní řešení s důrazem na zabezpečení a dostupnost serveru. Při návrhu serverového řešení vezměte v úvahu výběr dostupných cloudových řešení.
3. Otestujte funkčnost a realizujte na vybraném zařízení.

**DOPORUČENÁ LITERATURA:**

MQTT Essentials [online]. Landshut: HiveMQ, 2021 [cit. 2021-9-9]. Dostupné z: https://www.hivemq.com/mqtt-essentials/

***Termín zadání:*** 7.2.2022 ***Termín odevzdání:*** 18.5.2022

***Vedoucí práce:*** doc. Ing. Petr Fiedler, Ph.D.

**doc. Ing. Petr Fiedler, Ph.D.**
předseda rady studijního programu

## ABSTRAKT

Práce se zabývá návrhem modulu sběru dat z průmyslových zařízení. Jako vhodný protokol pro komunikaci s průmyslovými zařízeními je vybrán protokol OPC UA. Zdrojem dat jsou OPC UA servery na PLC výrobce Siemens. Pro zabezpečený transport přes TLS do veřejného cloudu Microsoft Azure je použit software IoT Edge a brána IoT Hub. Data jsou ukládána do Time Series databáze InfluxDB. Pro vizualizaci dat jsou použity webové aplikace Influx DB a Grafana. Software je testován v provozu na průmyslovém zařízení.



## KLÍČOVÁ SLOVA

IoT Edge, Iot Hub, Grafana, InfluxDB, Microsoft Azure



## ABSTRACT

Theses deals with development of a module for data collection from industrial equipment. As a suitable protocol for collecting data is chosen OPC UA. The source of data are OPC UA servers on PLCs from Siemens. For secure transport to public cloud Microsoft Azure over TLS protocol is used IoT Edge and IoT Hub gate. The data are stored in Time Series database InfluxDB. For visualization are used web applications Influx DB and Grafana. Software is tested on industrial equipment.



## KEYWORDS

IoT Edge, Iot Hub, Grafana, InfluxDB, Microsoft Azure

VÁVRA, Petr. *Modul pro sběr dat z průmyslových zařízení*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky, 2022, 93 s. Diplomová práce. Vedoucí práce: doc. Ing. Petr Fiedler, Ph.D.

# Prohlášení autora o původnosti díla

| | |
|---|---|
| **Jméno a příjmení autora:** | Bc. Petr Vávra |
| **VUT ID autora:** | 191371 |
| **Typ práce:** | Diplomová práce |
| **Akademický rok:** | 2021/22 |
| **Téma závěrečné práce:** | Modul pro sběr dat z průmyslových zařízení |

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

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

Brno ..... .....

podpis autora*

*Autor podepisuje pouze v tištěné verzi.

# Obsah

# Úvod

Práce vznikala ve spolupráci s firmou ACAM Solution s.r.o. Firma se zabývá zakázkovým vývojem a dodávkou automatizace pro průmyslové podniky; zaměstnává přibližně 30 zaměstnanců, zhruba polovina se zabývá vývojem, ostatní jsou pracovníci výroby a administrativy.

Firma má zájem o průzkum možností sběru a ukládání dat z instalovaných průmyslových zařízení. Předpokládaným využitím sebraných dat jsou interní potřeby firmy a dále jsou data zajišťována pro potřeby zákazníků. Interní potřeby jsou zejména analýzy využití průmyslového zařízení, zjišťování nestandardních průběhů měřených veličin, případně predikce poruch pro lepší plánování servisních akcí. Zákazníkům by bylo možné nabídnout grafickou vizualizaci sebraných dat. ACAM Solution s.r.o. předpokládá, že data z nasazených průmyslových zařízení, která se nyní používají pouze jako stavové veličiny pro řízení procesů a nejsou průběžně ukládána, mohou být cenná pro dlouhodobé analýzy.

Cílem této práce je provést průzkum trhu s již dostupnými řešeními, navrhnout a realizovat možná řešení pro tento problém. Data by měla být sbírána do databáze, která bude dostupná přes internet. Firma výrazně upřednostňuje řešení, ve kterém není třeba provozovat vlastní servery ani spravovat operační systémy na úrovni virtuálních strojů. Primárně tedy budou hledána dostupná řešení poskytovaná ve veřejných Cloudech ve formě PaaS nebo SaaS. Pro sběr dat se uvažuje o modulu, který bude mít přístup ke stavovým veličinám na průmyslovém zařízení. Modul by měl být především reprezentován softwarem, který bude možné provozovat na více druzích zařízení. Firma se tak chce vyhnout potřebě instalovat ke každé aplikaci dedikované zařízení určené na sběr dat a raději upřednostnit využití panelů HMI, PC, které již byly součástí jednotlivých dodávek.

Modul by měl v periodických intervalech odesílat zprávy zabezpečeným protokolem přes internet na sever, který data zpracuje a uloží do databáze. Dále by měla být k dispozici aplikace, jíž umožní vizualizaci dat z databáze. Při řešení bude snaha prozkoumat prostředky pro sběr dat ve veřejném Cloudu.

# 1 Analýza možných řešení

Úvodem kapitoly je probráno zadání práce a jsou navrženy možné architektury řešení. Dále jsou diskutovány komunikační protokoly pro sběr dat a podpůrné softwarové prostředky.

## 1.1 Architektura řešení

Cílem celého řešení je vytvořit možnost sbírat a uchovávat stavové veličiny, řídicí data a události z průmyslových zařízení. Tato data by měla být uložena nezávisle na průmyslovém zařízení a jejich dostupnost by neměla být svázána s dostupností průmyslového zařízení. Data by tedy měla být dostupná i v době odstavení průmyslového zařízení.

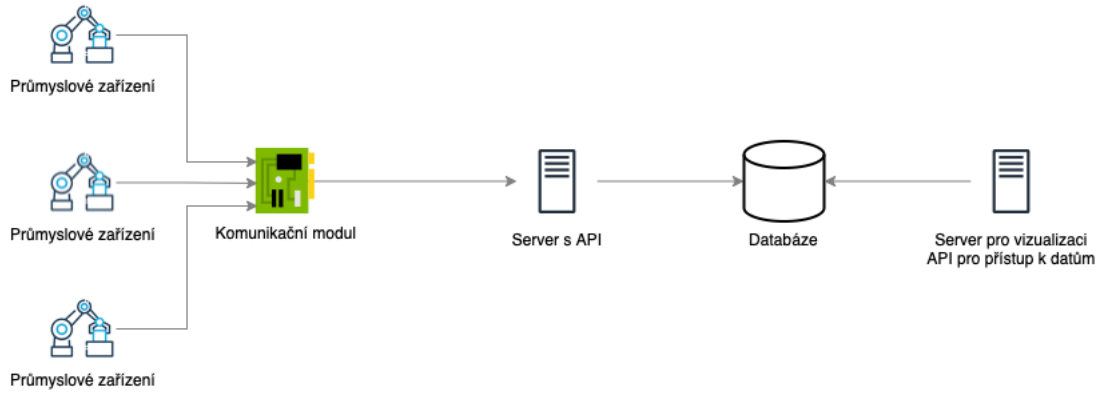
Jako vhodná průmyslová zařízení z automatizační pyramidy, která mají přehled o stavových veličinách a řídicích datech a zároveň jsou schopná komunikovat rozsáhlou řadou komunikačních protokolů s ostatními zařízeními, jsou PLC. Data by mělo být možné také extrahovat ze zařízení typu HMI, případně ze zařízení ze softwaru SCADA. Vyšší vrstvy automatizační pyramidy ERP a MES by sice také mohly být zdrojem dat k analýze, nicméně firma ACAM Solution se nezabývá dodávkou těchto produktů, a ne všichni zákazníci firmy tyto systémy využívají. Sběrem dat z těchto systémů se práce nezabývá.

Aby bylo možné data z jednotlivých lokací průmyslových zařízení (výrobních závodů a linek) agregovat na jednom místě, zároveň byla dostupná i mimo oblast nasazení průmyslového zařízení a bylo možné k nim bezpečně přistupovat téměř odkudkoliv na světě, bude pro komunikaci použita síť internet. Pro komunikaci od průmyslových zařízení směrem k úložišti dat by bylo možné využít i jiné sítě jako LoRA nebo SigFox. Hlavní výhody těchto sítí se promítnou při napájení komunikačního zařízení z baterie nebo při pohybu zařízení. Nevýhodou jsou nízké maximální datové toky v řádu desítek kb/s, případně limitovaný počet zpráv za den. Předností internetu je také vysoká dostupnost, schopnost PLC komunikovat pomocí protokolu IP, mnoho dostupných fyzických médií pro přenos dat.

Úložištěm dat bude databáze, která bude dostupná ze sítě internet. Aby bylo možné zajistit:

- jednodušší udržitelnost databázových požadavků při rozrůstající se velikosti databázového modelu,
- autentizaci a autorizaci různých průmyslových zařízení,
- případně filtraci dat pro úsporu úložiště databáze,

bude použit server s API. Místo přímého přístupu do databáze z komunikačního modulu bude zápis proveden prostřednictvím serveru s API. Většina dodávek firmy




Obr. 1.1: Obecné blokové schéma řešení

ACAM se obvykle skládá z více než jednoho průmyslového zařízení a více než jednoho PLC. Firma obvykle dodává PLC od firmy Siemens řady S7-1200, nebo S7-1500. Pro testování budou použity PLC těchto dvou řad, nicméně by měl být vybrán způsob komunikace, který bude podporován i jinými dodavateli PLC, případně jinými zařízeními jako HMI a SCADA. Pro usnadnění správy komunikace,

- snížení počtu jednotlivých dotazů na server API a zajištění možnosti agregace,
- zbavení PLC logiky, která by řešila ukládání dat pro případ výpadku serveru API nebo spojení se serverem,
- možnost vzdálené konfigurace komunikačního modulu ze serveru API,
- jednoduššího oddělení komunikace v lokální a vnější síti,
- možnost filtrovat a analyzovat data již v lokální síti,
- případně omezení zátěže méně výpočetně výkonných PLC zabezpečenou a šifrovanou komunikací s serverem API,

bude použit komunikační modul. To bude zařízení, které bude komunikovat se serverem API a koncovými průmyslovými zařízeními. Komunikační modul tvoří pro celé řešení kritický bod. Pokud přestane vykonávat svoji funkci, sběr dat bude zastaven. Výše zmíněné výhody převyšují tuto nevýhodu, výpadek komunikačního modulu neovlivní žádnou kritickou funkčnost průmyslového zařízení.

Požadavek na použití modulu bývá často udáván bezpečnostní politikou zákaznických firem. Pro správce sítí je jednodušší konfigurovat připojení z lokální sítě do sítě internet pouze pro jediné zařízení. Dalším požadavkem bývá komunikace pomocí jednoho protokolu aplikační vrstvy přes jeden port protokolu TCP. Dalším požadavkem může být komunikace s omezeným množstvím statických IP adres bez použití překladu DNS. V řešení se nepočítá s možností, že by modul zapisoval data do průmyslových zařízení nebo PLC, připojení modulu bude voleno pokud to bude možné v režimu pro čtení.

## 1.2 Komunikační protokoly průmyslových zařízení

Cílem této části je vytvořit přehled podporovaných komunikačních protokolů, které by bylo možné použít pro komunikaci s komunikačním modulem. Vzhledem k tomu, že je v plánu testovat řešení s PLC firmy Siemens řad S7-1200 a S7-1500, bude tato kapitola zaměřena hlavně na ně.

Tato PLC disponují ve všech variantách CPU komunikačním portem PROFINET. Zařízení jsou schopna komunikovat pomocí rozšiřujících modulů po dalších průmyslových sítích AS-Interface, Profibus, RS-485 a dalších. Vzhledem k tomu, že cílem odesílaných dat bude síť internet, je praktické zvolit rozhraní, které je kompatibilní s protokolem IP, tím je v tomto případě port PROFINET.

PROFINET na fyzické a linkové vrstvě ISO/OSI modelu je kompatibilní s protokolem Ethernetem. Na vyšších vrstvách ISO/OSI modelu jsou zařízení schopná komunikovat pomocí protokolů IP a TCP. Protože u sběru dat záleží na integritě, je vhodné použít protokol TCP místo UDP. Pro zajištění zabezpečení dat proti možnému zápisu a čtení útočníkem s fyzickým přístupem k síti se v [1] a [2] doporučuje použít protokol TLS. Pokud toto zabezpečení není podporováno komunikačním protokolem, pak se doporučuje síť chránit před útoky fyzicky. Dále v případě komunikace z lokální sítě do vnější je vhodné vždy zabezpečit komunikaci pomocí protokolu TLS, protože zabezpečení fyzického přenosového média již není možné. Jako hlavní výhody využití protokolu TLS jsou uvedeny:

- důvěrnost: data nemohou být čtena útočníky,
- integrita: zprávu dostane adresát ve stejné nezaměněné podobě,
- autorizace mezi koncovými body: identita obou stran komunikace je ověřena a nemůže být zaměněna.

Jako hlavní nevýhody jsou uvedeny:

- nedostupnost v předchozích verzích programovacího prostředí PLC Siemens TIA Portal a starších firmware zařízení,
- komunikace může být kvůli datové a výpočetní režii pomalejší.

Komunikační protokoly podporované na vyšších vrstvách s výstupně/vstupní nebo výstupní komunikací jsou uvedeny v tabulce 1.1 [1, 2]. Tyto protokoly jsou podporovány ve výchozí podobě programovacího prostředí TIA Portal bez instalace a použití dodatečných knihoven.

Další protokoly vyšších vrstev, které využívají protokol TCP, je možné na PLC zprovoznit s knihovnou Libraries for Communication for SIMATIC Controllers [3]. Všechny z uvedených protokolů v tabulce 1.2 podporují zabezpečení komunikačního kanálu přes protokol TLS.

Z uvedených protokolů v tabulkách 1.1 a 1.2 jsou pro přenos dat k zařízení s komunikačním modulem vhodné všechny, jejich konkrétní výhody a vlastnosti

| Vrstva | Protokoly |
|---|---|
| Aplikační | ISO-on-TCP, Modbus, OPC UA, S7 Communication GET/PUT |
| Transportní | TCP |
| Síťová | IP |
| Linková | Ethernet MAC |
| Fyzická | Ethernet |

Tab. 1.1: Vybrané podporované protokoly na rozhraní PROFINET

| Vrstva | Protokoly |
|---|---|
| Aplikační | HTTPS, MQTTS |
| Nižší vrstvy shodné jako v tabulce 1.1 | |

Tab. 1.2: Vybrané protokoly implementované v knihovně Libraries for Communication for SIMATIC Controllers

jsou diskutovány v následujících částech.

### 1.2.1 Modbus

Modbus je otevřený komunikační protokol, který je implementován v aplikační vrstvě ISO/OSI modelu a může být provozován na různých fyzických vrstvách. Na PLC Siemens lze provozovat na Ethernetu nebo na RS-485. Komunikace probíhá v režimu klient–server. Pouze klient může zahajovat dotazy a dotazovat se serveru na data. Server se pokusí na dotaz klienta odpovědět, případně odešle kód výjimky. Server nemá povědomí o klientovi. Protokol definuje datový model, který zařízení typu server zpřístupňuje klientům. Pro čtení a zápis z datového modelu jsou pak dostupné příslušné funkční kódy. [4].

Pro sdílení dat bude vhodné použít z datového modelu tabulku Hoding Registers, která poskytuje pro čtení a zápis 16bitový adresový prostor. Pomocí funkčních kódů může klient ze serveru vyžádat vyčtení nebo zápis jedné nebo více po sobě následujících adres. Postup pro zpřístupnění pole v paměti PLC jako tabulky Holding Registers Modbus serveru je uveden v [1].

### 1.2.2 S7 Communication PUT/GET

Protokol se využívá pro komunikaci mezi PLC výrobce Siemens. Komunikační protokol je proprietárním protokolem firmy Siemens a výrobce k jeho implementaci neposkytuje oficiální dokumentaci. Na internetu je přesto k dispozici několik volně dostupných implementací pro různé programovací jazyky. Příkladem takovéto knihovny je S7NetPlus [8] pro jazyk C#. Vlastnosti protokolu pro tyto implementace byly tedy

pravděpodobně získány pomocí reverzního inženýrství zachytáváním paketů sítě. Protokol nepodporuje šifrování komunikace a autentizaci účastníků komunikace.

Nevýhodou jsou bezpečnostní zranitelnosti tohoto protokolu. Programovací prostředí PLC TIA Portal umožňuje pouze explicitně povolit nebo zakázat tento způsob komunikace. Po povolení je možné se připojit z jakéhokoliv zařízení ve stejné síti. PLC kontroluje pouze maximální počet připojených zařízení. Protokol umožňuje téměř libovolně číst a zapisovat do paměti PLC.

Pokud by byla pominuta zmíněná bezpečnostní rizika, pak je protokol pro sběr dat z PLC pro vývojáře komfortní. K zpřístupnění dat není třeba upravovat program pro PLC, spuštění komunikace je jen otázkou konfigurace. Vybírat sbíraná data by bylo možné pouze konfigurací modulu sběru dat. Nevýhodou je absence jakékoliv anotace a významu dat na dané adrese PLC.

### 1.2.3 MQTTS

MQTT je síťový protokol aplikační vrstvy ISO/OSI modelu vystavěný nad protokolem TCP. Protokol je binární, je tedy úspornější na objem přenesených dat než protokoly textové jako třeba HTTP. Protokol pracuje na principu publish–subscribe. Jedná se o odlišný přístup proti principu klient–server, který využívá např. protokol Modbus.

Zprávy se organizují pomocí témat (Topic). Pro popis funkčnosti protokolu je třeba představit 3 základní role zařízení v přenosu dat. Publisher – klient, který publikuje informaci do tématu. Subscriber – klient, který se registruje k přijímání změn tématu. Broker – server nebo zprostředkovatel, který zajišťuje obsluhu komunikace mezi Publishery a Subscribery, tedy mezi klienty. Klienti komunikují pouze s Brokerem. Z tohoto principu plyne následující:

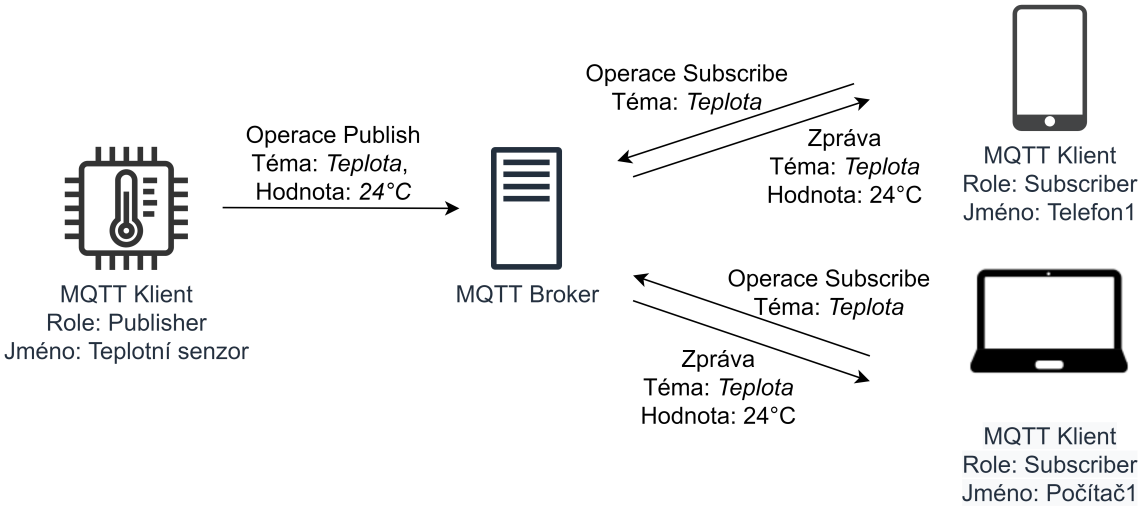
- klientů jednoho tématu může být až $n$,
- Publisher a Subscriber o sobě nemusí mít povědomí,
- Publisher a Subscriber nemusí běžet ve stejném čase. Broker může zprávu uložit a doručit Subscriberovi později [5].

#### Příklad komunikace

Příklad komunikace mezi klienty je naznačen na obrázku 1.2. Součástí komunikace jsou tři klienti a jeden Broker. Jeden klient je v roli Publisher (Teplotní senzor), ostatní klienti v roli Subscriber. Téma zprávy je *Teplota*. Komunikace probíhá krokově:

- Klienti v roli Subscriber provedou operaci Subscribe k Brokeru na téma *Teplota*. Tím žádají o doručování zpráv z tohoto tématu.
- Klient v roli Publisher publikuje do tématu *Teplota* zprávu s obsahem 24 °C.

- Broker zprávu přijme a rozešle ji klientům, kteří zaregistrovali Subscribe k tématu [5].




Obr. 1.2: Příklad komunikace přes protokol MQTT, upraveno z [6]

### Témata zpráv

Témata mají hierarchickou strukturu. Každá úroveň struktury je oddělena separátorem /. Kořenová úroveň nezačíná znakem /. Příkladem hierarchické struktury tématu může být:

- *VUT/FEKT/SE2.105/teplota*
- *VUT/FEKT/SE2.105/vlhkost*
- *VUT/Rektorat/A1.214/teplota*

Pro specifikaci tématu je možné využít zástupné znaky + a #. Znak + zastupuje jakékoliv téma na dané úrovni, znak # je zástupný pro všechny úrovně níže od něj. Vznik tématu klient dopředu neohlašuje, jednoduše do něj začne publikovat zprávy, obsluhu vzniku tématu zajišťuje Broker [5].

### QoS

Pro zajištění doručení zprávy protokol spoléhá na protokol TCP. Ten zajišťuje spolehlivost doručení, správné pořadí zpráv a kontrolu chyb pro případ jednoho TCP klienta. V IoT je třeba počítat s komunikací po méně spolehlivém připojení, tedy s výpadky spojení. Případně se spojení záměrně rozvazuje kvůli úspoře energie. Je tedy třeba do návrhu zahrnout, že doba výpadku připojení může být vyšší než TCP connection timeout. Protokol MQTT je schopen se postarat o konzistenci doručení

zpráv i přes opětovně navazované spojení TCP. Pro nastavení spolehlivosti je určen parametr QoS. Protokol poskytuje QoS ve třech úrovních:

- QoS 0 - at most once, kdy doručení dosahuje shodné spolehlivosti jako při použití běžného TCP,
- QoS 1 - at least once, kdy druhá strana potvrzuje přijetí kompletní zprávy, je ale možné, že zpráva bude doručena se stejným obsahem vícenásobně,
- QoS 2 - exactly once, kdy je záruka přijetí zprávy právě jednou [5].

#### Zabezpečení

Protokol obsahuje parametry pro autorizace klienta, a to username a password. Tyto informace jsou přenášeny jako text a protokol nezajišťuje jejich šifrování. Způsob a funkčnost autorizace záleží na implementaci Brokera. Protokol ve výchozí podobě neprovádí šifrování zpráv. Protokol je uzpůsoben pro používání nad protokolem TLS, při jeho využití se používá zkratka MQTTS [5].

#### Verze a implementace protokolu

Existují dvě standardizované verze protokolu. Verze 3.1.1, která byla standardizována organizacemi ISO a OASIS. Verze 5 byla standardizována organizací OASIS.

Klient je implementován v řadě programovacích jazyků pro množství hardwarových architektur, řada z nich je publikována jako open-source. Základní srovnání implementací brokerů a klientů nabízí [7].

### 1.2.4 TLS

TLS je kryptografický protokol, který je implementován na aplikační vrstvě ISO/OSI modelu. Implementaci lze také popsat jako prostor mezi transportní a aplikační vrstvou. V transportní vrstvě spoléhá typicky na protokol TCP, pro spolehlivou kryptografii je třeba zachovat pořadí bytů a zajistit bezchybné přijetí zprávy. Nad protokolem TLS lze provozovat řada internetových protokolů např. HTTP, FTP, MQTT nebo SMTP. Protokol zajišťuje bezpečný transport mezi stranami komunikace v sítích jako internet, kde nelze zabezpečit fyzickou vrstvu komunikace. Komunikace přes TLS probíhá v režimu klient–server. Před zahájením šifrování komunikace musí nejprve proběhnout domluva na jejím způsobu tak, aby jej podporovaly obě strany komunikace. Tato část se nazývá Handshake. Po úspěšném ukončení operace Handshake je všechna komunikace dále šifrována způsobem dohodnutým v průběhu Handshake. Podoba procesu Handshake se liší v závislosti na verzi protokolu TLS. Protokol při Handshake využívá asymetrického šifrování pomocí veřejných klíčů. Podstatné části procesu Handshake jsou následující:

- Klient žádá o spojení se serverem přes protokol TLS, nabízí seznam jím podporovaných šifer a hash funkcí.
- Server vybere jím podporovanou variantu šifry a hash funkce, poskytne svůj veřejný klíč a informaci zašifrovanou privátním klíčem.
- Klient ověří, že veřejný klíč serveru pochází od jemu důvěryhodné certifikační autority. Klient zašle pre-master secret serveru šifrované veřejným klíčem serveru, pokud jim server rozumí, má k dipozici svůj privátní klíč. Tím je ověřena identita serveru.
- Server může vyžádat veřejný klíč od klienta a ověřit jeho identitu.
- Na základě pre-master secret klient a server vygenerují master secret, což je unikátní klíč pro danou relaci, kterým je komunikace šifrována.

V průběhu Handshake se tedy na principu asymetrického šifrování ověří identita serveru a asymetrickým šifrováním se vytvoří symetrický klíč master secret, kterým je šifrován zbytek relace. Po ukončení relace je master secret zahozen. Pokud by byla komunikace zachytávána útočníkem a následně byl útočníkem získán i privátní klíč serveru, tak obsah komunikace nebude schopen kvůli způsobu generování master secret rozšifrovat [9, 10, 11].

#### Verze

V průběhu vývoje protokolu se mění kryptografické algoritmy hashování a šifrování, které se již nepovažují za bezpečné kvůli implementačním chybám nebo rostoucímu výpočetnímu výkonu. Dále se mění způsoby implementace operace Handshake a dalších. V poslední verzi protokolu 1.3 se výrazně změnil právě postup operace Handshake tak, že minimální počet transakcí od serveru ke klientovi se zmenšil na dvě.

| Verze | Podpora |
|---|---|
| SSL V1, V2, V3 | Zastaralé od 2013 |
| TLS 1.0, 1.1 | Zastaralé od 2020 |
| TLS 1.2, 1.3 | Podporované |

Tab. 1.3: Verze protokolu TLS, [11]

### 1.2.5 OPC UA

OPC UA je komunikační protokol s otevřenou specifikací, která je vydána pod licencí GPL. Protokol je spravován organizací OPC, standardizován je organizací IEC a převzat jako česká technická norma ČSN EN IEC 62541. Protokol je nezávislý na

platformě nasazení. K dispozici jsou softwarové implementace pro mikrokontroléry, tak pro vývojové platformy .NET, Java, Python.

V původní specifikaci protokolu, tedy verzi 1.00, je specifikována komunikace na principu klient–server, v novější verzi 1.04 přibyla možnost komunikace na principu publish–subscribe, ta je specifikována v části 14 (ČSN EN IEC 62541-14) protokolu. Komunikace v režimu publish–subscribe využívá typicky protokolu UDP a je vhodná pro časově kritickou komunikaci, nebo po komunikaci one-to-many, tedy když je třeba zprávy rozesílat více příjemcům. Výhodou implementace v OPC UA je možnost komunikovat na principu publish–subscribe i bez použití brokera. Pro monitorování dat s OPC UA je vhodnější využít principu klient–server.

Protokol definuje chování serveru, poskytované služby a adresní prostor serveru. Server může poskytovat přístup k aktuálním a historickým datům, alarmům, událostem a notifikovat o nich klienta. Server také může poskytovat informace (popis, metadata) o objektech v adresním prostoru. Data mohou být serverem odesílána ve formě binární struktury, XML nebo JSON formátu. Data mohou být zapouzdřena protokolem OPC UA TCP, HTTPS nebo WebSockets.

Protokol podporuje autentizaci a autorizaci. Jeden z možných způsobů vzájemné autentizace klienta a serveru je ověření pomocí certifikátů X.509. Protokol specifikuje možnost autorizace klienta pro přístup do vybraných oblastí adresního prostoru serveru. Protokol podporuje šifrování komunikačního kanálu, je definováno několik dostupných variant kryptografických algoritmů, kterými je možné komunikaci zabezpečit. Protokol nevyužívá přímo protokolu TLS, ale zabezpečení komunikačního kanálu vychází z podobných principů.

Sada objektů, které dává server k dispozici klientům, se nazývá adresní prostor. Ten je reprezentován sadou uzlů, které jsou vzájemně provázány referencemi. Jednotlivé uzly mají vlastnosti – atributy. Koncovými prvky stromu v adresním prostoru mohou být metody, události a proměnné.

Vlastnosti komunikace mezi serverem a klientem jsou definovány sadou služeb, které server poskytuje. Služby umožňují klientům zažádat server, aby je notifikoval o změnách jako jsou alarmy, změny hodnot proměnných, události a výsledky provádění metod nebo programů [12, 13].

### Implementace v Siemens PLC

Implementace nastavení OPC UA serveru do programovacího prostředí PLC Siemens TIA Portal je poměrně široká. Pro programátora se jedná pouze o konfiguraci serveru, uživatelů, zabezpečení a přístupu k jednotlivým datovým blokům.

Základní konfiguraci serveru lze provést ve vlastnostech CPU, modulu *Properties*, záložce *General* ve stromové struktuře pod výběrem možnosti *OPC UA*. Zde lze

OPC UA server aktivovat. IP adresa serveru je dána podle IP adresy přidělené portu PROFINET. Lze zvolit port serveru, minimální periodu vzorkování a odesílání hodnot klientovi serverem, maximální počet klientů a maximální počet odebíraných hodnot na klienta.

Autentizaci klienta lze nastavit pomocí uživatelského jména a hesla nebo pomocí certifikátu X.509. Server má automaticky vygenerovaný certifikát. Certifikát klienta lze nahrát do PLC a vynutit jeho ověření při navazování spojení. Pro uživatele lze povolit anonymní přístup nebo přístup s jménem a heslem. Ze seznamu lez volit kryptografické algoritmy, které server bude nabízet klientovi pro navázání a šifrování spojení. Variantou je i šifrování spojení zcela vypnout.

Pro přístup k jednotlivým datovým blokům (DB) lze v jejich vlastnostech (menu *Properties*, záložka *Attributes*) zvolit, zda budou dostupné přes OPC UA. Pro jednotlivé proměnné v datovém bloku lze zvolit zapisovatelnost (Writable) a přístupnost (Accessible) přes OPC UA. Při vypnutí možnosti přístupný je proměnná stále přes OPC UA viditelná, nicméně při pokusu o čtení hodnoty se objeví chybový kód.

Pro úspěšnou kompilaci projektu je třeba zvolit vybranou úroveň licence. Licence jsou ve třech úrovních - Basic, Medium, Large. Volba licence lze provést v Podle verze zvolené licence se liší schopnosti serveru. Pro testování s PLC řady S7-1200 byla zvolena licence Basic, pro testování s S7-1500 licence Large.

Zásadní rozdíl mezi implementacemi na PLC řady S7-1200 a S7-1500 je nutnost pro řadu S7-1200 vytvářet tzv. Server interface. Server interface specifikuje, které datové bloky a proměnné budou dostupné přes OPC UA. Zároveň zachovává konfigurace přístupnosti z předchozího odstavce. Konfigurace je jednoduchá, stačí vybrané datové bloky přesunout pod do Server inetrface. U řady S7-1500 jsou naopak všechny datové bloky dostupné implicitně.

S PLC řady S7-1200 bylo testováno zabezpečení komunikačního kanálu algoritmy Basic256Sha256 – Sign & Encrypt. Pro člověka je evidentní výrazné zpomalení navázání spojení proti nezabezpečenému spojení.

Testování proběhlo na TIA Portal V17 s STEP 7 Professional V17 v kombinaci s PLC S7-1214 (6ES7 214-1AG40-0XB0) a S7-1512SP (6ES7 512-1SK01-0AB0). V roli klienta byl použit program UaExpert.

### 1.2.6 Shrnutí

Komunikace mezi PLC a modulem sběru dat bude probíhat v lokální síti, proto lze bezpečnost zajistit ochranou fyzického média. Přesto je výhodné používat protokol, který alespoň podporuje šifrování komunikačního kanálu. Protože se nepočítá s potřebou zápisu dat, je vhodné, aby protokol podporoval možnost konfigurace přístupu k datovým blokům a omezení zápisu do PLC. Výhodou je využít protokoly, které

jsou již součástí programovacího softwaru TIA Portal a bylo možné je doplnit pouze konfigurací do stávajících aplikací.

Vhodným protokolem pro sběr dat z PLC Siemens je OPC UA. Protokol podporuje zabezpečení šifrování komunikačního kanálu. Dále podporuje navázání spojení s autentizací a autorizací obou komunikačních stran. V programovacím prostředí TIA Portal lze zvolit, které datové bloky a proměnné budou zpřístupněny pro komunikaci přes OPC UA. Implementace serveru OPC UA umožňuje klientovi vybrat konkrétní proměnné a nastavit interval pozorování změn a notifikovat klienta při změně hodnoty. Tuto aplikační logiku tedy není nutné implementovat v PLC, navíc tyto hodnoty jsou voleny operativně klientem. Princip notifikace o změně proti principu polling u některých veličin, jako jsou alarmy, může značně uspořit datový tok. Další výhodou je možnost vyčtení stromové struktury datových bloků a proměnných v PLC za běhu programu.

Žádný jiný protokol mimo OPC UA neumožňuje nativně z PLC Siemens odesílat anotované názvy datových bloků v programu PLC.

## 1.3 Kontejnery softwaru

Kontejnery jsou způsob izolace programu a jeho podpůrného sotware (knihovny, frameworky) do odděleného prostředí. Oddělené prostředí vzniká na úrovní operačního systému, nevzniká tak další virtualizovaný operační systém jako u virtuálních strojů. Je umožněna kontrola prostředků, které program využívá – síťové prostředky, přístup k file-systému, prostředky procesoru a paměti. V kontejneru se programu jeví, jako by na operačním systému běžel sám. Na rozdíl od virtualizace celého operačního systému je tento způsob oddělení méně náročný na výpočetní výkon. Výhodou je také jednodušší správa nasazených programů.

Kontejnery jsou schopné vyřešit problém s různými závislostmi programů na podpůrném softwaru. Příkladem je program, který běží nad Node.js na verzi 16 a je třeba otestovat, zda poběží na verzi 18. Při klasické instalaci by bylo třeba odinstalovat z počítače aktuální verzi Node.js, instalovat novou, testovat. Návrat zpět k původní verzi by opět znamenal reinstalaci podpůrného softwaru. V případě využití kontejneru lze spustit obě verze na jednom stroji souběžně. V konfiguraci vytvoření kontejneru lze pouze modifikovat závislostí programu na jinou verzi a spustit novou verzi kontejneru. Zároveň může běžet verze stará. Pokud by program používal síťový port třeba pro hostování webového prostředí, lze jednoduše upravit v konfiguraci kontejneru propagaci na jiný port pro hostující operační systém.

Typickým prostředkem, který se využívá pro kontejnery, je Docker. Docker poskytuje software pro běh kontejnerů nad operačním systémem, CLI pro správu běžících kontejnerů a nástroje pro zapouzdření programu do kontejneru, tedy tvorbu

image. Dále zajišťuje hostování image na webu Docker Hub, kde jsou image přístupné pro stažení. Na Docker Hubu lze nalézt např. oficiální image Node.js. Při distribuci programu na libovolný operační systém nebo architekturu procesoru podporující Docker by chování programu mělo být obdobné.

# 2 Microsoft Azure

V této kapitole budou diskutovány služby v Cloudu Microsoft Azure pro sběr dat z průmyslových zařízení.

Na trhu s Cloudovými službami se odvětví, které zajišťuje komunikaci a správu dat z fyzických zařízení s přístupem k internetu, nazývá internet věcí neboli Internet of Things (IoT). Pro průmyslový internet věcí se používá výrazu Industrial Internet of Things (IIoT). Největšími poskytovateli Cloudových řešení na trhu jsou podle [14] v pořadí Amazon Web Services, Microsoft Azure a Google Cloud. Tento výzkum se týká veškerých poskytovaných služeb v Cloudu. Statistiky o obratu nebo počtu zákazníků v kategorii internet věcí poskytovatelé Cloudu neuvádějí. O největších poskytovatelích Cloudových služeb pro internet věcí lze tedy usuzovat z ankety vývojářů v tomto odvětví. Podle [15] jsou největšími poskytovateli veřejného Cloudu v pořadí opět Amazon Web Services (40 %), Microsoft Azure (31 %) a Google Cloud (26 %).

Cloudové služby se standardně účtují jednou za měsíc podle provozu služeb. Obvyklá smlouva pro běžné uživatele je typu Pay-As-You-Go. Cena za využité služby je jednou měsíčně fakturována a stržena z bankovní karty. Ostatní obchodní smlouvy obvykle využívají větší zákazníci jako firemní korporace. Porovnávat předem cenu jednotlivých poskytovatelů je náročné, protože každý poskytovatel obvykle nabízí jiné způsoby výpočtu konečné ceny za jednotlivé služby. Celé Cloudové řešení se bude skládat z několika služeb. Poskytovatelé nenabízí obvykle přímo si odpovídající služby. Každá služba obvykle má jinou účetní jednotku. Účetní jednotka může být např. průtok dat službou nebo velikost využitého úložiště. Alternativně se služby nabízí v různých edicích (Editions) a úrovních (Pricing Tier), ke kterým jsou přiřazený náklady. Úrovně definují dostupné vlastnosti služby např. možnost zasílat zprávy do zařízení. Edice určují limity služeb, např. maximální objem přenesených dat. Nejpraktičtější srovnání nákladů nabízí testování celého řešení u jednotlivých poskytovatelů pod různou zátěží a následné porovnání celkových nákladů.

Pro testovací implementaci byl vybrán poskytovatel Microsoft Azure. Poskytovatel nabízí pro studenty starší 18 let, kteří studují prezenční studijní program zakončený titulem, benefiční program s názvem Azure for Students. V rámci tohoto programu student získá kredit 100 USD ročně na prostředky v Cloudu Azure. Zároveň není třeba k účtu registrovat platební kartu. Kredit lze uplatnit na veškeré služby poskytované firmou Microsoft, nelze je uplatnit na služby a softwarové produkty třetích stran poskytovaných v Azure [16].

Výchozí měnou v Azure je americký dolar (USD). Podle udané adresy zákazníkem se určuje z tabulky účetní měna, která je strhávána z bankovní karty. Pro adresy v České republice je účetní měna euro (EUR). Kurz pro převod mezi těmito měnami

je stanoven na konci kalendářního měsíce pro každý následující kalendářní měsíc podle Thomson Reuters benchmark [17].

Azure nabízí služby v několika datových centrech po celém světě. Datová centra se obvykle vzájemně liší cenou poskytovaných služeb a jejich dostupností. Totožná služba IoT Hub může mít jinou cenu v datacentru North Europe a West Europe. Dále tato služba může být dostupná v datacentru North Europe a zároveň nemusí být dostupná v datacentru Norway West. Obecně se doporučuje celé řešení nasazovat do jednoho datacentra. Dále je vhodné vybrat geograficky blízké datacentrum k připojovaným zařízením. Příklady cen budou uváděny z datacentra North Europe, které je jedno z nejstarších datových center Cloudu Azure v Evropě. Fyzicky je umístěno v Dublinu v Irsku a jeho předností je dostupnost veškerých služeb z portfolia Microsoft v kategorii IoT a v porovnání s ostatními datovými centry v Evropě má obvykle nižší náklady za služby.

## 2.1 IoT Hub

Vstupní datovou bránou pro IoT data z koncových zařízení je služba Azure IoT Hub. Brána zajišťuje bezpečné připojení koncových zařízení, správu koncových zařízení a doručení zpráv z koncových zařízení k dalším službám. Zároveň umožňuje odesílat zprávy z Cloudu do koncového zařízení.

Komunikace mezi koncovým zařízením a bránou v obou směrech se tedy nazývá zpráva. Obsahem zprávy budou stavové veličiny ze zařízení a diskrétní události neboli eventy. Obsah zpráv standardně používá formátování JSON. Zpráva je jednotkou pro účtování přenosu dat. Podle počtu zpráv a jejich velikostí se také volí Pricing Tier a Edition služby IoT Hub.

### 2.1.1 Komunikace z koncových zařízení

Podporované komunikační protokoly aplikační vrstvy pro komunikaci zpráv mezi IoT Hub a koncovým zařízením jsou:

- MQTT
- MQTT over WebSockets
- AMQP
- AMQP over WebSockets
- HTTPS

Dokumentace doporučuje využít protokol MQTT pro zařízení s nižším výpočetním výkonem jako jsou mikrokontroléry a dále pro zařízení, která připojují IoT Hub pouze sami sebe. Protokol MQTT nepodporuje rozlišení na další koncová zařízení.

Protokol AMQP je doporučen pro zařízení typu Field Gateway. To zprostředkovává připojení k IoT Hubu pro další zařízení, která nejsou sama schopná komunikovat s IoT Hubem. Tento protokol podporuje rozlišení původního zařízení a doručení zprávy z IoT Hub přímo do koncového zařízení. Protokol je obecně obsáhlejší a náročnější na výpočetní výkon než protokol MQTT.

Protokoly *over WebSockets* jsou podporovány pouze pro zajištění komunikace přes port 443 v sítích, kde se komunikace přes jiné porty nepovoluje.

Posledním podporovaným protokolem je HTTPS. Pro komunikaci se neuvažuje, že by koncové zařízení mělo být dostupné z internetu přes veřejnou IP adresu a provozovat server. Komunikace přes HTTPS je tedy neefektivní pro komunikaci z IoT Hubu směrem ke koncovému zařízení, jež se musí v pravidelných intervalech dotazovat IoT Hubu na dostupnost nových příchozích zpráv. Maximální interval pro produkční nasazení je 25 minut [18]. Výhodou je, že protokol je velmi rozšířený a jednoduchý pro implementaci na koncovém zařízení.

Výhodou protokolů AMQP a MQTT je, že byly navrženy pro stálé udržování spojení, nativně umí řešit potvrzení přijetí zpráv, výpadky připojení a opakování odeslání zprávy po selhání. Jsou také méně náročné na množství přenesených dat, protože se jedná o binární protokoly. HTTPS oproti tomu používá textový zápis hlavičky, který je výrazně delší. Při přenášení telemetrických dat může být sama hlavička protokolu výrazně větší než přenášená data.

Všechny zmíněné protokoly používají v IoT Hubu protokol TLS pro navázání spojení. Standardně podporovaná verze protokolu je 1.2. Každé koncové zařízení musí využívat k autentizaci jednu z následujících možností:

- symetrický klíč,
- certifikát X.509 podepsaný sebou,
- certifikát X.509 podepsaný certifikační autoritou.

Obecně za bezpečnější lze požadovat varianty s certifikátem.

### 2.1.2 Správa koncových zařízení

Koncová zařízení lze připojit k IoT Hubu přímo s pomocí zmíněných protokolů. Pro tato zařízení IoT Hub může zabezpečovat služby:

- bezpečné připojení ke Cloudu,
- přijímání telemetrie zařízení,
- aktualizace konfigurace zařízení,
- zahájení nahrávání souboru,
- aktualizace firmware zařízení.

Celé řešení je navrženo tak, aby bylo škálovatelné a bylo schopné připojit i tisíce zařízení. Konečnou podobu zmíněných služeb je třeba implementovat ve firmware

zařízení. Pro usnadnění implementace společnost Microsoft poskytuje pro koncová zařízení řadu vývojových knihoven. Knihovny jsou dostupné pro různé vývojové platformy, embedded zařízení i pro zařízení na procesorových architekturách ARM a x64. Knihovny jsou poskytovány pro řadu platforem, jmenovitě .NET, C, Java, Node.js, Python.

Zároveň jsou poskytovány knihovny pro softwarovou automatizaci správy koncových zařízení v IoT Hubu a IoT Hubu samotného. S pomocí těchto knihoven je možné např. zaregistrovat předpis pro několik zařízení a vygenerovat jim přístupové certifikáty.

### 2.1.3 Konektivita PLC Siemens s IoT Hubem

Konektivita ke Cloudovým službám PLC firmy Siemens je diskutována v materiálu firmy [19]. Z protokolů zmíněných v kapitole 1.2 je přímo kompatibilní protokol MQTT a HTTPS. Microsoft neposkytuje SDK pro připojení PLC Siemens k IoT Hubu, neposkytuje jej ani společnost Siemens, která v materiálu pouze zmiňuje [19], že může být funkcionalita naprogramována. Komunita vývojářů PLC Siemens na veřejných fórech diskutuje možnosti přímého spojení, nicméně univerzální knihovna pro spojení nebyla nalezena. Microsoft v materiálu [20] silně doporučuje použít jejich SDK a pokud není k dispozici, pak protokol MQTT pokud jej zařízení podporuje. Jako hlavní výhody užívání SDK je jeho pravidelná údržba a důraz na bezpečnost při vývoji.

Z vlastností IoT Hub pro správu koncových zařízení se očekává využití bezpečného připojení a přijímání telemetrie. O ostatních vlastnostech vzhledem k způsobu programování a konfigurace koncového zařízení PLC nemá smysl příliš uvažovat.

V dokumentaci [21] je zmíněna možnost využití namísto přímého spojení připojení pomocí prostředníka IoT Edge. Tento koncept odpovídá více zadání práce a také umožňuje uspokojit obvyklou bezpečnostní politiku koncových zákazníků společnosti ACAM Soulution, která zakazuje komunikaci PLC s vnějšími sítěmi. IoT Edge umožní přijímat zprávy z koncových zařízení a odesílat je do IoT Hubu [21].

IoT Edge není jediný způsob, jakým navázat spojení přes prostředníka s IoT Hubem z PLC Siemens. Siemens v [19] doporučuje využití vlastních hardwarových a softwarových řešení. Dále je možné využít řešení třetích stran.

### 2.1.4 Směrování zpráv, koncové body

IoT Hub zajišťuje přijetí zpráv a směrování k dalším službám v rámci Cloudu Azure nebo jiným konzumentům v síti. Souhrnně lze tuto skupinu pojmenovat konzumenty zpráv. Jednu zprávu ze zařízení lze směrovat k více konzumentům. Data ze zařízení

lze před odesláním ke konzumentům filtrovat. Dále je možné ke zprávě přidat např. identifikátor zdrojového zařízení. Výhodou přidání těchto dat na úrovni IoT Hubu je, že se tím sníží velikost příchozí zprávy do IoT Hubu. To může vést k snížení objemu přenesených dat ve zprávě, a tím pádem ceny za službu IoT Hub. Dále se neúčtuje žádná cena za přesměrovaní zpráv k více konzumentům. Konzument pro zpracování zpráv ze zařízení do Cloudu musí být kompatibilní s protokolem Event Hub Endpoint. Předpis vlastností Event Hub Endpoint je popsaný v [21] a je k němu k dispozici vývojové SDK. Použitým protokolem aplikační vrstvy, na kterým je vystavěn protokol Event Hub Endpoint, je AMQP. Další možností je zprávy přímo uchovávat v Azure Storage containers [21].

IoT Hub monitoruje stavy konzumentů. Stavy konzumentů jsou v tabulce 2.1.

| Stav | Popis stavu |
|---|---|
| healthy | Akceputje příchozí zprávy dle očekávání. |
| unhealthy | Neakceptuje příchozí zprávy a IoT Hub se pokouší znovu odeslat zprávy. |
| unknown | IoT Hub se nepokusil odeslat zprávy do koncového bodu. |
| degraded | Akceptuje zprávy pomaleji než očekáváno nebo se zotavuje ze stavu unhealthy. |
| dead | IoT Hub se nepokouší nadále doručovat zprávy, opětovné pokusy o doručení selhaly. |

Tab. 2.1: Stavy konzumentů IoT Hubu [21]

Pokud je nějaký z konzumentů nedostupný, pak IoT Hub pozdrží zprávy, které se k němu směrují. Pro každou skupinu konzumentů a jednotlivé cesty směrování si IoT Hub udržuje ukazatel na poslední doručenou zprávu. Maximální doba uložení zpráv je konfigurovatelná až na 7 dní. Nicméně maximální kapacita uložení je určena součinem maximálního počtu zpráv za den a maximální velikostí zprávy ve zvolené cenové kategorii IoT Hubu. S jistotou tedy lze tvrdit, že zprávy budou pro danou cenovou kategorii uchovány alespoň jeden den.

### 2.1.5 Cenové kategorie

Služba se poskytuje ve dvou edicích: Standard a Basic. V rámci každé edice je k dispozici několik cenových kategorií. Edice Standard má navíc tyto vlastnosti proti verzi Basic:

- zasílání zpráv z IoT Hub do koncového zařízení,
- připojení produktu IoT Edge,

- udržování a synchronizace předpisu funkcionality a vlastností koncových zařízení (Device Twin),
- službu Device Streams [22].

| Typ edice | Cena za jenotku služby IoT Hub (za měsíc) [EUR] | Celkový počet zpráv za den na jednotku | Maximální velkost zprávy [kB] |
|---|---|---|---|
| F1: Free | Zdarma | 8000 | 0,5 |
| S1 | 22,329 | 400 000 | 4 |
| S2 | 223,285 | 6 000 000 | 4 |
| S3 | 2232,841 | 300 000 000 | 4 |

Tab. 2.2: Ceny za úroveň Standard služby IoT Hub [22]

Edice bude zvolena na úroveň Standard, protože bude snaha otestovat propojení IoT Edge a IoT Hub. Ve verzi Standard je k dispozici edice Free, která je zdarma. Její nevýhodou je, že ji nelze automaticky škálovat do jiné edice ani úrovně. Ostatní edice lze libovolně škálovat. Lze tedy podle potřeby plynule přejít mezi verzí S1 a S2 bez výpadku dostupnosti služeb. Dále není nutné z edice S1 přecházet přímo do edice S2, lze zvětšovat počet zakoupených jednotek v edici S1. S každou jednotkou se účtuje uvedená cena za jednotku a na každou jednotku je navázán příslušný počet zpráv.

Pokud zpráva přesahuje maximální velikost, pak dojde k jejímu rozpočtení do více zpráv. Velikost je podělena maximální velikostí, výsledek je zaokrouhlen vždy nahoru a odečten od celkového dostupného množství zpráv za den. Pokud dojde při užívání služby k vyčerpání celkového počtu zpráv na den, pak se služba pozastaví. Následně se přestanou přijímat a odesílat zprávy, nelze zobrazit a administrovat zařízení. K odblokování dojde po následující půlnoci času UTC.

Služba má další limity např. maximální množství zpráv přijatých z jednoho zařízení na den, maximální velikost zprávy a další. Tyto limity se pro účel práce i pro budoucí nasazení nejeví jako kritické. Velká část z nich je pravděpodobně zavedena kvůli zabezpečení služby proti zneužívání uživatelem k jiným účelům. Všechny limity jsou uvedeny v [23].

## 2.2 IoT Edge

IoT Edge je open-source software vyvíjený společností Microsoft. Software je určen k nasazení v průmyslových provozech a lze jej provozovat na široké škále zařízení. Typickým účelem je zprostředkování připojení k IoT Hubu zařízením, která nejsou schopná komunikovat protokolem kompatibilním s IoT Hubem. Dalším využitím

je provozování software (databází, strojového učení) v blízkosti průmyslových zařízení. Nasazený software má při provozu v lokální síti výrazně nižší latenci než při komunikaci s Cloudem, zároveň však není třeba všechna zdrojová data přenášet do Cloudu. Výhodou využití IoT Edge je možnost spravovat, monitorovat, konfigurovat a instalovat software pomocí IoT Hubu.

IoT Edge se skládá z těchto komponent:

- IoT Edge moduly,
- IoT Edge runtime,
- IoT Edge Cloudové rozhraní.

IoT Edge moduly jsou kontejnery kompatibilní s Dockerem, ve kterých může běžet vlastní kód, služba třetí strany a případně lokálně provozované obdoby služeb z Cloudu (Azure Services). Moduly mezi sebou mohou komunikovat přes IoT Edge. Pro implementaci této komunikace je k dispozici SDK a příklady.

IoT Edge runtime se stará o instalaci jednotlivých modulů. Dále kontroluje dostupnost jednotlivých modulů, monitoruje jejich stav, reportuje stav na Cloud, zajišťuje komunikaci s Cloudem, mezi jednotlivými moduly a koncovými zařízeními.

Cloudové rozhraní slouží k správě a monitoringu jednotlivých IoT Edge zařízení z IoT Hubu. Správa z Cloudu by měla usnadnit škálovatelnost celého řešení. Cílem je zjednodušení a automatizace nasazení software bez nutnosti přímé konfigurace jednotlivých zařízení [24].

### 2.2.1 Podporovaná zařízení a operační systémy

IoT Edge je nyní podporován ve dvou verzích, jsou to verze 1.1 a 1.2. Verze 1.1 je podporována v režimu LTS a bude aktualizována do konce podpory vývojového SDK .NET Core 3.1, na kterém je postavená. Tato podpora skončí 3. 12. 2022. Verze 1.2 je aktivní vývojovou větví, ukončení její podpory zatím není naplánováno.

Verze 1.1 lze provozovat nad operačními systémy Windows a Linux, ale verze 1.2 lze již provozovat pouze nad Linuxem, pro Widnows je třeba využít virtuálního stroje nad Windows. Pro testování bude nasazena nejnovější verze 1.2, další specifika jsou uvedena pro tuto verzi.

Při výběru konkrétního operačního systému a architektury procesoru jsou poskytovány dvě úrovně podpory. V úrovni 1 jsou operační systémy a architektury, které jsou aktivně testovány s IoT Edge. V úrovni 2 jsou systémy, jež jsou z principu kompatibilní, nicméně nejsou při vývoji testovány. V úrovni 1 je pro práci zásadní

- Raspberry Pi OS Stretch na architektuře ARM32v7,
- Ubuntu Server 18.04 na architekturách AMD64 a ARM64,
- Windows 10/11 Pro na architekturách AMD64 a ARM64 [25].

Kvůli omezení problémů s případnou nekompatibilitou, bude v práci použita pouze úroveň 1.

### 2.2.2 Předpis Deployment Manifest

Deployment manifest je strukturovaný předpis požadovaných modulů a jejich konfigurace na zařízení IoT Edge. Definici lze upravit pomocí portálu Azure, pomocí nadstavby Azure IoT Edge pro Visual Studio Code. Součástí Deployment Manifestu by měly být dva základní systémové moduly edgeAgent a edgeHub. Tyto moduly zajišťují vykonávání funkcí IoT Edge Runtime. V Deployment Manifestu se specifikují:

- názvy modulů k nasazení na zařízení IoT Edge,
- registr kontejnerů, ze kterých lze stáhnout obraz modulu,
- konfigurace modulu,
- směrování zpráv z modulu a do modulu,
- parametr TTL (Time to live) zpráv.

Příklad Deployment Manifestu je uveden v příloze A.1.

Nová verze Deployment Manifest se indikuje v Device Twin v IoT Hubu. Zařízení IoT Edge automaticky po zjištění rozdílu mezi lokální verzí Device Twin a verzí v IoT Hubu zahájí stažení nového Deployment Manifestu. IoT Edge runtime následně obstará požadované změny v modulech z Deployment Manifestu.

Pro splnění zadání práce je podstatný parametr TTL (Time to live). Určuje dobu zachování zpráv v sekundách při výpadku konektivity. V případě výpadku spojení s IoT Hub nebo mezi jednotlivými moduly jsou zprávy ukládány do zásobníku a jsou vykonávány opětovné pokusy o jejich doručení. Po uplynutí doby TTL jsou zprávy, které se nepodařilo doručit ze zásobníku, zahazovány. Výchozí hodnotou je 7200 sekund.

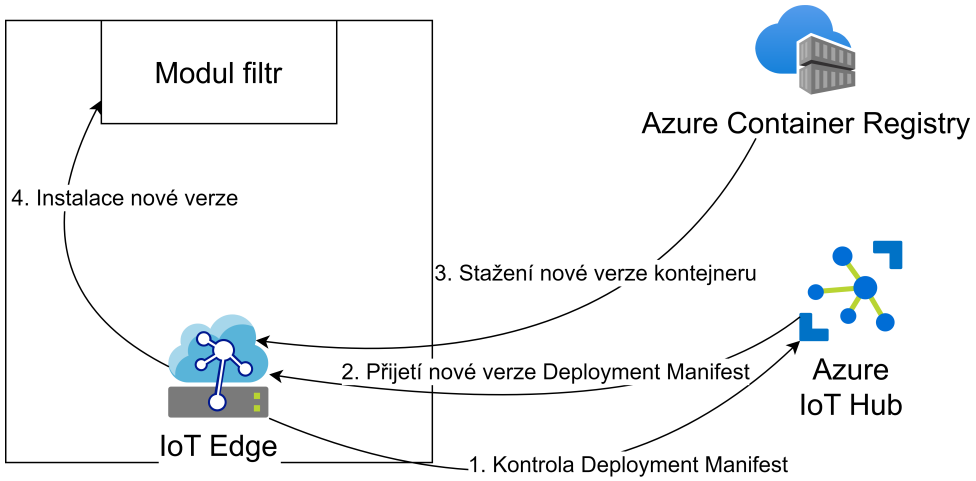
### 2.2.3 Vývoj modulu pro IoT Edge

Pro IoT Edge lze vyvíjet vlastní moduly. Podporovanými programovacími jazyky pro vývoj modulu jsou C, C#, Java, Node.js a Python. Doporučeným vývojovým prostředím je Visual Studio Code. Pro tyto programovací jazyky jsou k dispozici funkční příklady modulů a knihovny pro vývoj [26].

### 2.2.4 Instalace obrazu modulu z Container Registry

Postup instalace modulu je ilustrován na obrázku 2.1. Před zahájením instalace je třeba nahrát vyvinutý software ve formě image kontejneru v aktuální verzi do Azure Container Registry. Tato činnost je automatizována ve Visual Studio Code. Dalším

krokem je publikování předpisu Deployment Manifest v Azure IoT Hub. Následně zařízení s IoT Edge detekuje změnu v předpisu Deployment Manifest. IoT Edge runtime se následně postará o stažení obrazu modulu sensor z Azure Container Registry a jeho instalaci na zařízení. Po instalaci reportuje stav instalovaného modulu filtr do IoT Hubu. Modul filtr se spustí a začne vykonávat svou funkci.




Obr. 2.1: Schéma instalace modulu IoT Edge z Container Registry

## 2.3 Azure Container Registry

Služba slouží pro zpřístupnění image kontejnerů pro stažení z internetu. Dále poslouží pro správu a zabezpečení uložených obrazů. Výhodou je dobrá integrace s vývojovými prostředími od firmy Microsoft.

### 2.3.1 Cenové kategorie

Služba se nabízí ve třech úrovních. Úroveň Basic nabízí 10 GB úložiště pro kontejnery. Alternativní službou je Docker hub.

| Úroveň | Cena (za den) [EUR] | Zahrnuté úložiště [GB] | Webhooky celkem |
|---|---|---|---|
| Basic | 0,149 | 10 | 2 |
| Standard | 0,596 | 100 | 100 |
| Premium | 1,489 | 500 | 500 |

Tab. 2.3: Ceny za službu Container Registry [27]

## 2.4 Azure Industrial IoT Platform

Azure Industrial IoT Platform je softwarové řešení pro propojení lokálních OPC UA serverů do Azure. Řešení je publikované jako open-source pod licencí MIT. OPC UA servery lze v lokální síti aktivně vyhledávat, z těchto serverů sbírat a operativně vyčítat data. Data lze dále v Azure pomocí dalších služeb ukládat a analyzovat. Dále umožňuje správu certifikátů X.509 pro OPC UA servery a jejich klienty.

Řešení využívá pro komunikaci IoT Hub popsaný v kapitole 2.1 a pro překlad protokolu OPC UA sadu modulů pro IoT Edge, popis IoT Edge v kapitole 2.2. Pro ovládání modulů v IoT Edge, zpracování telemetrie a jejich konfiguraci je připravena sada webových služeb nazývaných Microservices. Microservices zpřístupňují pro programátory API přes HTTPS, které umožňuje povelovat moduly v IoT Edge.

Autentizace a autorizace pro povelování a přístup k API Microservices je řešena přes Azure Active Directory, která umožňuje správu uživatelů.

Platforma poskytuje tyto moduly pro IoT Edge:

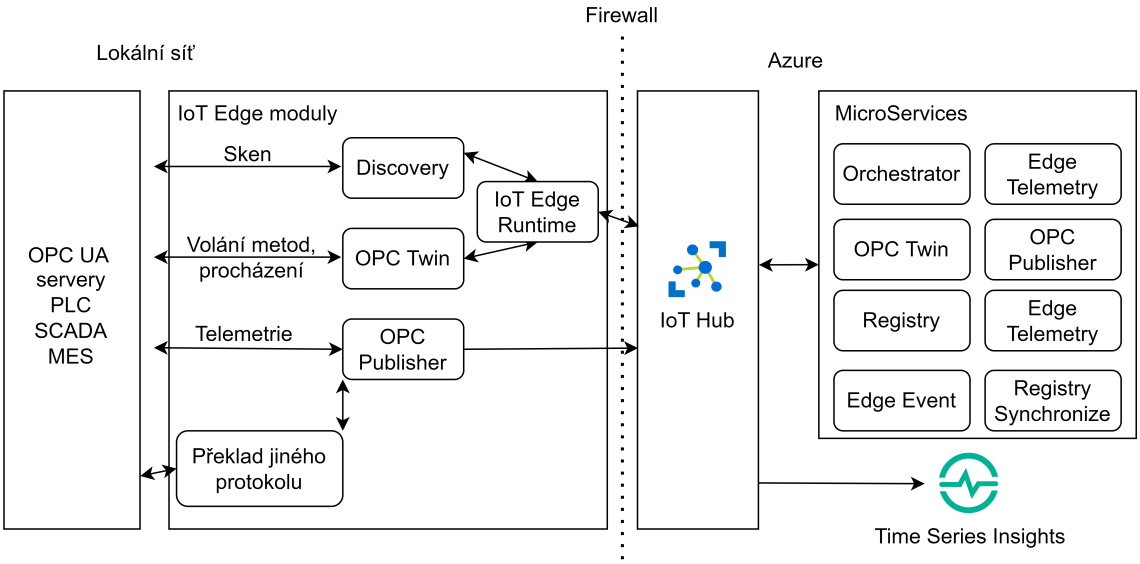
- OPC Publisher Modul,
- OPC Tiwn Modul,
- Discovery Modul.

Využití řešení lze demonstrovat s dodávanou testovací webovou aplikací, jež umožňuje zobrazení všech nasazených IoT Edge zařízení s potřebnými moduly připojených k IoT Hubu. Na vybraném zařízení IoT Edge umožňuje spustit vyhledávání OPC UA serverů v lokální síti. V případě nalezení OPC UA serveru, zobrazit jeho stromovou strukturu. Zobrazit hodnoty v libovolném uzlu OPC UA serveru. Začít monitorovat změny uzlů OPC UA serveru a nastavit na serveru periodu vzorkování. Změny lze následně pozorovat ve službě Time Series Insights [28].

### 2.4.1 Modul OPC Discovery

Modul zajišťuje vyhledání serverů OPC UA v lokální síti. Po vyhledání dostupných serverů společně s Registry Onboarding Processor Microservice zajistí registraci zařízení v IoT Hubu. O zahájení sledování se následně starají další moduly. Modul může být konfigurován pro trvalé aktivní skenování sítě pro skenování vybraných adres.

Funkce Discovery je součástí specifikace OPC UA. Každý server dle specifikace povinně vystavuje základní informace o sobě a podporovaném způsobu navázání spojení (podporovaném zabezpečení).




Obr. 2.2: Schéma architektury Azure Industrial IoT platform [29]

### 2.4.2 Modul OPC Twin

Modul slouží pro operativní procházení OPC UA serveru a jeho uzlů. Tyto uzly je možné zpřístupnit přes OPC Twin Microservice a jeho API. Pomocí tohoto API lze číst i zapisovat hodnoty, volat metody a vyčítat historická data podle definice OPC UA.

### 2.4.3 Modul OPC Publisher

OPC Publisher je modul pro IoT Edge, který umožňuje připojení k OPC UA serveru v lokální síti a přihlášení k odběru změn uzlů OPC UA serveru. Modul má dva režimy běhu:

- Standalone,
- Orchestrated.

Chování v těchto režimech se významně liší.

#### Režim Standalone

V režimu Standalone se konfiguruje pomocí příkazové řádky a konfiguračního souboru *publishednodes.json*. Konfigurační soubor je umístěn ve file sytému a nelze modifikovat pomocí Deployment Manifest z IoT Hubu. V konfiguračním souboru lze pro každý OPC UA server specifikovat adresy uzlů, které se budou odebírat. Každému odebíranému uzlu lze nastavit:

- periodu vzorkování OPC UA serverem (SamplingInterval),

- periodu odesílání změn OPC serverem modulu (PublishingInterval),
- periodu vynuceného odeslání zprávy i pro případy, kdy nenastala změna hodnoty (Heartbeat).

Dále modul může zajistit ukládání zpráv do fronty a odeslání do IoT Hubu až bude velikost zprávy optimální pro odeslání. Tím se sice prodlouží doba doručení zprávy do Cloudu, ale zajistí se celková úspora zpráv, což je vzhledem k cenové politice IoT Hubu výhodné.

Zprávy jsou kódovány vždy formátem JSON. Je možné využít dvou formátů JSON: normou standardizovaný OPC UA PubSub formát, nestandardizovaný a konfigurovatelný formát vytvořený původně pro zpracování v Time Series Insights. Nevýhodou standardizovaného formátu je významně větší velikost [29, 30].

Ve verzi 2.8.2 platformy byla pro režim Standalone znovu přidána možnost konfigurace pomocí IoT Hub Direct methods. Tato vlastnost byla odebrána po verzi 2.5, které skončila podpora na konci roku 2021. Tyto metody umožňují modifikovat konfiguraci v souboru *publishednodes.json*. Podporované metody jsou popsané v [31]. Metody umožňují:

- vyčíst připojené OPC UA servery,
- přidat nebo aktualizovat připojené OPC UA servery,
- vyčíst monitorované uzly na daném OPC UA serveru,
- začít sledovat uzly na OPC UA serveru,
- přestat sledovat uzly na OPC UA serveru.

Argumentem metod je definovaný formát JSON. Návratovou hodnotou je informace o provedení operace, případně odpověď v definovaném formátu JSON [32].

### Režim Orchestrated

V režimu Orchestrated je počáteční konfigurace pomocí příkazové řádky, další konfigurace je spravována z Cloudu pomocí Microservices. Režim počítá s nasazením více IoT Edge zařízení do jedné lokální sítě kvůli redundanci a fail-over. Rozhodování o přidělení úkolů na monitorování má na starosti Orchestrator Microservice v Cloudu. OPC Publisher se v pravidelných intervalech dotazuje na své úkoly a mění podle nich svou konfiguraci. Dotazování konfigurace probíhá staticky přes protokol HTTPS, čímž může být porušen princip jediného komunikačního protokolu mezi zařízením IoT Edge a IoT Hubem. Navíc tento protokol není navržen pro notifikaci změn, v případě okamžité potřeby přerozdělit úlohy tak změna nebude okamžitá. Dále komunikace již neprobíhá přes komunikaci modulů v IoT Edge, ale rovnou do IoT Hubu. Tím je sice snížena latence, nicméně je tím odstraněno ukládání v IoT Edge pro případ výpadku komunikace mezi IoT Edge zařízením a IoT Hubem. Požadavek na ukládání dat v případě výpadku komunikace je uveden v zadání práce.

Konfigurace odebíraných uzlů z OPC UA serverů se zajišťuje přes OPC Publisher Microservice z Cloudu pomocí HTTPS API. Aby bylo kde evidovat rozdělení úloh mezi OPC Publishery je třeba přidat závislost na databázi. Zvolená databáze v řešení je Azure Comsos DB.

Kvůli nutnosti spoléhat na správné organizování úloh pro případ fail-over jednoho IoT Edge zařízení v lokální síti je třeba spolehlivě hostovat v Cloudu Orchestrator Microservice. Doporučeným způsobem je Azure Kubernetes Service [29].

### 2.4.4 Nasazení a testování platformy

Řešení lze nasadit do Azure pomocí skriptu pro Power Shell. Pro funkčnost platformy je třeba nasadit podpůrný software v Azure. Podpůrný software obsahuje minimálně tyto služby z Azure: IoT Hub, Cosmos DB, Service Bus, Event Hub, Key Vault, Storage. Skript umožňuje několik variant nasazení řešení:

- Minimum - nasadí pouze zmíněný podpůrný software v Azure,
- Local - navíc k Minimum přidá několik služeb v Azure; určená pro lokální hostování Microservices,
- Services - navíc k Local přidá hostování Microservices v App Service Plan,
- Simulation - navíc k Minimum nasadí simulaci OPC serverů a IoT Edge do virtuálních strojů,
- App - navíc k Services nasadí testovací webovou aplikaci,
- All - nasadí App a Simulation.

Se samotným instalačním skriptem bylo zaznamenáno několik problémů. Dokumentace [33] doporučuje stáhnout instalační skript pomocí příkazu *git clone* z výchozí větve repozitáře. V této větvi nejsou pouze publikované verze řešení, ale i verze vývojové. Při lokálním hostování a debuggování Microservices docházelo ke kompilačním chybám, protože se jednalo o vývojovou verzi. Řešením bylo vybrat publikovanou verzi z repozitáře. Při pokusu o nasazení verze All se standardním účtem Azure do datacentra North Europe byla opakovaně obdržena hláška o dosažení limitu počtu nasazených virtuálních procesorových jader. Tento počet se liší v různých datových centrech. Řešením bylo nenasazovat verzi Simulation, která nasazuje virtuálních strojů, a hostovat tyto prostředky lokálně.

Testována byla tedy varianta App, nasazená z verze řešení 2.8.2 do datacentra North Europe, ke které bylo připojena lokální zařízení IoT Edge s modulem OPC-PLC Server pro simulaci serveru OPC-UA s náhodně generovanými daty z výchozí konfigurace dle [34]. Ostatní moduly na IoT Edge byly nasazeny dle šablony pro hromadné nasazení s názvem *iiotedge* dokumentované v [35]. Byla ověřena funkčnost sběru dat a testována výchozí uživatelská webová aplikace pro konfiguraci a vizualizování službou Time Series Insights.

Testování probíhalo během čtyř dní na přelomu března a dubna 2022. Na testovacím účtu se toto nasazení projevilo cenou 69,5 EUR, nejvýznamnější položky z ceny jsou v tabulce 2.4. Z rozkladu je evidentní, že největší část ceny jsou služby, které nejsou součástí minimálních požadavků na provoz řešení. V této konfiguraci jsou Microservices nasazeny ve službě App Service, zároveň je shodným způsobem nasazena testovací aplikace.

| Služba | Cena [EUR] |
|---|---|
| Azure Data Explorer Cluster | 34,7 |
| App Service | 9,8 |
| Signal R | 5,9 |
| Time Series Insights | 4,1 |
| Iot Hub | 3,6 |
| Event Hub | 3,0 |
| Storage | 1,4 |

Tab. 2.4: Ceny za nasazení řešení Azure Industrial IoT, položky nad 1 EUR

Obecně celé řešení společně s dokumentací navozuje dojem, že se jedná o stavebnici a jednotlivé komponenty (moduly IoT Edge, Microservices a služby) lze jednoduše přidávat a odebírat. Například u OPC Publisher Microservice lze očekávat, že dochází k přímé konfiguraci modulu OPC Publisher v IoT Edge. Nicméně tento Microservice je nastaven pouze pro spolupráci s modulem OPC Orchestrator Microservice a separace tak, aby běžel pouze OPC Publisher Microservice, není triviální. Z celé části řešení hostované v Cloudu je pro zamýšlený sběr dat vhodná pouze jeho vzdálená konfigurace pomocí HTTPS API z webu. To by nahradilo potřebu připojovat se přes VPN do sítí, kde jsou nasazeny IoT Edge a zde modifikovat nastavení sběru.

Nasazení modulu OPC Publisher v režimu Orchestrated pouze přidává komplexitu a bez zajištění hostování Microservices v Azure Kubernetes reálně nepřidá spolehlivost, protože v řešení se stále nachází single point of failure. Hostovaní ve službě Azure Kubernetes je významně nákladnější než testovaná služba App Service. Selhání Orchestrator Microservice by mělo důsledky pro všechna nasazená zařízení IoT Edge. Naopak v režimu Standalone výpadek na jednotlivém zařízení IoT Edge ostatní zařízení IoT Edge neovlivní.

Dokumentace ([29]) celého řešení je poměrně strohá, obsahuje nefunkční odkazy a je nepřehledně organizována. Nasazení je komentované pouze pro několik úzce zaměřených variant. Možnosti separace jednotlivých komponent, postup pro úpravy řešení a strukturu kódu dokumentace neobsahuje.

Pro zamýšlený účel sběru dat se lépe hodí Standalone režim modulu OPC Publisher. Jediným podporovaným a dokumentovaným způsobem vzdálené konfigurace v tomto režimu je pomocí IoT Hub Direct Methods. Nad těmito metodami je možné vyvinout webovou aplikaci, která by tuto konfiguraci zajišťovala. Alternativou je modifikovat soubor *publishednodes.json* pomocí vlastního modulu, který by tento soubor přepisoval například podle konfigurace z externí databáze. Externí databází by byla také zajištěna záloha konfigurace modulu na externím zařízení.

### 2.4.5 Verze a podpora

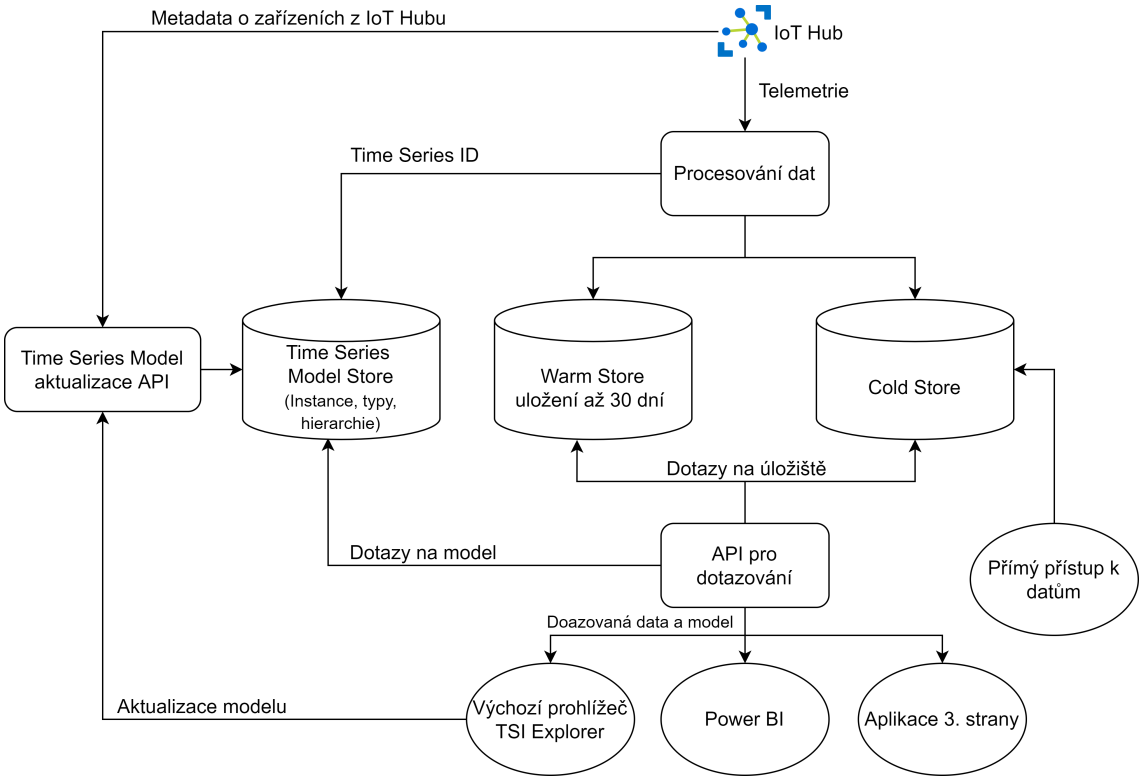
Aktuální verze je 2.8, poslední dostupný release 2.8.2. Jedná se o verzi LTS. Verze LTS mají podporu dva roky od jejich prvního vydání. Konec podpory této verze je naplánován na 15. 7. 2023.

## 2.5 Azure Time Series Insights

Time Series Insights (TSI) je Cloudová služba typu SaaS, která slouží pro analytiku, ukládání a vizualizaci dat v časových řadách. Služba se nyní poskytuje ve dvou generacích. První generace je nyní pouze udržována, druhá generace je hlavní vývojová větev; tato kapitola se dále zabývá druhou generací. Dostupnost této služby není ve všech datových centrech Azure v Evropě, nicméně je dostupná v datacentrech North Europe a West Europe.

Na obrázku 2.3 je znázorněn detail služby. Vstupními daty je telemetrie z koncových zařízení, která přichází z IoT Hubu. TSI získává data z IoT Hubu přes Event Hub Endpoint. Následně jsou data zprocesována a uložena do úložiště. Výchozí formát zpracovávaných dat je JSON. Hlavní činností procesování je zpracování dat do hierarchického modelu. TSI se nabízí se dvěma druhy úložišť Cold Store a Warm Store. Nezbytné je vždy úložiště Cold Store. Warm Store ukládá data po dobu maximálně 30 dní, tuto dobu lze zvolit při zakládání služby. Po uplynutí této doby jsou data automaticky přesouvána do Cold Store. Pokud není Warm Store aktivní, data se ukládají přímo do Cold Store. Výhodou Warm Store je vyšší rychlost zpracování dotazů. Další přednosti proti Cold Store vychází z cen služby. Nevýhodou Warm Store je, že k jeho datům nelze zatím přistupovat jinak než prostřednictvím API pro dotazování.

Služba zpřístupňuje data pomocí webového API. Autentizace k API se provádí pomocí Azure Active Directory. Data lze vizualizovat pomocí výchozí webové aplikace TSI Explorer. Alternativně lze využít pro vizualizaci službu Power Bi, případně integrovat vizualizace do webové aplikace třetí strany pomocí dodaných JavaScriptových knihoven. Power Bi má výhodu v nativním zobrazení na mobilních zařízeních




Obr. 2.3: Diagram služby Time Series Insights, přeloženo z [36]

s operačními systémy Android a iOS, pro které poskytuje aplikaci. K datům lze také přistupovat přímo připojením k úložišti Cold Store.

### 2.5.1 Datový model

Při založení služby je třeba zvolit TimeSeriesId. Jedná se o povinný parametr – unikátní primární klíč každé instance (záznamu). Příkladem instance může být zařízení. Instance může mít několik vlastností tzv. Property. Příkladem vlastnosti je rychlost otáček a teplota motoru. Dalším povinným parametrem pro zpracování zprávy je Timestamp, který definuje čas pořízení zprávy. Způsob nastavení je lépe patrné z praktické implementace v další části.

### 2.5.2 Cena za službu

Ceník služby je v tabulce 2.5. Za provozování služby v Azure je třeba každý měsíc platit „Jednotku zpracování dat“. Tato cena se započítá každý měsíc provozování služby i pokud služba nezpracovává žádná data. V rámci této ceny je procesování až 100 GB vstupních dat. Nad 100 GB dat jsou účtovány poplatek „Další zpracovaná

data“ za každý další GB. Dále je účtováno za „Úložiště metadat“ v Time Series Model Store a úložiště Warm Store a Cold Store.

Na úložišti Warm Store lze bezplatně provádět neomezené množství dotazů, platí se pouze měsíční cena za GB úložiště. Oproti tomu se na Cold Store účtuje za jednotku dat, která je analyzována. Cena úložiště Cold Store se účtuje podle ceníku služby Blob Storage a je obvykle výrazně nižší (i řádově) než cena za Warm Store.

| | Jednotka | Cena za měsíc [EUR] |
|---|---|---|
| Jednotka zpracování dat | [100 GB] | 31,557 |
| Další zpracovaná data | [GB] | 0,215 |
| Úložiště metadat | [MB] | 0,043 |

| | Analýza Hot Store | Analýza Cold Store |
|---|---|---|
| Úložiště | 3,216 EUR za GB | Podle ceníku Blob Storage |
| Dotazy | Bez poplatku | 0,011 EUR za GB |

Tab. 2.5: Ceny za Time Series Insights [37]

## 2.6 Výběr hardware pro provozování IoT Edge

Microsoft certifikuje zařízení pro provozování IoT Edge na žádost výrobce. Certifikovaná zařízení jsou pak zveřejněna v seznamu na webu Azure Certified Device catalog [38]. Z tohoto seznamu pak lze vybírat zařízení pro produkční nasazení. K 20. 12. 2021 seznam obsahuje 269 certifikovaných zařízení.

### Raspberry Pi

Doporučenou vývojovou platformou pro provozování IoT Edge je Raspberry Pi. Výchozí operační sytém Raspberry Pi OS Stretch je ve třídě 1 podporovaných operačních systémů pro IoT Edge. Microsoft poskytuje příklady jak využít tento hardware k provozování IoT Edge.

Raspberry Pi jsou jednodeskové počítače s procesory architektur ARMv6, ARMv7 a ARMv8. Hardware Raspberry Pi je na trhu v několika verzích. Pro vývoj a provoz je třeba volit kompromis mezi cenou, výpočetním výkonem, příkonem a konektivitou. Pro zajištění konektivity s PLC budou vybírány verze s konektorem Etehernet, tedy Modely B. V tabulce 2.6 jsou uvedeny verze, které vyhoví ostatním kritériím. Všechny tyto modely mají čtyři procesorová jádra architektury ARMv8 64-bit. Příkony energie v tabulce 2.6 jsou uváděny ve stavu Idle, tedy bez zatížení.

K cenové náročnosti je nutné dále připočítat krabičku pro zavěšení na nosnou lištu v rozvaděči, zdroj a SD kartu. Tyto náklady se v závislosti na modelu Raspberry Pi příliš neliší, pouze k Modelu 4 je třeba uvažovat nad výkonnějším zdrojem. Ceny jednotlivých modelů jsou nyní ovlivněny světovým výpadkem dostupnosti polovodičových součástek [39]. U všech těchto modelů výrobce Raspberry Pi Foundation uvádí možnost zakoupení do ledna roku 2026 [40]. Model 3B výpočetním výkonem stačí na úlohu pořízení, zpracování obrazu, vyhodnocení a odeslání do Cloudu, která běží v modulech na IoT Edge [41]. Z toho lze usuzovat, že pro vyčítání dat z PLC, filtrování, odesílání do Cloudu a běh několika modulů IoT Edge by měl také postačovat.

| Model | Frekvence jádra [GHz] | RAM [GB] | Cena [Kč] | Příkon [W] |
|---|---|---|---|---|
| 3B | 1,2 | 1 | 949 | 1,5 |
| 3B+ | 1,4 | 1 | 1039 | 1,9 |
| 4 | 1,5 | 1 | 949 | 2,7 |
| 4 | 1,8 | 2 | 1249 | 2,7 |
| 4 | 1,8 | 4 | 1539 | 2,7 |

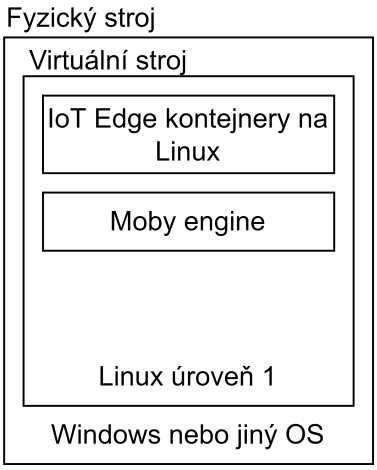
Tab. 2.6: Srovnání specifikací Raspberry Pi [42], [40]

### 2.6.1 Průmyslové počítače

IoT Edge lze provozovat na standardním PC nebo průmyslovém PC. Při instalacích průmyslových PC společností ACAM ostře převažují stroje s operačním systémem Windows. Nativní distribuce kontejnerů s IoT Edge pro Windows je ukončena s verzí 1.1 LTS. V dalších verzích budou podporovány pouze kontejnery pro Linux.

Pro kompatibilitu IoT Edge s Windows se doporučuje využít instalace distribuce operačního systému Linux z úrovně 1 podpory jako virtuální stroj nad Windows [25]. Ve virtuálním stroji je pak provozován kontejnerový engine Moby a moduly IoT Edge.

Alternativou je využití obrazu virtuálního stroje Linux přímo určeného pro nasazení IoT Edge na Windows. Ten je postavena na CBL Mariner Linux a nazývá se Azure EFLOW. Část instalace a nastavení lze spravovat přímo z Windows pomocí konzole Power Shell. Pro kompletní správu je nutné použít připojení přes protokol SSH k virtuálnímu stroji a další administraci obsloužit přes něj. EFLOW je nyní k dispozici ve dvou verzích, které odpovídají životnímu cyklu IoT Edge. Verze EFLOW 1.1 LTS, která podporuje instalaci IoT Edge verze 1.1. Verze 1.2, která podporuje IoT Edge verzi 1.2, je ve stádiu vývoje public preview [43].




Obr. 2.4: Způsob provozování IoT Edge na Windows, přeloženo z [25]

# 3 Databázové a vizualizační systémy

Cílem této kapitoly je vytvořit přehled databází, které je vhodné využít pro ukládání stavových veličin, řídicích dat a událostí z průmyslových zařízení. Tato data jsou charakteristická tím, že ke každému záznamu mají přiřazenu časovou značku. Přednostně tedy budou vybírány databáze, které jsou tomuto charakteru dat uzpůsobené.

Data z databáze bude třeba vizualizovat uživatelům. Vizualizace by měla zajistit autentizaci a autorizaci uživatele. Využití vizualizace je plánováno pro vnitřní potřeby firmy ACAM Solution, tak pro potřeby zákazníků. Vizualizace by měla být kompatibilní s vybranou databází.

## 3.1 Databáze Time Series

Time Series databáze jsou určené k ukládání dat, které obsahují časové značky a hodnoty. Mimo hodnot záznam může obsahovat značky (tags), které se využívají k seskupení a řazení. Data ukládaná do těchto databází mohou mít tyto vlastnosti:

- jsou ukládána v chronologickém pořadí jejich vzniku,
- interval mezi jednotlivými záznamy má stejnou velikost,
- jedná se o měřená data, tedy hodnoty po zápisu neměnné.

Time Series databáze mají obvykle tyto charakteristiky:

- jsou optimalizované pro zmíněné charakteristiky dat,
- data zapsaná do databáze není možné zpětně editovat,
- smazání jednotlivé hodnoty není možné,
- při ukládání se využívá kompresních metod, které snižují potřebu diskových operací a diskového prostoru,
- lze nastavit agregaci, výmaz po uplynutí určité doby archivace,
- bývají spojeny se systémy notifikací a alarmování.

Hlavními optimalizacemi proti klasickému prostému zápisu dat na disk v binárním souboru je komprese dat. Typickým využitím těchto databází je IoT, kde je třeba právě centralizovat měřená data z různých zařízení a DevOps, tedy zaznamenávání aktivity a vytížení na nasazených výpočetních technologiích (serverech, routerech, kontejnerech).

Počet dostupných Time Series databází je 34 [44]. InfluxDB je podle výzkumu v [44] nejpopulárnější Time Series databáze mezi vývojáři. Srovnání InfluxDb s ostatními Time Series databázemi se věnuje práce [45]. Pro srovnání jsou využita veřejně dostupná data jízd taxi služby z New Yorku. Tato data jsou importována do srovnávaných databází a nad testovacím datasetem je vytvořeno 12 databázových dotazů. Kvůli charakteru importu dat není porovnávána časová náročnost zápisů.

Testování proběhlo na standardním notebooku. Použitá verze databáze InfluxDB je 1.3. Srovnání je provedeno s MSSQL serverem verze 13.4, tedy běžnou relační databází. Data v InfluxDB zaberou 27× méně místa na disku. Průměrná doba vykonávání dotazu je také výrazně nižší u InfluxDB. Další srovnání je v práci převzato z oficiálního testovacího nástroje InfluxDB proti ostatním databázím. Z těchto srovnání lze konstatovat, že si InfluxDB nevede proti ostatním Time Series databázím špatně. Nicméně test byl prováděn v roce 2017, od té doby lze očekávat vývoj v Time Series databázích. Testovací algoritmus včetně testovacích dat byl navržen přímo vývojáři InfluxDB, proto jej nelze brát jako zcela relevantní.

Alternativou porovnání Time Series databází nabízí článek [46]. V článku je uveden návrh algoritmu pro testování jejich výkonu. K tomuto účelu jsou použita zaznamenaná data z provozu farmy větrných elektráren. Opět je navržena sada testovacích databázových dotazů a jsou zaznamenávány nároky na výpočetní výkon, diskové úložiště a další. Na rozdíl od předchozího testu je porovnávána i rychlost zápisu. Testování je provedeno na stroji s diskem HDD s rotující plotnou o 7200 otáčkách za minutu. Porovnávanými databázemi jsou InfluxDB (v 1.7), Timescale DB, Druid a OpenTSDB. Shrnutím srovnání pro InfluxDB jsou:

- nejlepší poměr komprese uložených dat ze všech srovnávaných databází,
- dobrá rychlost zápisu na úkor speciální techniky ukládání dat,
- díky vytvořenému indexu dobrá rychlost čtení.

Lze tedy konstatovat, že InfluxDB intenzivně komprimuje data, při zápisu na to využívá značné množství výpočetních prostředků. Výsledkem je výborná úroveň komprese. Díky ní je nutné při čtení použít méně vstupně-výstupních operací, což obecně zrychlí vybavení dat z disku. K dekompresi po čtení je následně nutné použít opět vyšší množství výpočetního výkonu.

### 3.1.1 Time Series Insights

Time Series Insights je databázový a vizualizační systém, který lze provozovat jako SaaS v Microsoft Azure, je popsán v kapitole 2.5.

### 3.1.2 InfluxDB

InfluxDB je Time Series databáze napsaná v programovacím jazyce Go. Rozdíl proti některým Time Series databázím je schopnost uchovávat události, které přichází v zcela náhodných okamžicích mimo pravidelné intervaly vzorkování. Pro ukládání se používá algoritmus Time-structured merge tree, což je modifikovaný algoritmus Log-structured merge tree.

Operace s databází lze provádět dvěma databázovými jazyky. Historicky starším jazykem je InfluxQL, který je uzpůsoben pro dotazování databáze ve verzi 1.x. Pro

verze 2.x byl vytvořen jazyk Flux. InfluxQL připomíná syntaxí jazyk SQL. Kvůli změně datového modelu mezi verzemi 1.x a 2.x nejsou všechny příkazy jazyka InfluxQL použitelné ve verzi 2.x. Verze 1.x jsou nyní pouze udržované, aktivní vývojová větev je 2.x. Pro tuto práci bude použita verze 2.x.

K samotné databázi, tedy k softwaru pro ukládaní a vyčítaní dat, je součástí distribuce webová aplikace pro analýzu dat, vizualizaci a alarmování.

### Datový model

Schéma zapisovaných dat není třeba deklarovat předem jako u relačních databází. Data lze zapsat a databáze obstará vytvoření obsluhu vzniku datového modelu za běhu.

Jak jsou v databázi organizovány data do schématu je vhodné demonstrovat na příkladu. Existují 3 typy základní anotace ve schématu: Measurement, Tag a Field. Předpis schématu pro měření z senzoru vzduchu a senzor kvality vody je v tabulce 3.1

| Measurement | Tag | Tag | Field | Field |
|---|---|---|---|---|
| Vzduchovy_senzor | Id_senzoru | Stanice | Vlhkost | Teplota |
| Kvalita_vody_senzor | Id_senzoru | Stanice | PH | Teplota |

Tab. 3.1: Příklad schématu InfluxDB

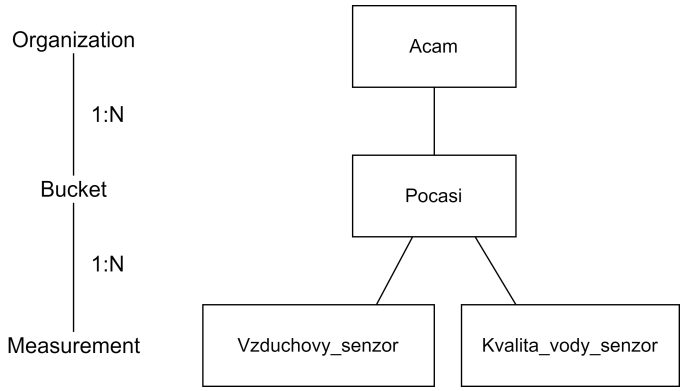
Příkladem zápisu hodnot do schématu řádkovým protokolem podporovaným podporovaným přes HTTPS API je

```
Vzduchovy_senzor,Id_senzoru=A0100,Stanice=Harbor,
Vlhkost=35.0658,Teplota=21.667 časová značka
```

Povinným parametrem každého zápisu je název Measurement. Data jsou pak uspořádána pomocí Tags a Fields, kdy ke každému z nich je přiřazena hodnota. Zásadní je rozdíl mezi Tags a Fields je, že hodnoty zapsané do Tags jsou indexované ve speciální tabulce. V případě příkladu tedy název stanice Harbor je indexován. Hodnoty Field je naopak prostý zápis hodnoty. Názvy Field a Tags jsou také indexovány, název Tagu je v uvedeném příkladu třeba Stanice [47].

Všechny volby značení Tag a Field je třeba dobře promyslet před prvním zápisem. Po prvním zápisu jsou záměny mezi Tag a Field komplikované.

Každý Measurement povinně náleží do Bucket. Funkce Bucket je je organizovat Measurement do skupin. Ke každému Bucket je přidělena tzv retention period, tedy doba po které jsou automaticky odstraněna. Tato doba může být také nastavena na nekonečno. Každý Bucket povinně náleží k Organization. Organization slouží k správě uživatelů a jejich oprávnění. Pro ilustraci jsou vazby naznačeny na obrázku 3.1.




Obr. 3.1: Organizace dat v InfluxDB

### Distribuce

Software je distribuován v těchto verzích:

- open source (OSS),
- Enterprise,
- Cloud.

První verze má otevřený zdrojový kód pod licencí MIT. Jsou k ní také k dispozici kompilované instalační soubory pro operační systémy Linux, macOS, Windows. Verze Enterprise má uzavřený zdrojový kód a je určena pro individuální nasazení na vlastní hardware a cena licence je určena individuálně podle specifikace použitého hardwaru. Enterprise verze umožňuje nasadit databázi v clusteru, tedy v režimu kdy je databáze provozována na více strojích současně a kvůli této redundaci je více odolná vůči výpadku. Dále umožňuje pokročilé autorizace a autentizace uživatele např. pomocí Active Directory.

Poslední možností je nasazení v Cloudu. Jedná se o službu typu SaaS. Hostování je zajištěno ve vybraných datacentrech Microsoft Azure, Google Cloud Platform nebo Amazon Web Services. Konkrétně v Evropě je pro Azure k dispozici hostování v data centru West Europe. Pro hostování v Cloudu se využívá modifikovaná varianta verze Enterprise. Verze je hostována v režimu cluster. Měla by tedy poskytovat vyšší spolehlivost než varianta OSS. Varianta Cloud je nabízena v omezené bezplatné variantě, variantě účtování dle využití a ročních předplacených plánech. Varianta zdarma je omezena maximální dobou uložení dat na 1 měsíc, maximální rychlostí zápisu dat, počtem Bucketů. U varianty účtování dle využití jsou sledovanými parametry využití přijatá data, provedené dotazy, použité úložiště a odchozí data. Ceník je shodný pro všechny datacentra a je uveden v tabulce 3.2. Varianta ročního předplatného je naceněna individuálně na vyžádání. Jako jediná taky poskytuje smlouvu s garancí poskytování služeb (SLA) [48].

| Metrika | Jednotka | Cena [USD] |
|---|---|---|
| Přijatá data | [MB] | 0,002 |
| Provedené dotazy | [100 provedení] | 0,01 |
| Použité úložiště | [GB×hodina] | 0,002 |
| Odchozí data | [GB] | 0,09 |

Tab. 3.2: Cena Cloudu ve variantě účtování dle použití [48]

#### Nástroje pro komunikaci

Komunikace s databází InfluxDB je zajištěna pomocí síťového protokolu aplikační vrstvy HTTP nebo HTTPS. Nad tímto protokolem je vystaveno REST API pro čtení a zápis do databáze. Pro zápis lze využít jednoduché volání typu POST bez nutnosti využití jazyka Flux. Pro dotazování a další operace s databází je využití jazyka Flux již nutné. Nad tímto HTTPS API je vystavěna řada nástrojů pro programátory, které komunikaci usnadňují. Jedná se o CLI a sadu knihoven pro různé programovací jazyky. Pro tuto práci je zásadní knihovna pro jazyk C#. Knihovna umožňuje anotovat v běžném objektu typu třída příznaky pro zápis do databáze InfluxDB. Tedy, zda se jedná o Measurement, Field nebo Tag. Dále umožňuje obsloužit asynchronní zápis kolekce Measurement do databáze [49].

#### Autentizace a autorizace

Autentizace se liší mezi distribucemi. Obecně platí, že uživatelé jsou vždy přiděleny k Organization. Uživatelé se autentizují pomocí uživatelského jména a hesla. V distribuci OSS jsou uživatelské účty spravovány pomocí CLI.

Ve verzi Cloud lze přidávat uživatele prostřednictvím webové aplikace pozváním do organizace. Uživatel je následně vyzván prostřednictvím e-mailu k vytvoření uživatelského účtu.

Pro přístup aplikací pro zápis nebo čtení dat je možné vygenerovat token pro přístup k API. Přístup pak lze volit k jednotlivým Bucketům, pro každý Bucket lze volit oprávnění pro čtení nebo zápis.

## 3.2 Vizualizace

### 3.2.1 Influx

Součástí všech distribucí InfluxDB je webová aplikace pro vizualizaci a konfiguraci. Tato aplikace je vystavěna nad HTTPS API. Je dostupná na stejném TCP portu

jako API. Aplikace umožňuje tvorbu nástěnek a operativní procházení dat. Pro práci je zásadní tvorba vizualizace a nástěnek.

Grafická a funkční podoba aplikace se mírně liší mezi distribucemi. Uživatelské účty z InfluxDB jsou shodné pro přístup k aplikaci.

### 3.2.2 Grafana

Grafana je nástroj pro webovou vizualizaci dat. Podobně jako vizualizace Influx je možné tvořit nástěnky a operativně dotazovat data. Grafana podporuje řadu databázových zdrojů. Příkladem jsou MYSQL, MSSQL, InfluxDB.

#### Distribuce

Grafana má podobný způsob distribuce jako InfluxDB. Poskytuje varianty

- open source,
- Enterprise,
- Cloud.

Verze open-souce je uvolněna pod licencí AGPLv3. Opět se nabízí instalační soubory pro řadu operačních systémů. Verze Enterprise poskytuje prémiové funkce pro korporace. Jmenovitě přidává další databázové zdroje jako SAP Hana. Hostování opět probíhá na vlastních serverech a je k dispozici profesionální podpora. Verze Cloud je hostována v Cloudu jako služba SaaS.

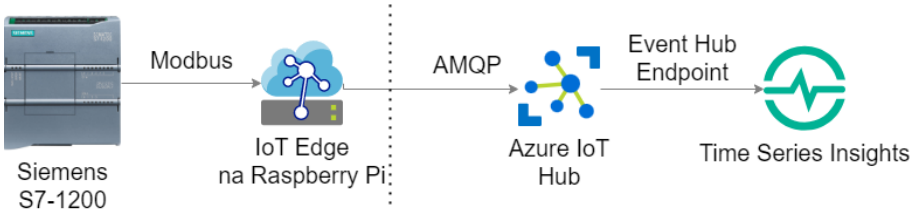
#### Autentizace a autorizace

Autentizovat uživatele je možné pomocí lokálně uložených uživatelských účtů, případně lze integrovat s Azure Active Directory nebo jinými poskytovateli OAuth2 autentizace [50].

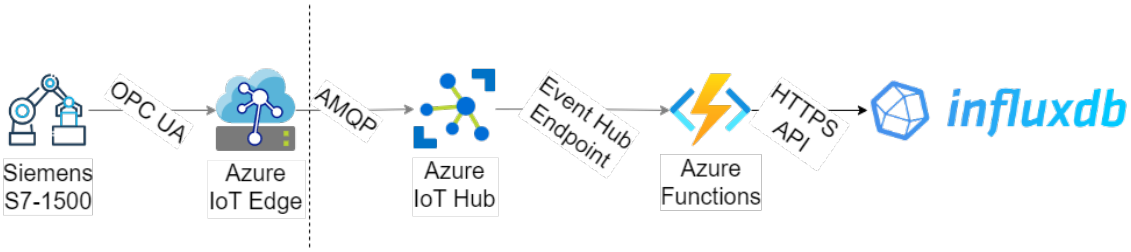
# 4 Implementace řešení v Microsoft Azure

V této kapitole je rozebrána implementace dvou řešení. První řešení je implementováno dle blokového schématu na obrázku 4.1. Řešení využívá pro sběr dat z PLC protokol Modbus, pro provozování IoT Edge se využívá Raspberry Pi a pro vizualizaci a ukládání dat službu Time Series Insights. Blokové schéma druhého řešení je na obrázku 4.2. Řešení využívá pro sběr dat z PLC protokol OPC UA, pro provozování IoT Edge se používá PC s OS Windows, jako databáze InfluxDB a pro vizualizaci Influx a Grafana.

Obě řešení pro příjem dat v Cloudu Azure používají IoT Hub a pro připojení k IoT Hubu software IoT Edge. Ačkoliv by bylo možné pro obě řešení využít jednu instanci IoT Hubu, byla pro každé řešení využita vlastní instance. Dále jsou popsány zkušenosti s provozováním obou řešení, jejich výhody a nevýhody.




Obr. 4.1: Blokové schéma prvního řešení, ikona PLC převzata z [1]




Obr. 4.2: Blokové schéma druhého řešení

## 4.1 První řešení

Pro testování bylo vybráno PLC, které ovládá žaluzie a termostaty, jeho činnost a datový model je popsána v [4]. Z datového modelu lze zvýraznit předpis pro žaluzii v tabulce 4.2 a termostat v tabulce 4.1. Protokol Modbus nativně nepodporuje reorientaci čísel s plovoucí řádovou čárku. Proto jsou číselné hodnoty teploty reprezentovány jako typ uint16 s pevnou řádovou čárkou.

| Adresa | Název | Význam |
|---|---|---|
| $n$ | CurrentTemperature | Aktuální teplota, [°C] |
| $n+1$ | TargetTeperature | Cílová teplota, [°C] |
| $n+2$ | ValveState | Stav ventilu, [-], 0 zavřeno, > 1 otevřeno |
| $n+3$ | CurrentHeatingCoolingState | Režim termostatu, výčet hodnot |

Tab. 4.1: Předpis datového modelu pro termostat

| Adresa | Název | Význam |
|---|---|---|
| $n$ | CurrentPosition | Aktuální pozice, [-], 0 – 100 |
| $n+1$ | TargePosition | Žádaná pozice, [-], 0 – 100 |
| $n+2$ | CurrentHorizontalTiltAngle | Úhel natočení lamely, [°], 0 ° – 90 ° |

Tab. 4.2: Předpis datového modelu pro žaluzii

### 4.1.1 IoT Hub

Služba byla zřízena v datacentru North Europe. Úvodní konfigurace byla následující:

- nastavení konektivity na *Public acess*,
- cenová kategie na *F1: Free*,
- služba *Defender for IoT* deaktivována,
- přístupová práva na *Shared acess policy + Role Based Access Control*.

### 4.1.2 IoT Edge

Pro testování IoT Edge byla vybrána zařízení Raspberry Pi Model 3B+ a 3B. Všechny testovací instance byly osazeny SD kartou s obrazem Raspberry Pi OS Lite Release 2021-05-07. Nahrání obrazu na SD kartu proběhlo pomocí nástroje Raspberry Pi Imager. Pomocí tohoto nástroje bylo na zařízení konfigurováno spuštění SSH serveru, výchozí uživatel pi a jeho heslo, lokalizace. Postup instalace vychází z [51]. Další konfigurace již probíhala přes protokol SSH.

Pro instalaci software IoT Edge na zařízení byl použit postup uvedený v [52]. Postup je navržen pro vytvoření IoT Edge zařízení s již vytvořeným IoT Hubem pomocí webového prohlížeče a portálu Azure. Autentizace zařízení k IoT Hubu je zajištěna pomocí symetrického klíče. Postup zajistí:

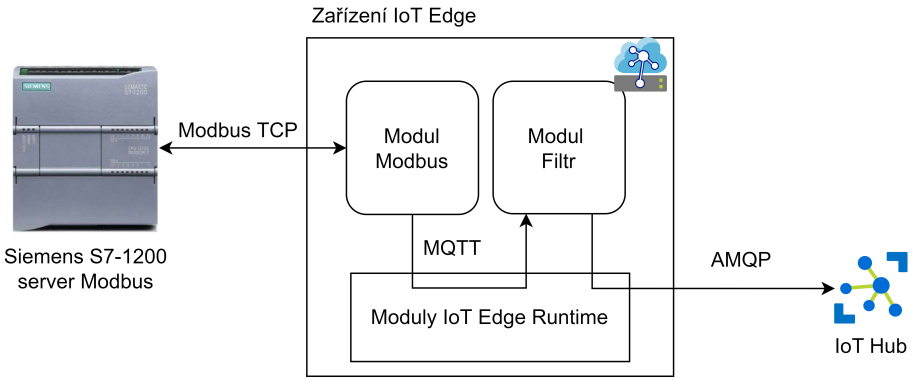
- vytvoření předpisu nového zařízení IoT Edge v Iot Hubu a přidělení autentizačních klíčů,
- přidání veřejného klíče serveru *https://packages.microsoft.com* do instalátoru balíků APT (Advanced Packaging Tool),

- přidání obsahu serveru *https://packages.microsoft.com* jako zdroje instalátoru balíků APT,
- instalace kontejnerového engine Moby,
- instalace softwaru IoT Edge,
- konfigurace *connection string* pro přístup k IoT Hubu se symetrickým klíčem pro autentizaci,
- spuštění IoT Edge,
- automatické spuštění po startu zařízení.

Deployment Manifest celého zařízení je uveden v příloze D.

### Modul Modbus

Microsoft jako příklad využití zařízení IoT Edge pro překlad protokolů z koncových zařízení poskytuje modul, který zajišťuje překlad protokolu Modbus [53]. Způsob využití je naznačen na obrázku 4.3. Modul odesílá požadavky na čtení na koncové zařízení. Zpracované odpovědi od koncových zařízení odešle protokolem MQTT ke zpracování v IoT Edge runtime. IoT Edge runtime zajistí odeslání přes protokol AMQP do IoT Hubu.




Obr. 4.3: Schéma nasazení IoT Edge, převzato z [53]

Modul umožňuje komunikovat s Modbus zařízením po sériové lince nebo přes protokol TCP/IP. Konfigurace modulu umožňuje nastavit IP adresu a port koncového zařízení, čtené adresy nebo skupiny adres v registrech Modbus, interval dotazování koncového zařízení, interval odesílání zpráv. Příklad konfigurace je uveden v příloze A.1 pod názvem Modbus. Příklad výstupu z modulu je uveden v příloze B. Výstup z modulu je velmi obsáhlý a obsahuje velké množství redundantních nebo nevýznamných údajů jako je adresa registru. Dále v tomto formátu byl poměrně složitý pro vytvoření datového modelu služby Time Series Insights. Z tohoto důvodu je třeba data před odesláním do IoT Hubu filtrovat.

#### Modul filtrování dat

Modul filtrování dat je vytvořen v jazyce C# nad frameworkem .NET Core 3.1. Pro programování bylo použito vývojové prostředí Visual Studio Code s rozšíření Azure IoT Tools. Při tvorbě modulu bylo vycházeno z [54] a [26].

Modul zajistí změnu struktury dat z výstupu modulu Modbus tak, aby odpovídala datovému modelu připojených zařízení k PLC. Modul byl navržen přímo pro filtrování předem známé struktury JSON. Ke každému zařízení je přidána vlastnost *DeviceId*, která určuje název zařízení a je využita při parsování ve službě Time Series Insights.

Modul byl zkompilován a publikován do Azure Container Registry. Na vstup modulu je přiveden výstup modulu Modbus. Výstup modulu je směrován do Cloudu do IoT Hubu.

Příklad výstupu modulu je uveden v příloze B.2.

### 4.1.3 Time Series Insights

Služba byla zřízena v datacentru North Europe. Úvodní konfigurace byla následující:

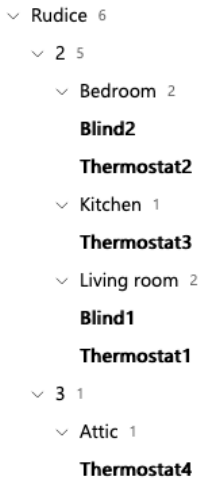
- Time series ID nastavena na *devices.DeviceId*,
- Timestamp property nastavena na *devices.timeCreated*,
- Warm storage aktivní, doba ukládání 14 dní,
- Cold Storage jako StorageV2 (general purpose), Locally redundant storage,
- byl zřízen IoT Hub Endpoint s právy *iothubowner*.

Pro zpracování byly vytvořeny předpisy *Type definitions* pro připojená zařízení k PLC, tedy termostaty a žaluzie (Thermostats, Blinds). Tyto definice odpovídají popisu v úvodu kapitoly. V definici termostatů bylo zařízeno přetypování číselných záznamů s teplotou na záznamy čísel s plovoucí řádovou čárkou a její umístění. Dále je záznam *ValveState* převeden do typu *Categorical*, který nabývá hodnot *On* a *Off*, případně pro neznámé hodnoty *N/A*.

Celkově jsou k dispozici čtyři zařízení typu termostat a dvě zařízení typu žaluzie. *Hierarchy definitions* byly vytvořeny tak, aby odpovídaly fyzickému umístění zařízení. Kořenovou úrovní je název umístění, nižší úrovní číslo patra, poslední úrovní místnost. Přiřazení do hierarchie je vázáno na *Time Series Id* každého zařízení. Jedno *Time Series Id* lze přiřadit do více hierarchií. Výsledná hierarchie je zobrazena na obrázku 4.4.

#### Autentizace a autorizace

Koncoví uživatelé portálu Time Series Insights se autentizují pomocí Azure Active Directory. Uživatelé služby mohou být přidávaní pouze z Azure Active Directory,




Obr. 4.4: Hierarchie zařízení ve službě Time Series Insights

ke které patří *Subscription*, v němž je služba provozována. Pro studenty VUT, kteří využívají nabídku Azure for Students, se automaticky zřídí *Subscription* s názvem Azure for Students v Active Directory VUT. V Azure lze následně vytvořit novou Active Directory a *Subscription* do ní přesunout. V Azure Active Directory lze pak vytvořit uživatele. Následně lze v administraci služby Time Series Insights přidělit těmto uživatelům přístup ke službě. Pro účely testovacího nasazení bylo *Subscription* ponecháno v Active Directory VUT. Testování přihlašování jiných uživatelů proběhlo s dalšími účty v této Active Directory.

Pro uživatele lze volit ze dvou rolí: Reader a Contributor. Reader může zobrazovat veškerá data. Nemůže modifikovat datový model. Neexistuje možnost volit přístupová práva k datům na základě uživatele.

#### Vizualizace

Podoba vizualizace je naznačena na obrázcích, které jsou součástí přílohy, viz příloha D. Cena byla získána podle strženého kreditu na účtu v Azure.

### 4.1.4 Cena testovacího provozu

Provoz řešení probíhal od 1. 12. 2021 do 31. 12. 2021. Cenu jednotlivých komponent za toto období shrnuje tabulka 4.3. Z tabulky je patrné, že z účtovaných položek pro službu Time Series Insights ostře převažuje Jednotka zpracování dat. Tento konstantní paušál je nutné platit za každý měsíc provozu služby. Službu není možné na určitou dobu pozastavit a vyhnout se účtování poplatků.

| Služba | Druh | Cena v USD |
|---|---|---|
| Time Series Insights | Jednotka zpracování dat | 30,0 |
| Time Series Insights | Úložiště metadat | 0,31 |
| Time Series Insights | Warm Store | 0,03 |
| Time Series Insights | Dotazy Cold Store | 0,00 |
| Úložiště | Hot LRS Write Operations | 0,42 |
| Úložiště | Hot Read Operations | 0,11 |
| Container registry | Úroveň Basic | 4,67 |
| IoT Hub | Úroveň F1: Free | 0,00 |
| | Celkem | 35,54 |

Tab. 4.3: Cena prvního řešení za kalendářní měsíc

### 4.1.5 Diskuze prvního řešení

Celé řešení je nasazení v Cloudu ve formě služeb PaaS. Hlavním nedostatkem řešení je služba úložiště a vizualizace Time Series Insights. Služba je v Cloudu Azure jako výchozí a doporučená pro vizualizaci dat v časových řadách z IoT zařízení. Zároveň je vyvíjena společností Microsoft, což zajistilo nativní napojení na službu IoT Hub a možnost čerpání prostředků na provozování z programu Azure for Students. Služba je přímo kompatibilní s IoT Hubem, není třeba data překládat. Služba neumožňuje filtrovat přístup k datům podle uživatele. Pro komerční nasazení firmou ACAM Solution by tedy bylo nutné pro každého zákazníka zřídit samostatnou instanci Time Series Insights, do které by byla směrována příslušná data pomocí IoT Hubu. Služba je zdaleka nejnákladnější z celého řešení na měsíční provoz. Služba také neumí moc dobře vizualizovat diskrétní události jako jsou alarmy apod. V dokumentaci služby se po ukončení testování řešení objevila zpráva, že její podpora bude ukončena k 31. 3. 2025. Pro uživatele služby se doporučuje migrovat na službu Azure Data Explorer. Tato služba je i v nejlevnější variantě nasazení výrazně nákladnější než služba Time Series Insights.

Služba IoT Hub v kombinaci se software IoT Edge se osvědčila. Možnost nasazovat, konfigurovat software z portálu Azure bez nutnosti využití lokálního přístupu nebo přístupu přes VPN je užitečný. IoT Hub je užitečný pro zajištění zabezpečeného připojení koncových zařízení.

Jako ne příliš vhodný se ukazuje protokol Modbus, který je třeba konfigurovat na Siemens PLC a monitorované proměnné kopírovat do datového bloku, který je přístupný pro server Modbus.

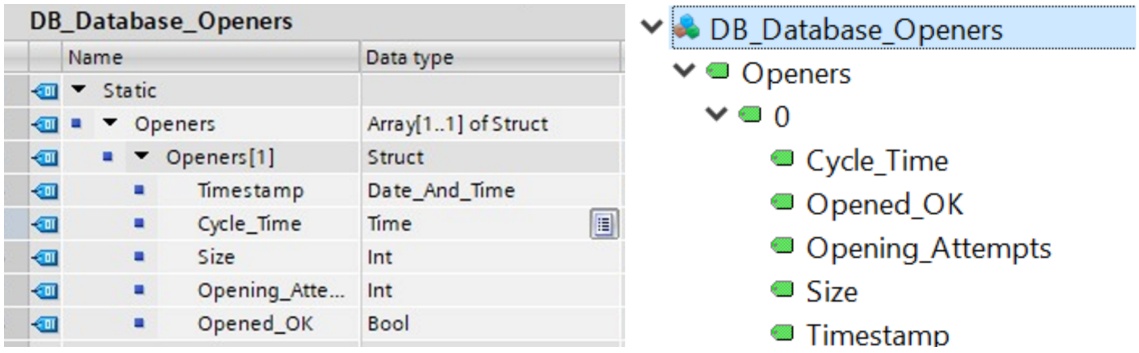
## 4.2 Druhé řešení

Nasazení bylo provedeno na vývojovém projektu firmy ACAM Solution. Dodávka se zabývá návrhem a konstrukcí strojů na vyskládání přepravek na pásový dopravník z palety (depaletizér) a otvírač přepravek. Přepravky následně míří po pásovém přepravníku do myčky. Stroje budou nasazeny v závodě s nepřetržitým provozem. Plánováný tok přepravek otvíračem je 1125 přepravek za hodinu.

Otvírač má při otevírání přepravky variabilní cyklus. Otevření tedy vždy nezabere shodný čas. Zároveň se otevření nemusí zdařit na první pokus, případně se nemusí zdařit vůbec. Standardní délky cyklu stroje je 3 s, kdy k otevření dojde na první pokus. Přepravky mohou mít různou výšku. Stroj má několik režimů běhu, základními je manuální a automatický. Depaletizér má také variabilní cyklus, standardní délka je 2,5 s. Hlava stroje pojíždí v ose $z$ a postupně shazuje jednotlivé přepravky. Otvírač a depaletizér je každý řízen samostatným PLC Siemens řady S7-1500.

### 4.2.1 Sbíraná data

Jako příklad sbíraných dat je vybrán datový blok Otvírače. Na obrázku 4.5 na levé straně je zvýrazněno zobrazení datového bloku v programovacím prostředí PLC, na pravé straně ekvivalentní zobrazení z OPC UA serveru pomocí programu UA Expert.




Obr. 4.5: Datový blok Otevírače v TIA Portálu a programu UA Expert

Datový blok se modifikuje po každém proběhnutém cyklu stroje. Význam hodnot je následující:

- Cycle_Time – čas proběhnutého cyklu v milisekundách,
- Timestamp – časová značka provedení,
- Size – velikost přepravky,
- Opening_Attemps – počet pokusů otevření,
- Opened_OK – indikace korektnosti výstupu.

Předpis pro data Otvírače v InfluxDB je znázorněn v tabulce 4.4. Data jsou obohacena o tag *Machine*, který říká, o jakou instanci stroje se jedná. Pro tento otvírač byl zvolen název *Opener_1* Obdobný tag Machine je v databázi využit i u dalších datových bloků.

| measurement | tag | field | field | field | field |
|---|---|---|---|---|---|
| Opener | Machine | Cycle_Time | Size | Opening_Attempts | Opened_OK |

Tab. 4.4: Předpis schématu pro zápis do InfluxDB

Dalšími sbíranými daty je datový blok depaletizéru, spotřeby elektrické energie, spotřeby stlačeného vzduchu.

### 4.2.2 Iot Hub

Kvůli datovému toku a počtu událostí během dne již nepostačuje úroveň F1 – Free IoT Hubu a je nutné využít placenou instanci z úrovně Standard, kategorie S1, jedna jednotka. Služba byla zřízena v datacentru North Europe. Úvodní konfigurace byla následující:

- nastavení konektivity na *Public acess*,
- cenová kategie na *S1: Standard*,
- služba *Defender for IoT* deaktivována,
- přístupová práva na *Shared acess policy + RBAC*.

### 4.2.3 Iot Edge

IoT Edge byl instalován na PC, které bylo součástí dodávky. Na PC běží operační systém Windows 10 Pro, je osazen procesorem architektury Intel X64. PC má k dispozice dvě síťová rozhraní. První směřuje do lokální sítě s průmyslovými zařízeními. Druhé do lokální sítě s přístupem k internetu. Instalace IoT Edge proběhla dle [55] s využitím symetrického klíče. Byla využita EFLOW verze 1.2, tedy aktuální vývojová větev. Virtualizace byla nastavena na jedno virtuální jádro procesoru, 1 GB paměti RAM a 10 GB diskového prostoru. Nebylo nastaveno sdílení grafické karty.

Pomocí modifikace Deployment Manifest v IoT Hubu bylo nastaveny runtime moduly edgeAgent a edgeHub na verzi 1.2. Runtime moduly se v této konfiguraci automaticky aktualizují po publikaci nové verze 1.2.x.

#### OPC Publisher

Pomocí modifikace Deployment Manifest v IoT Hubu byl na zařízení IoT Edge instalován modul OPC Publisher. Pro nasazení byla vybrána verze 2.8.2. Aktualizace

po publikaci nové verze musí proběhnout v tomto případě ručně. Mezi verzí 2.8.1 a 2.8.2 byly zavedeny změny, které mohly způsobit nefunkčnost některých řešení, jak se uvádí v [56]. Proto je v tomto případě vhodné provádět aktualizovat ručně.

Konfigurace byla nastavena na odesílání zpráv do IoT Hubu po naplnění maximální velikosti zprávy nebo po dosažení intervalu 200 s. Přístup k OPC UA serverům je povolen v režim bez autentizace a bez šifrování. Pro přístup k OPC UA serverům jsou v konfiguraci specifikovány jmenné názvy pro použité IP adresy na PLC, jmenné názvy jsou využívány v souboru *publishednodes.json*. Byla vypnuta vlastnost provozování lokálního serveru s přístupem k metrikám. Formát odesílaných zpráv je nastaven na výchozí. Nastavení konfigurace vychází z dokumentu [57]. Z něho je také patrné, že parametry konfigurace se postupným vývojem mění. Vývojový tým se snaží zachovat kontinuitu použitých parametrů, nicméně kombinace nových a dříve užívaných názvů parametrů je značně matoucí. Nasazená konfigurace používá pouze názvy parametrů z nejnovější verze.

Konfigurace spojení s OPC UA servery se provádí v souboru *publishednodes.json*, který je umístěn na disku virtuálního stroje. Pro zajištění přístupu modulu k souboru, je třeba v konfiguraci deklarovat umístění úložiště modulu a do tohoto úložiště soubor umístit. Jde o typickou vlastnost běhu programu v kontejneru, kdy program v kontejneru nemá přístup k celému disku, pouze k jeho vybrané části, která se nazývá úložiště modulu. Na hostujícím operačním systému byla vybrána složka */var/iiotedge*, která je pro modul viditelná pod cestou */mount*. Dále je třeba nakonfigurovat práva přístupu uživatele *iotedge*, pod kterým modul na hostujícím operačním systému běží, k tomuto umístění. Toho bylo dosaženo využitím příkazu *chown iotedge /var/iiotedge*.

V souboru *publishednodes.json* jsou specifikovány uzly OPC UA serverů na PLC, které jsou odebírány. Jsou zde specifikovány požadované vzorkovací intervaly, požadované intervaly odesílání pro OPC UA servery a požadovaný klíč *DisplayName*, podle kterého se hodnota parsuje v Azure Functions.

Kompletní konfigurace je součástí přílohy.

### 4.2.4 Azure Functions

Pro přijetí dat z IoT Hubu je třeba implementovat protokol Event Hub Endpoint. Pro implementaci v Azure jsou v dokumentaci doporučené tyto způsoby:

- Azure Functions,
- Azure Stream Analytics,
- Time Series Insights,
- a další [21].

Z těchto služeb by měla být pro nižší datové toky nejméně nákladná na provoz služba Azure Functions.

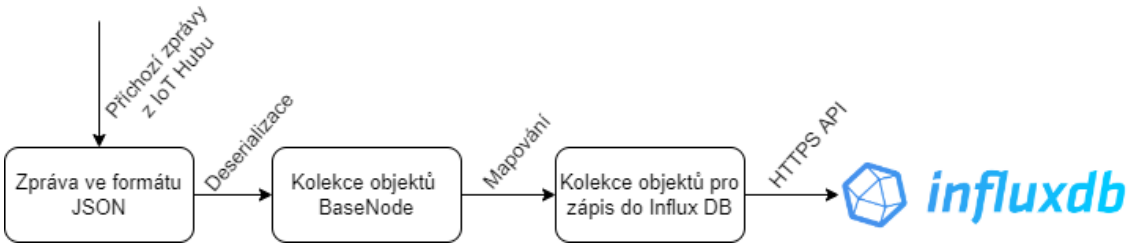
Obecně v této službě běží programy, které se spouští po detekci nějaké události. Pro tuto implementaci je zásadní reakce na dostupnost nové zprávy z protokolu Event Hub endpoint. Azure Functions zajišťuje volání funkce po přijetí nové zprávy, argumentem funkce je obsah zprávy. Pro vývoj byl vybrán programovací jazyk C# nad frameworkem .NET 6.0. vývoj probíhal v programovacím prostředí Visual Studio. Zprávy jsou doručeny ve formátu JSON od modulu OPC Publisher. Pro deserializaci byla vybrána knihovna System.Text.Json. Knihovna je součástí frameworku .NET, není tedy třeba pro deserializaci instalovat další knihovny.

Pro parsování se využívá hodnota *DisplayName*, ktera je konfigurována pro každý odebíraný uzel OPC UA serveru v souboru *publishednodes.json*. Komplikací pro deserializaci je polymorfní tvar formátu JSON. Příklad je uveden v příloze B.3. Zde je vidět, že objekt *Value* na stejné úrovni JSON má zcela jinou strukturu. To znemožňuje staticky přiřadit datový typ pro daný název uzlu zprávy. Místo toho je třeba rozhodnout o typu objektu pro deserializaci na základě hodnoty *DisplayName*. Pro každý nově přidaný typ je třeba přidat implementaci deserializace. V případě, že typ není rozpoznán, je deserializován do objektu *NotImplementedJsonNode*. Tento objekt je následně zapsán do databáze a tím je propagována informovace o přijetí nové neznámé hodnoty, kterou není možné deserializovat. Tím vznikne kolekce objektů BaseNode. Jedná se o datový ekvivalent původní zprávy, pouze ve formátu kolekce tříd. Následně jsou tyto objekty podle typu přemapovány do objektů pro zápis do databáze Influx. Při mapování je také použita logika překladu některých hodnot. Příkladem je časová značka, kterou je opatřen záznam z otvírače. Tato časová značka je vytvořena v PLC a odpovídá datovému typu *DATE_AND_TIME*, který je popsán v [1]. Časové značky ostatních veličin vzniknou při vzorkování OPC UA serverem. U diskretizace spojitých veličin jako je spotřeba elektřiny to není na obtíž. Při vzorkování událostí otvírače by časová hodnota nebyla přesná.

Pro zapsání do databáze InfluxDB je využita knihovna InfluxDb.Client, která je uvolněna pod licencí MIT a lze ji importovat pomocí nástroje NuGet.

Konfigurace přístupu k IoT Hubu a InfluxDB je uložena v soubor *local.settings.json*. Přístupové klíče tak nejsou součástí zdrojového kódu a jeho hodnoty jsou po nasazení v Azure zašifrovány. Ke kopírování těchto hodnot do nasazené varianty nedochází automaticky s nasazením, je třeba změnu konfigurace vynutit. To lze provést pomocí příkazu přes CLI nebo změnou v portálu Azure.

Funkce je vytvořena tak, aby nezachytávala neošetřitelné výjimky a v případě chyby raději ukončila svou činnost, než zapsala nekorektní hodnoty do databáze. Zaznamenávání výjimek je zajištěno službou Application Insights. Pomocí uchování zpráv po dobu retention period (typicky 1 den) v IoT Hubu lze zprávy, u kterých




Obr. 4.6: Diagram zpracování zprávy v Azure Functions

došlo k chybě, znovu uložit. Za tímto účelem lze spustit funkci lokálně a využít nového konzumenta v IoT Hubu. Nový konzument dodá všechny zprávy, které jsou uloženy v rámci retention period. Dále lze využít vlastnost InfluxDB, že měření se stejným obsahem nejsou ukládána znovu. Není tedy třeba vyhledávat v kolekci zpráv tu, u které k selhání došlo, ale postačí prosté spuštění funkce.

V Application Insights je dostupný kompletní Call stack zachycených výjimek. Lze tedy vyčíst na jakém řádku kódu došlo k chybě. Pokud chyba nastala při deserializaci zdrojové zprávy, pak lze předpokládat, že k chybě dojde při deserializaci stejné zprávy znovu. V lokální instanci lze tedy vypnout zápis do databáze a otestovat deserializaci na novém konzumentovi IoT Hubu.

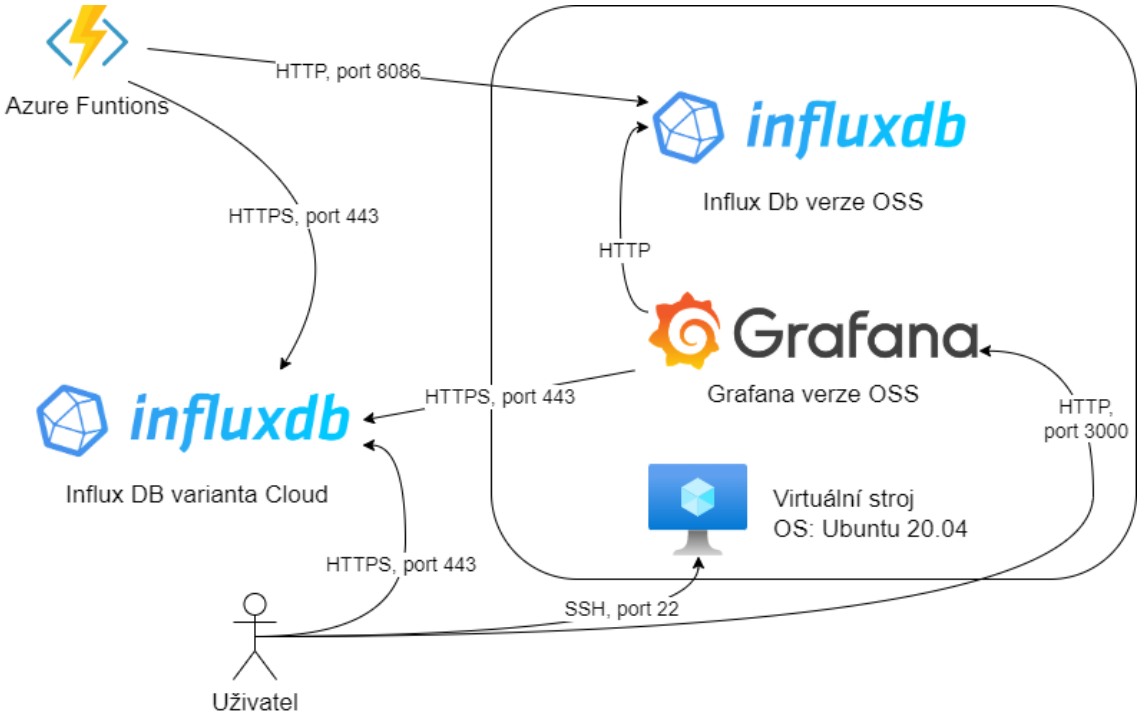
Z diagramu v následující části 4.7 je vidět, že se ukládají data do dvou databází InfluxDB současně. Kvůli tomu jsou nasazeny dvě instance obdobné verze funkce, které se liší pouze konfigurací. Pro každou instanci je přidělen jeden konzument IoT Hubu a přístupová práva k instanci databázi.

### 4.2.5 Databáze a vizualizace

InfluxDB je nasazena ve dvou instancích. První instance je použita v distribuci Cloud, tedy jako služba ve formě SaaS a hostována v Cloudu v datacentru West Europe. Druhá varianta je provozována na virtuálním stroji v Azure ve variantě OSS. Na virtuálním stroji je zároveň provozován vizualizační nástroj Grafana. Schéma nasazení s využitými porty je na obrázku 4.7.

Virtuální stroj byl nasazen s operačním systémem Ubuntu 20.04 LTS. Kategorie hostování byla zvolena na B1s, v této kategorii je jedno virtuální jádro procesoru, 1 GB RAM a 30 GB úložného prostoru na disku.

Instalace InfluxDB proběhla podle dle postupu uvedeného v [58]. Spuštění databáze po startu operačního systému bylo zajištěno pomocí příkazu *systemd*. Firewall v portálu Azure byl nastaven na povolení přístupu přes protokol TCP a port 8086 k webovému rozhraní InfluxDB.




Obr. 4.7: Diagram nasazení databáze a vizualizace, ikona převzata z [50]

Instalace Grafana proběhla podle [59]. Instalována byla verze OSS, byly respektovány pokyny pro konfiguraci spuštění po startu operačního systému. Přístup Grafany do obou instancí InfluxDB byl zajištěn vygenerováním přístupového tokenu k API s nastavením práva pro čtení.

Ve webovém prostředí obou vizualizačních nástrojů byly vytvořeny nástěnky, které vizualizují činnost stroje. Grafické podoby vizualizací jsou na obrázcích C.2 a C.1 v příloze. Konfigurační soubory pro replikaci vizualizací jsou v příloze D. V konfiguračních souborech jsou uvedeny databázové dotazy na InfluxDB.

Grafická podoba webového prostředí Influx se mírně liší mezi distribucemi. Zároveň se liší funkčnost. Například u distribuce OSS lze exportovat konfigurace nástěnky do formátu JSON. U distribuce Cloud tato varianta není. Konfigurační soubor lze do nástěnky v distribuci Cloud importovat a grafická podoba je totožná.

Grafana poskytuje více vizualizačních prvků (grafy, tabulky, indikátory), které lze přidat na nástěnku. Lze je také lépe nastavovat a upravovat. Podoba nástěnky je automaticky verzována se zachováním předešlých verzí.

### 4.2.6 Cena testovacího provozu

Provoz kompletního popsaného řešení probíhal od 10. 4. 2022 do 14. 5. 2022. Stabilní provoz průmyslového zařízení s reálnými datovými toky probíhal mezi 1. 5. a 14. 5.

Cenu jednotlivých komponent za toto období shrnuje tabulka 4.3.

| Služba | Druh | Cena [USD] |
|---|---|---|
| IoT Hub | Standrad S1, 1 jednotka | 10,48 |
| Virtuální stroj | B1 | 0,00* |
| Úložiště | Disk virtuálního stroje | 0,89 |
| Úložiště | Diskové operace virtuálního stroje | 0,61 |
| Functions | Execution Time | 0,02 |
| Úložiště | LRS Write Operations | 0,13 |
| Úložiště | Read Operations | 0,10 |
| | Celkem | 12,23 |

Tab. 4.5: Cena druhého řešení za 14 dní provozu

*Během provozu došlo k čerpání služeb, které jsou v Azure zdarma pro každého uživatele. To se dotklo hlavně virtuálního stroje, ten je zdarma 750 h užívání. Standardní provoz by stál 0,0113 USD za hodinu, za testovací období tedy 3,80 USD. Dále byl čerpán fond zdarma pro vykonávání Azure Functions [60].

Při provozu distribuce Cloud InfluxDB byly na metrikách za testovací období naměřeny hodnoty v tabulce 4.6. Služba byla používána v variantě zdarma. Výpočet ceny při přechodu na platbu za spotřebované jednotky ukazuje tabulka 4.6. Podstatnou část ceny tvoří provedené dotazy, tuto jednotku by bylo možné snížit vypnutím automatické aktualizace nástěnky vizualizace.

| Metrika | Jednotka | Spotřeba | Cena [USD] |
|---|---|---|---|
| Přijatá data | [MB] | 80,72 | 0,161 |
| Provedené dotazy | [100 provedení] | 161,10 | 1,611 |
| Použité úložiště | [GB×hodina] | 8,16 | 0,016 |
| Odchozí data | [GB] | 0,03 | 0,003 |
| | Celkem | | 1,79 |

Tab. 4.6: Cena InfluxDB distribuce Cloud, jednotková cena z 3.2

### 4.2.7 Rozšiřitelnost architektury pro další zařízení

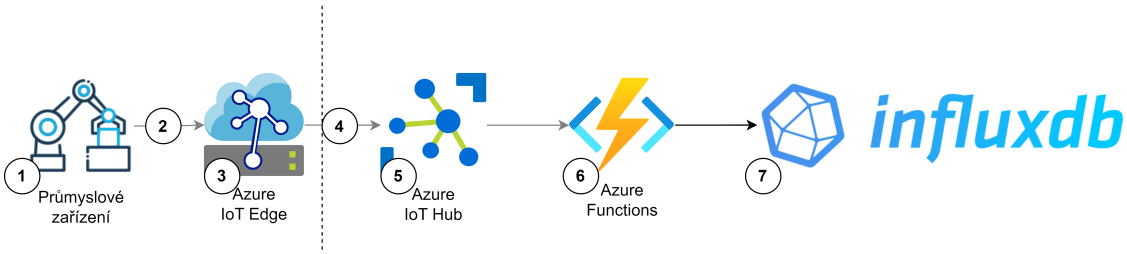
Popsané způsoby nasazení obsahovaly vždy jednu nasazenou instanci zařízení IoT Edge připojenou k IoT Hubu. Při rozšíření na několik zařízení je možné začít využívat předpis Deployment Manifest pro skupinu zařízení. Lze tak systematicky např. aktualizovat skupinu zařízení.

Při nasazení více zařízení může začít docházet potřebná kapacita přenesených zpráv v rámci IoT Hubu. V případě druhého řešení se počet vyčerpaných zpráv za den nepřesahoval 3 500. Z celkového dostupného množství zpráv 400 000 za den v kategorii S1 jde o necelé procento vytížení. Při přiblížení se hranici maximálního počtu zpráv lze opět zvážit, zda nenasadit modul filtrování zpráv přímo na zařízení IoT Edge.

V případě, že by finanční prostředky pro hostování kategorie S1 IoT Hubu byly vyhodnoceny za výrazný náklad, pak lze IoT Hub použít v kategorii Free a využívat jej pouze pro konfiguraci a monitorování připojených zařízení. Data do InfluxDB lze nahrát nasazením obdobného software jako běží v Azure Functions na zařízení IoT Edge. Toto nasazení je podporováno, je zdarma a bylo by třeba provádět minimální změny v kódu a data by byla ukládána přímo do databáze. V tomto případě bude porušena vlastnost, že ze zařízení se komunikuje přes jediný protokol aplikační vrstvy a port protokolu TCP. Zároveň by byla narušena vlastnost lokálního ukládání zpráv při výpadku síťového připojení. Toto by bylo možné nahradit nasazením kontejneru databáze Influx přímo na zařízení IoT Edge a nastavením replikace této databáze do Cloudové instance InfluxDB. Toto nastavení databáze InfluxDB je podporováno od nejnovější verze OSS 2.2 a řeší výpadky připojení.

### 4.2.8 Analýza selhání komponent

Cílem této části je diskutovat možné problémy a důsledky selhání jednotlivých komponent systému, možnosti obnovení provozu a zotavení. Na obrázku 4.8 jsou číslem označeny body, kde bude selhání diskutováno.




Obr. 4.8: Blokové schéma architektury s naznačenými uvažovanými body výpadku

#### 1 – Výpadek průmyslového zařízení nebo OPC UA serveru

Při výpadku nebo vypnutí PLC není možné stroj spustit a sběr dat nemá význam. Modul OPC Publisher se s nastavenou výchozí periodou 10 s pokouší spojení obnovit. Při odmítnutí připojení OPC UA serveru na PLC, které může být způsobené

překročením maximálního počtu připojených klientů nebo chybou autentizace, lze tuto informaci odchytit z logu modulu OPC Publisher. Ve výpisu se objeví informace, že OPC UA server aktivně odmítá připojení a důvod. Logy modulů lze odchytávat přes IoT Hub.

#### 2 – Výpadek připojení k síti průmyslového zařízení

Při výpadku připojení k síti průmyslového zařízení, která může být způsobena výpadkem síťového kabelu nebo síťového prvku, nelze rozlišit mezi chybou nedostupnosti OPC UA serveru popsanou v části 1. Pro přesnou identifikaci chyby je nutné se k počítači připojit přes VPN nebo lokálně a detekovat stav připojení na lokálním síťovém rozhraní přímo z prostředí Windows.

#### 3 – Výpadek IoT Edge

Při výpadku hardwaru PC, operačního systému Windows, chyb spuštění virtualizovaného operačního systému, případně runtime modulů IoT Edge lze tyto potíže odstranit buď přímo lokálně případně některé přes VPN. Výpadky runtime modulů jsou sice vzácné, nicméně byly zaznamenány při testování provozu na Raspberry Pi. Chyba se projevovala po restartu zařízení při spuštění runtime modulu edgeAgent chybovou hláškou *permission denied* při pokus o přístup ke komunikaci modulů. Chybu bylo možné odstranit dowgradem verze IoT Edge nebo ručním přidělením potřebných práv. K odstranění chyby ve zdrojovém kódu došlo po devíti dnech od nahlášení. Po odstranění ve zdrojovém kódu se čekalo na uvolnění nové verze [61].

Při výpadku modulu OPC Publisher či potřeby detekce jeho chyb je možné zobrazit logy modulů edgeAgent, který moduly instaluje a spouští. V případě chyb modulu lze zobrazit jeho logy z IoT Hubu a modul restartovat, upravit jeho konfiguraci.

V případě nutnosti vyšší spolehlivosti sběru dat z konkrétního zařízení lze zvážit obdobnou redundantní architekturu, která je popsána 2.4.3. Alternativou je prostá redundance zařízení s IoT Edge se shodnou konfigurací, které se budou ukládat do samostatných bucket v databázi Influx. V případě výpadku lze využít data z druhého zařízení. Nevýhodou této architektury je vyšší zatížení OPC UA serveru. Získaná data nebudou totožná, protože každý modul OPC Publisher založí v OPC UA serveru vlastní spojení.

#### 4 – Výpadek spojení s IoT Hubem

Při výpadku spojení s IoT Hubem jsou zprávy uchovány po nastavenou dobu TTL pro každý modul v IoT Edge. výchozí hodnota je 7 200 vteřin a celková doba v sekundách je omezena maximální velikostí datového typu integer. Data se mohou uchová-

vat v souborovém systému na hostujícím operačním systému místo vnitřního úložiště runtime modulu kontejneru. Je tedy nutné udržet kapacitu diskového prostoru tak, aby nemohlo dojít k jeho zaplnění telemetrickými daty v případě výpadku. Zprávy v tomto úložišti jsou uchovány i v případě restartu zařízení s IoT Edge [62]. Po obnovení spojení jsou všechny zprávy odeslány do IoT Hubu.

#### 5 – Výpadek IoT Hubu

Výpadek IoT Hubu je poměrně nepravděpodobný vzhledem ke způsobu jeho nasazení v clusteru. Microsoft poskytuje pro placenou variantu S1 smlouvu typu SLA, dostupnost by měla být 99,9 % času v účetním měsíci. V případě výpadku celého domácího datacentra North Europe se IoT Hub automaticky replikuje do datacentra West Europe. Proces replikace je možné otestovat a spustit z portálu Azure. Vpřípadě replikace může dojít ke ztrátě neodebraných zpráv z Event Hub endpoints a zpráv, které jsou v období retention period IoT Hubu. Zároveň při replikaci není záruka, že connection stringy k Event Hub endpoint nebudou změněny. Při využití čtení pomocí Azure Functions je možné, že bude třeba službu restartovat [63].

#### 6 – Výpadek Azure Functions

Diskuze selhání kódu a obnovení dat je uvedena v kapitole 4.2.4. Zkráceně lze konstatovat, že pokud dojde k zotavení z výpadku, ať už na lokálním stroji do doby retention period, pak by nemělo dojít ke ztrátě dat.

#### 7 – Výpadek InfluxDB

V architektuře druhého řešení jsou databáze ve variantě Cloud a hostovaná na virtuálním stroji redundantní. V obou jsou tedy shodná data. Databáze jsou provozovány ve 2 odlišných datacentrech – North Europe a West Europe. V případě výpadku kratšího než retention period IoT Hubu lze dohrát data s pomocí Azure Functions. V případě kompletní ztráty dat v jedné databázi jsou k dispozici data v té druhé. V případě ztráty určitého časového úseku lze data za tento časový úsek překopírovat a dostat databáze opět do konzistentního stavu.

# Závěr

V rámci diplomové práce byl vyvinut systém sběru dat z průmyslových zařízení. Při návrhu byl kladen důraz na možnost integrace do stávajících projektů firmy ACAM Solution. Při výběru technologií pro sběr dat byly vybírány přednostně zavedené open source projekty, za kterými stojí a zajišťuje podporu komerční společnost. Tento přístup by mohl zajistit rozumnou životnost navrženého systému.

První část práce se zabývala rešerší a výběrem vhodného síťového protokolu pro sběr dat z PLC s ohledem na dostupnost a implementaci protokolu v PLC výrobce Siemens. Pro implementaci byl zvolen protokol OPC UA v režimu komunikace klient-server. Hlavními přednostmi jsou autentizace klienta, podpora šifrování, možnost procházet datové bloky PLC za běhu programu a možnost konfigurovat přístup k datovým blokům.

Systém byl navržen s ohledem na dostupné služby ve veřejném cloudu Microsoft Azure. Průzkumem těchto technologií se zabývá druhá část práce. Pro zabezpečené připojení modulů sběru dat a distribuci telemetrie k dalšímu softwaru je vybrána služba IoT Hub. Pro nasazení modulu sběru dat do lokální sítě k průmyslovým zařízením software Iot Edge. IoT Edge umožňuje podle konfigurace v IoT Hubu lokálně nasadit kontejnery Docker. Běh kontejnerů lze monitorovat přes IoT Hub. IoT Edge zajišťuje transport telemetrie do IoT Hubu a lokální ukládání telemetrie pro případ výpadku spojení s IoT Hubem. Provoz IoT Edge byl otestován na PC a Raspberry Pi.

Pro sběr dat z OPC UA serveru je vybrán Docker kontejner OPC Publisher. Modul umožňuje přístup k OPC UA serverům v režimu klient. Podporuje autentizaci, autorizaci, šifrování a vzdálenou konfiguraci sbíraných datových bloků z PLC přes IoT Hub. O nasazení kontejneru bez IoT Edge lze uvažovat i na vybrané panely HMI výrobce Siemens řady Unified.

Pro hostování programu pro zpracování telemetrických dat a uložení do databáze byla zvolena služba Azure Functions. Pro ukládání telemetrických dat byla vybrána Time Series databáze Influx DB. Vizualizaci dat z databáze umožňuje webová aplikace Grafana.

Navržený systém byl nasazen na vývojovém projektu firmy Acam Solution. K projektu byla vytvořena vizualizace dat. Byla zaznamenána cena za testovací nasazení.

Cloud umožňuje zbavit se potřeby využití vlastního hardware a správy operačních systémů. Návrh systému v Cloudu je pak částečně podřízen hledání vhodných a kompatibilních služeb. Výhodou je vyšší spolehlivost nasazeného řešení kvůli clusterování serverů a přesouváním mezi datovými centry v případě výpadku. Nevýhodou zvoleného nasazení v Cloudu je závislost na konkrétním poskytovateli služeb a jeho cenové politice. Databáze a vizualizace lze migrovat k jinému poskytovateli

veřejného Cloudu. Služba IoT Hub je svázána pouze s poskytovatelem Microsoft Azure. Jako největší komplikaci Cloudových služeb vnímám nepředvídatelnost ceny. Pro zjištění reálných nákladů na provoz je třeba provést testování služby a ani to nemusí být pro konečnou cenu v provozu zcela směrodatné, protože do výpočtu konečné ceny jsou zahrnuty různé slevy.

# Literatura


[1] SIEMENS. *Simatic S7-1200 Programmable Controller: System Manual*. V4.5. NÜRNBERG, 05/2021n. l., 864 s. Dostupné také z: https://support.industry.siemens.com/cs/document/109797241?lc=en-AR

[2] SIEMENS. *Simatic S7-1500, ET 200MP Automation system: System Manual*. NÜRNBERG, 05/2021n. l., 1418 s. Dostupné také z: https://support.industry.siemens.com/cs/document/59191792/simatic-s7-1500-et-200mp-automation-system?dti=0&lc=en-GB

[3] SIEMENS. *Libraries for Communication for SIMATIC Controllers: SIMATIC Controllers, TIA Portal*. V1.1.1. NÜRNBERG, 09/2021n. l., 1418 s. Dostupné také z: https://support.industry.siemens.com/cs/document/109780503/

[4] VÁVRA, Petr. *Systém řízení rekreačního objektu* [online]. Brno, 2020 [cit. 2022-01-01]. Dostupné z: https://www.vutbr.cz/studenti/zav-prace/detail/124282. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky. Vedoucí práce Petr Fiedler.

[5] MQTT Essentials. *HiveMQ* [online]. Landshut: HiveMQ, c2022 [cit. 2022-04-11]. Dostupné z: https://www.hivemq.com/mqtt-essentials/

[6] *MQTT* [online]. mqtt.org, c2022 [cit. 2022-04-10]. Dostupné z: https://mqtt.org/

[7] Comparison of MQTT implementations. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2022-04-10]. Dostupné z: https://en.wikipedia.org/wiki/Comparison_of_MQTT_implementations

[8] S7NetPlus/s7netplus. *Github* [online]. San Francisco: Github, c2021, 10 Jun 2021 [cit. 2021-12-12]. Dostupné z: https://github.com/S7NetPlus/s7netplus

[9] POUND, Michael. TLS Handshake Explained - Computerphile. In: *Youtube* [online]. Mountain View (CA): Google [cit. 2022-04-11]. Dostupné z: https://www.youtube.com/watch?v=86cQJ0MMses

[10] POUND, Michael. Transport Layer Security (TLS) - Computerphile. In: *Youtube* [online]. Mountain View (CA): Google [cit. 2022-04-11]. Dostupné z: https://www.youtube.com/watch?v=0TLDTodL7Lc

[11] Transport Layer Security. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2022-04-11]. Dostupné z: https://en.wikipedia.org/wiki/Transport_Layer_Security

[12] STEINKRAUS, Uwe. OPC UA Introduction: Overview - Technology - Security - Getting Started. In: *Youtube* [online]. Mountain View (CA): Google [cit. 2022-04-11]. Dostupné z: https://www.youtube.com/playlist?list=PLROM1mLWekVCzxGMs6kzpBnJiDh3fiquD

[13] *ČSN CLC IEC/TR 62541-1: Sjednocená architektura OPC - Část 1: Přehled a koncepty*. 01.03.2022. Praha: Úřad pro technickou normalizaci, metrologii a státní zkušebnictví, 2022.

[14] KIMBERLY, Mlitz. Cloud infrastructure services vendor market share worldwide from 4th quarter 2017 to 3rd quarter 2021. *Statista* [online]. New York: Statista, 2021, Oct 29, 2021 [cit. 2021-12-05]. Dostupné z: https://www.statista.com/statistics/967365/worldwide-cloud-infrastructure-services-market-share-vendor/

[15] ECLIPSE FOUNDATION. 2020 IoT Developer Survey Key Findings. *Eclipse iot* [online]. Ottawa: Eclipse Foundation, 2020 [cit. 2021-12-12]. Dostupné z: https://iot.eclipse.org/community/resources/iot-surveys/assets/iot-developer-survey-2020.pdf

[16] MICROSOFT. Frequently asked questions about the Education Hub. MICROSOFT. *Microsoft technical documentation* [online]. Redmond: Microsoft, 2021 [cit. 2021-12-12]. Dostupné z: https://docs.microsoft.com/en-us/azure/education-hub/azure-dev-tools-teaching/program-faq#azure-for-students

[17] MICROSOFT. Azure pricing and billing FAQ. MICROSOFT. *Microsoft Azure* [online]. Redmond: Microsoft, 2021 [cit. 2021-12-12]. Dostupné z: https://azure.microsoft.com/en-gb/pricing/faq/

[18] MICROSOFT. Choose a device communication protocol. MICROSOFT. *Microsoft technical documentation* [online]. Redmond: Microsoft, 2021, 11/16/2021 [cit. 2021-12-12]. Dostupné z: https://docs.microsoft.com/en-us/azure/architecture/reference-architectures/iot

[19] HOFÍREK, Radek, V. ŠEVČÍK a T. SKOČIL. SIEMENS. CloudConnect, Edge Computing, Edge Aplikace. SIEMENS. *TIA na dosah* [online]. Praha: Siemens, c1996-2021, 08/2020 [cit. 2021-12-19]. Dostupné z: https://www.tianadosah.cz/upload/350-1709367469.pdf

[20] MICROSOFT. Develop without using an Azure IoT Hub SDK. MICROSOFT. *Microsoft technical documentation* [online]. Redmond: Microsoft, 2021, 11/16/2021 [cit. 2021-12-12]. Dostupné z: https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-devguide-no-sdk

[21] MICROSOFT. Reference - IoT Hub endpoints. MICROSOFT. *Microsoft technical documentation* [online]. Redmond: Microsoft, 2021, 11/16/2021 [cit. 2021-12-12]. Dostupné z: https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-devguide-endpoints

[22] MICROSOFT. Ceny za Azure IoT Hub. MICROSOFT. *Microsoft Azure* [online]. Redmond: Microsoft, 2021 [cit. 2021-12-12]. Dostupné z: https://azure.microsoft.com/cs-cz/pricing/details/iot-hub/#pricing

[23] MICROSOFT. Reference - IoT Hub quotas and throttling. MICROSOFT. *Microsoft technical documentation* [online]. Redmond: Microsoft, 2021, 11/16/2021 [cit. 2021-12-12]. Dostupné z: https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-devguide-quotas-throttling

[24] MICROSOFT. What is Azure IoT Edge. MICROSOFT. *Microsoft technical documentation* [online]. Redmond: Microsoft, 2021, 09/29/2021 [cit. 2021-12-18]. Dostupné z: https://docs.microsoft.com/en-us/azure/iot-edge/about-iot-edge?view=iotedge-2020-11

[25] MICROSOFT. Azure IoT Edge supported systems. MICROSOFT. *Microsoft technical documentation* [online]. Redmond: Microsoft, 2022, 05/10/2022 [cit. 2022-05-12]. Dostupné z: https://docs.microsoft.com/en-us/azure/iot-edge/support

[26] MICROSOFT. Tutorial: Develop a C# IoT Edge module using Linux containers. MICROSOFT. *Microsoft technical documentation* [online]. Redmond: Microsoft, 2021, 08/12/2021 [cit. 2021-12-14]. Dostupné z: https://docs.microsoft.com/en-us/azure/iot-edge/tutorial-csharp-module

[27] MICROSOFT. Ceny za Container Registry. MICROSOFT. *Microsoft Azure* [online]. Redmond: Microsoft, 2021 [cit. 2021-12-12]. Dostupné z: https://azure.microsoft.com/cs-cz/pricing/details/container-registry/

[28] Azure/Industrial-IoT. *Github* [online]. San Francisco: Github, c2022 [cit. 2020-04-22]. Dostupné z: https://github.com/Azure/Industrial-IoT/tree/release/2.8.2

[29] Azure/Industrial-IoT: Industrial-IoT/docs/manual/readme.md. *Github* [online]. San Francisco: Github, c2022 [cit. 2020-04-22]. Dostupné z: https://github.com/Azure/Industrial-IoT/blob/release/2.8.2/docs/manual/readme.md

[30] Azure/Industrial-IoT: Industrial-IoT/docs/modules/publisher.md. *Github* [online]. San Francisco: Github, c2022 [cit. 2020-04-22]. Dostupné z: https://github.com/Azure/Industrial-IoT/blob/release/2.8.2/docs/modules/publisher.md

[31] Azure/Industrial-IoT: Industrial-IoT/docs/modules/publisher-directmethods.md. *Github* [online]. San Francisco: Github, c2022 [cit. 2020-04-25]. Dostupné z: https://github.com/Azure/Industrial-IoT/blob/2.8.2/docs/modules/publisher-directmethods.md

[32] Azure / Industrial-IoT: Industrial-IoT/docs/modules/publisher.md. *Github* [online]. San Francisco: Github, c2021, 16 Dec 2021 [cit. 2021-12-22]. Dostupné z: https://github.com/Azure/Industrial-IoT/blob/main/docs/modules/publisher.md

[33] Azure/Industrial-IoT: Industrial-IoT/docs/deploy/howto-deploy-all-in-one.md. *Github* [online]. San Francisco: Github, c2022 [cit. 2020-04-25]. Dostupné z: https://github.com/Azure/Industrial-IoT/blob/release/2.8.2/docs/deploy/howto-deploy-all-in-one.md

[34] Azure-Samples / iot-edge-opc-plc: README.md. *Github* [online]. San Francisco: Github, c2022 [cit. 2020-04-25]. Dostupné z: https://github.com/Azure-Samples/iot-edge-opc-plc/tree/2.2.0

[35] Azure/Industrial-IoT: Industrial-IoT/docs/deploy/howto-install-iot-edge.html. *Github* [online]. San Francisco: Github, c2022 [cit. 2020-04-25]. Dostupné z: https://github.com/Azure/Industrial-IoT/blob/2.8.2/docs/deploy/howto-install-iot-edge.md

[36] MICROSOFT. What is Azure Time Series Insights Gen2. MICROSOFT. *Microsoft technical documentation* [online]. Redmond: Microsoft, 2021, 04/22/2021 [cit. 2021-12-18]. Dostupné z: https://docs.microsoft.com/en-us/azure/time-series-insights/overview-what-is-tsi

[37] MICROSOFT. Ceny za Azure Time Series Insights. MICROSOFT. *Microsoft Azure* [online]. Redmond: Microsoft, 2021 [cit. 2021-12-19]. Dostupné z: https://azure.microsoft.com/cs-cz/pricing/details/time-series-insights/

[38] MICROSOFT. Edge Managed Certification Requirements. MICROSOFT. *Microsoft technical documentation* [online]. Redmond: Microsoft, 2021, 12/08/2021 [cit. 2021-12-29]. Dostupné z: https://docs.microsoft.com/en-gb/azure/certification/program-requirements-edge-managed

[39] UPTON, Eben. Supply chain, shortages, and our first-ever price increase. *Raspberry Pi* [online]. Cambridge: Raspberry Pi Foundation, c2021, 20th Oct 2021 [cit. 2021-12-29]. Dostupné z: https://www.raspberrypi.com/news/supply-chain-shortages-and-our-first-ever-price-increase/

[40] Raspberry Pi. *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2021-12-29]. Dostupné z: https://en.wikipedia.org/wiki/Raspberry_Pi

[41] Azure-Samples/Custom-vision-service-iot-edge-raspberry-pi. *Github* [online]. San Francisco: Github, c2021, 22 Sep 2021 [cit. 2021-12-18]. Dostupné z: https://github.com/Azure-Samples/custom-vision-service-iot-edge-raspberry-pi

[42] Raspberry Pi Model B. *RPishop.cz* [online]. Roudné: RPishop, c2021 [cit. 2021-12-29]. Dostupné z: https://rpishop.cz/801-model-b

[43] MICROSOFT. What is Azure IoT Edge for Linux on Windows. MICROSOFT. *Microsoft technical documentation* [online]. Redmond: Microsoft, 2022, 04/20/2022 [cit. 2022-05-07]. Dostupné z: https://docs.microsoft.com/en-us/azure/iot-edge/iot-edge-for-linux-on-windows?view=iotedge-2020-11

[44] DB-Engines Ranking - Trend of Time Series DBMS Popularity. *DB-Engines* [online]. Heilbronn: solid IT, c2022 [cit. 2022-04-28]. Dostupné z: https://db-engines.com/en/ranking_trend

[45] NAQVI, Syeda Noor Zehra, Sofia YFANTIDOU a Esteban ZIMÁNYI. *Time series databases and influxdb*. Université Libre de Bruxelles, December 17, 2017, 12 s. Dostupné také z: https://cs.ulb.ac.be/public/_media/teaching/influxdb_2017.pdf

[46] HAO, Yuanzhe, Xiongpai QIN, Yueguo CHEN, Yaru LI, Xiaoguang SUN, Yu TAO, Xiao ZHANG a Xiaoyong DU. TS-Benchmark: A Benchmark for Time Series Databases. *2021 IEEE 37th International Conference on Data Engineering (ICDE)* [online]. IEEE, 2021, 2021, 588-599 [cit. 2022-04-28]. ISBN 978-1-7281-9184-3. Dostupné z: doi:10.1109/ICDE51399.2021.00057

[47] InfluxDB schema design. *InfluxData Documentation* [online]. San Francisco: InfluxData, c2022 [cit. 2022-05-12]. Dostupné z: https://docs.influxdata.com/influxdb/v2.2/write-data/best-practices/schema-design/

[48] InfluxDB Cloud Pricing. *InfluxDB* [online]. San Francisco: influxdata, c2022 [cit. 2022-05-07]. Dostupné z: https://www.influxdata.com/influxdb-pricing/

[49] Influxdata / influxdb-client-csharp. *Github* [online]. San Francisco: Github, c2021, 19 Apr 2022 [cit. 2022-05-07]. Dostupné z: https://github.com/influxdata/influxdb-client-csharp/tree/v4.1.0

[50] *Grafana Documentation* [online]. Grafana Labs, c2022 [cit. 2022-05-14]. Dostupné z: https://grafana.com/docs/grafana/latest/

[51] Getting Started. *Raspberry Pi* [online]. Cambridge: Raspberry Pi Foundation, c2021 [cit. 2021-10-30]. Dostupné z: https://www.raspberrypi.org/documentation/computers/getting-started.html#setting-up-your-raspberry-pi

[52] MICROSOFT. Create and provision an IoT Edge device on Linux using symmetric keys. MICROSOFT. *Microsoft technical documentation* [online]. Redmond: Microsoft, 2021, 11/24/2021 [cit. 2021-12-18]. Dostupné z: https://docs.microsoft.com/en-us/azure/iot-edge/how-to-provision-single-device-linux-symmetric

[53] Azure / iot-edge-modbus. *Github* [online]. San Francisco: Github, c2021, 17 May 2019 [cit. 2021-12-18]. Dostupné z: https://github.com/Azure/iot-edge-modbus

[54] MICROSOFT. Tutorial: Develop IoT Edge modules with Linux containers. MICROSOFT. *Microsoft technical documentation* [online]. Redmond: Microsoft, 2021, 10/02/2021 [cit. 2021-12-22]. Dostupné z: https://docs.microsoft.com/en-us/azure/iot-edge/tutorial-develop-for-linux?view=iotedge-2020-11

[55] MICROSOFT. Create and provision an IoT Edge for Linux on Windows device using symmetric keys. MICROSOFT. *Microsoft technical documentation* [online]. Redmond: Microsoft, 2022, 03/08/2022 [cit. 2022-05-09]. Dostupné z: https://docs.microsoft.com/en-us/azure/iot-edge/how-to-provision-single-device-linux-on-windows-symmetric?view=iotedge-2020-11&tabs=azure-portal%2Cpowershell

[56] Breaking Change in OPC Publisher 2.8.2 in way how NodeId is reported back. *Github* [online]. San Francisco: Github, c2022, 25 Mar 2022 [cit. 2020-04-22]. Dostupné z: https://github.com/Azure/Industrial-IoT/issues/1655

[57] Azure/Industrial-IoT: Industrial-IoT/docs/modules/publisher-commandline.md. *Github* [online]. San Francisco: Github, c2022 [cit. 2020-04-22]. Dostupné z: https://github.com/Azure/Industrial-IoT/blob/release/2.8.2/docs/modules/publisher-commandline.md

[58] Install InfluxDB: Linux. *InfluxData Documentation* [online]. San Francisco: InfluxData, c2022 [cit. 2022-05-11]. Dostupné z: https://docs.influxdata.com/influxdb/v2.2/install/?t=Linux

[59] Install on Debian or Ubuntu. *Grafana Documentation* [online]. Grafana Labs, c2022 [cit. 2022-04-05]. Dostupné z: https://grafana.com/docs/grafana/latest/installation/debian/

[60] MICROSOFT. Explore free Azure services. MICROSOFT. *Microsoft Azure* [online]. Redmond: Microsoft, 2021 [cit. 2022-05-02]. Dostupné z: https://azure.microsoft.com/en-us/pricing/free-services/

[61] EdgeAgent doesn't start after reboot - permission denied. *Github* [online]. San Francisco: Github, c2022 [cit. 2022-05-11]. Dostupné z: https://github.com/Azure/iotedge/issues/5672

[62] MICROSOFT. Understand extended offline capabilities for IoT Edge devices, modules, and child devices. MICROSOFT. *Microsoft technical documentation* [online]. Redmond: Microsoft, 2021, 03/24/2022 [cit. 2022-05-11]. Dostupné z: https://docs.microsoft.com/en-us/azure/iot-edge/offline-capabilities?view=iotedge-2020-11

[63] MICROSOFT. IoT Hub high availability and disaster recovery. MICROSOFT. *Microsoft technical documentation* [online]. Redmond: Microsoft, 2021, 04/16/2022 [cit. 2022-05-11]. Dostupné z: https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-ha-dr

# Seznam symbolů a zkratek

| | |
|---|---|
| **ISO** | Mezinárodní organizace pro normalizaci |
| **ČSN** | česká technická norma |
| **OASIS** | Organization for the Advancement of Structured Information Standards |
| **PaaS** | Platform as a Service, platforma jako služba |
| **SaaS** | Software as a Service, software jako služba |
| **HMI** | Human machine interface |
| **PC** | osobní počítač |
| **CPU** | central processing unit, centrální procesorová jednotka |
| **PLC** | programmable logic controller, programovatelný logický automat |
| **HMI** | human machine interface, rozhraní mezi člověkem a strojem |
| **SCADA** | Supervisory Control And Data Acquisition |
| **ERP** | Enterprise Resource Planning, plánování podnikových zdrojů |
| **MES** | Manufacturing Execution Systems, výrobní informační systémy |
| **SCADA** | Supervisory Control And Data Acquisition |
| **API** | Application Programming Interface |
| **IP** | Internet Protocol |
| **TCP** | Transmission Control Protocol |
| **UDP** | User Datagram Protocol |
| **DNS** | Domain Name System |
| **TLS** | Transport Layer Security |
| **MQTTS** | MQ Telemetry Transport Secured |
| **AMQP** | Advanced Message Queuing Protocol |
| **HTTP** | Hypertext Transfer Protocol |

| | |
|---|---|
| **HTTPS** | Hypertext Transfer Protocol Secured |
| **SMTP** | Simple Mail Transfer Protocol |
| **FTP** | File Transfer Protocol |
| **OPC UA** | OPC Unified Architecture |
| **QoS** | Quality-of-service |
| **GPL** | GNU General Public License |
| **MIT** | Massachusetts Institute of Technology |
| **AGPLv3** | Affero General Public License version 3 |
| **JSON** | JavaScript Object Notation |
| **XML** | Extensible Markup Language |
| **CLI** | Command Line Interface, příkazová řádka |
| **SDK** | Software development kit |
| **LTS** | Long-term support, verze s dlouhodobou podporou |
| **TTL** | Time to live |
| **DB** | databáze |
| **HDD** | Hard Disk Drive, Pevný disk |
| **IoT** | Internet of Things |
| **USD** | americký dolar |
| **EUR** | euro |
| **UTC** | Coordinated Universal Time, Koordinovaný světový čas |
| **VUT** | Vysoké učení technické v Brně |
| **VPN** | Virtual Private Network |

# Seznam příloh

# A Deployment Manifest IoT Edge

Výpis A.1: Deployment Manifest pro zařízení IoT Edge v prvním řešení

```
{
    "modulesContent": {
        "$edgeAgent": {
            "properties.desired": {
                "modules": {
                    "SampleFilterModule": {
                        "settings": {
                            "image": "XYZ.azurecr.io/
                                samplefiltermodule:0.0.6-arm32v7",
                            "createOptions": "{}"
                        },
                        "type": "docker",
                        "version": "1.0",
                        "status": "running",
                        "restartPolicy": "always"
                    },
                    "Modbus": {
                        "settings": {
                            "image": "mcr.microsoft.com/
                                azureiotedge/modbus:1.0",
                            "createOptions": ""
                        },
                        "type": "docker",
                        "version": "1.0",
                        "status": "running",
                        "restartPolicy": "always"
                    }
                },
                "runtime": {
                    "settings": {
                        "minDockerVersion": "v1.25",
                        "registryCredentials": {
                            "pvregistry": {
                                "address": "XYZ.azurecr.io",
                                "password": "",
                                "username": ""
                            }
                        }
                    },
                    "type": "docker"
                },
                "schemaVersion": "1.1",
                "systemModules": {
```

```
                    "edgeAgent": {
                        "settings": {
                            "image": "mcr.microsoft.com/
                               azureiotedge-agent:1.2",
                            "createOptions": "{}"
                        },
                        "type": "docker"
                    },
                    "edgeHub": {
                        "settings": {
                            "image": "mcr.microsoft.com/
                               azureiotedge-hub:1.2",
                            "createOptions": "{\"HostConfig\":{\"
                               PortBindings\":{\"5671/tcp\":[{\"
                               HostPort\":\"5671\"}],\"8883/tcp\":[
                               {\"HostPort\":\"8883\"}],\"443/tcp\"
                               :[{\"HostPort\":\"443\"}]}}}"
                        },
                        "type": "docker",
                        "status": "running",
                        "restartPolicy": "always"
                    }
                }
            }
        },
        "$edgeHub": {
            "properties.desired": {
                "routes": {
                    "SampleFilterModuleToIoTHub": "FROM /messages/
                       modules/SampleFilterModule/outputs/* INTO
                       $upstream",
                    "ModbusFilterModule": "FROM /messages/modules/
                       Modbus/outputs/modbusOutput INTO
                       BrokeredEndpoint(\"/modules/
                       SampleFilterModule/inputs/inputFromSensor\")
                       "
                },
                "schemaVersion": "1.1",
                "storeAndForwardConfiguration": {
                    "timeToLiveSecs": 7200
                }
            }
        },
        "SampleFilterModule": {
            "properties.desired": {
                "TemperatureThreshold": 25
            }
```

```
        },
        "Modbus": {
            "properties.desired": {
                "PublishInterval": "30000",
                "SlaveConfigs": {
                    "Slave01": {
                        "SlaveConnection": "192.168.2.6",
                        "TcpPort": "503",
                        "HwId": "RudicePLC",
                        "Operations": {
                            "Op01": {
                                "PollingInterval": "30000",
                                "UnitId": "1",
                                "StartAddress": "40001",
                                "Count": "3",
                                "DisplayName": "Blind1"
                            },
                            "Op02": {
                                "PollingInterval": "30000",
                                "UnitId": "1",
                                "StartAddress": "40005",
                                "Count": "3",
                                "DisplayName": "Blind2"
                            },
                            "Op03": {
                                "PollingInterval": "30000",
                                "UnitId": "1",
                                "StartAddress": "40031",
                                "Count": "6",
                                "DisplayName": "Thermostat1"
                            },
                            "Op04": {
                                "PollingInterval": "30000",
                                "UnitId": "1",
                                "StartAddress": "40037",
                                "Count": "6",
                                "DisplayName": "Thermostat2"
                            },
                            "Op05": {
                                "PollingInterval": "30000",
                                "UnitId": "1",
                                "StartAddress": "40043",
                                "Count": "6",
                                "DisplayName": "Thermostat3"
                            },
                            "Op06": {
                                "PollingInterval": "30000",
```

```
                                    "UnitId": "1",
                                    "StartAddress": "40049",
                                    "Count": "6",
                                    "DisplayName": "Thermostat4"
                                }
                            }
                        }
                    }
                }
            }
        }
}
```

# B Výstupy modulů IoT Edge

Výpis B.1: Výstup modulu Modbus, zkráceno


```
{
  "PublishTimestamp":"2021-11-2611:09:06",
  "Content":[
    {
      "HwId":"RudicePLC",
      "Data":[
        {
          "CorrelationId":"DefaultCorrelationId",
          "SourceTimestamp":"2021-11-2611:08:37",
          "Values":[
            {
              "DisplayName":"Blind1",
              "Address":"40001",
              "Value":"100"
            },
            {
              "DisplayName":"Blind1",
              "Address":"40002",
              "Value":"100"
            },
            {
              "DisplayName":"Blind1",
              "Address":"40003",
              "Value":"0"
            },
            {
              "DisplayName":"Thermostat1",
              "Address":"40031",
              "Value":"167"
            },
            {
              "DisplayName":"Thermostat1",
              "Address":"40032",
              "Value":"145"
            },
            {
              "DisplayName":"Thermostat1",
              "Address":"40033",
              "Value":"2"
            },
            {
              "DisplayName":"Thermostat1",
              "Address":"40034",
```

```
            "Value":"0"
          },
          {
            "DisplayName":"Thermostat1",
            "Address":"40035",
            "Value":"145"
          },
          {
            "DisplayName":"Thermostat1",
            "Address":"40036",
            "Value":"2"
          },
          {
            "DisplayName":"Thermostat2",
            "Address":"40037",
            "Value":"162"
          },
          {
            "DisplayName":"Thermostat2",
            "Address":"40038",
            "Value":"145"
          },
          {
            "DisplayName":"Thermostat2",
            "Address":"40039",
            "Value":"2"
          },
          {
            "DisplayName":"Thermostat2",
            "Address":"40040",
            "Value":"0"
          },
          {
            "DisplayName":"Thermostat2",
            "Address":"40041",
            "Value":"145"
          },
          {
            "DisplayName":"Thermostat2",
            "Address":"40042",
            "Value":"2"
          }
        ]
      }
    ]
  }
]
```

```
}
```

Výpis B.2: Výstup modulu Filtr, zkráceno

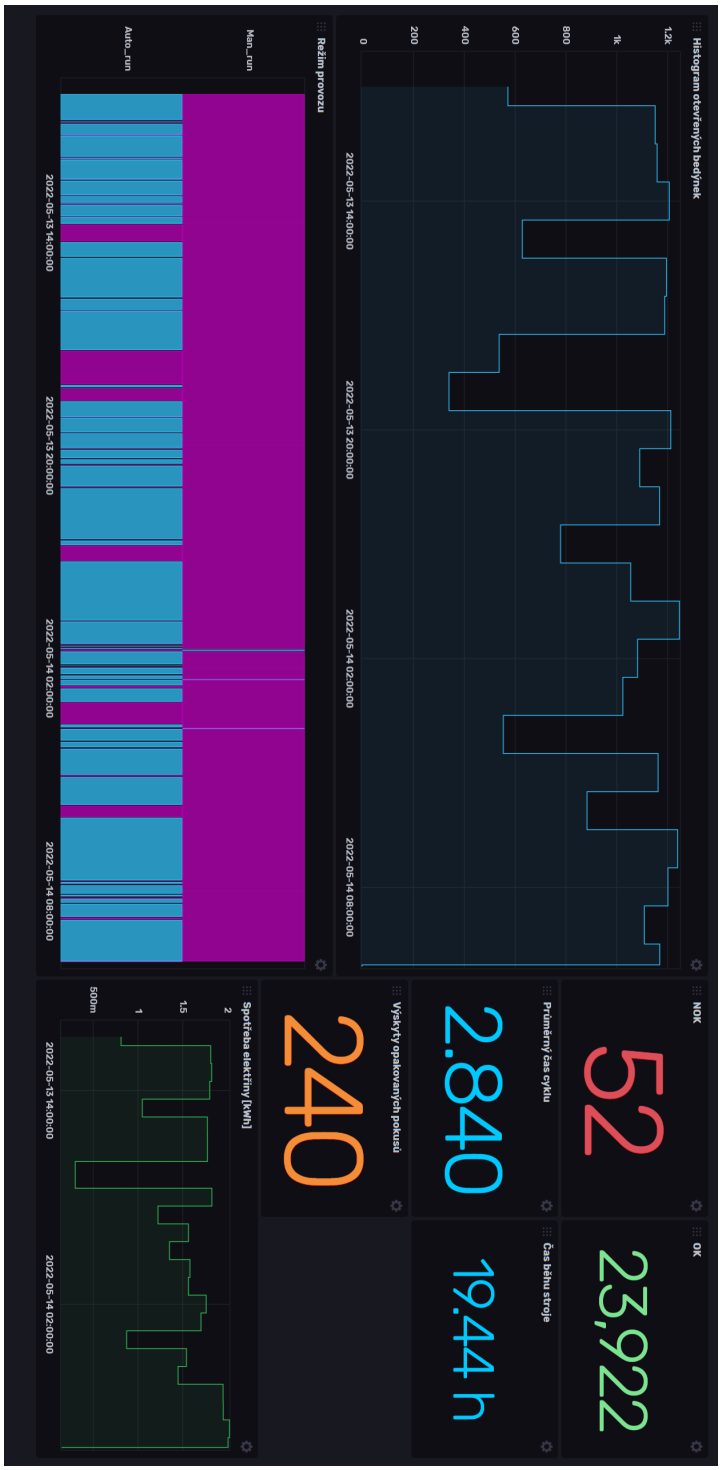
```
[
   {
      "devices":[
         {
            "CurrentTemperature":167,
            "TargetTeperature":145,
            "ValveState":2,
            "CurrentHeatingCoolingState":145,
            "DeviceId":"Thermostat1"
         },
         {
            "CurrentTemperature":162,
            "TargetTeperature":145,
            "ValveState":2,
            "CurrentHeatingCoolingState":145,
            "DeviceId":"Thermostat2"
         },
         {
            "CurrentPosition":100,
            "TargePosition":100,
            "CurrentHorizontalTiltAngle":0,
            "DeviceId":"Blind1"
         }
      ],
      "timeCreated":"2021-11-2611:08:37"
   }
]
```

Výpis B.3: Příklad výstupu modulu OPC Publisher

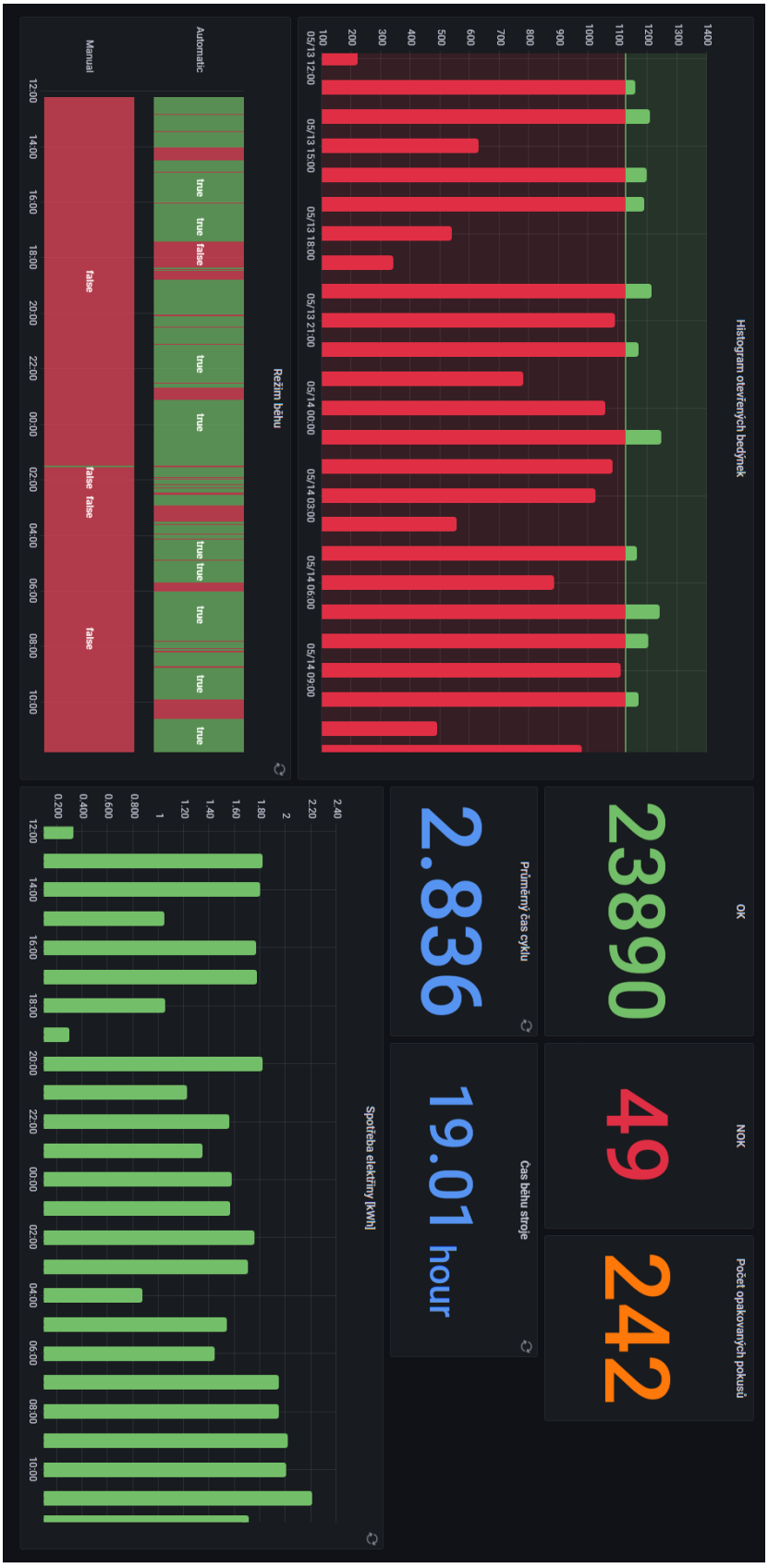
```
  {
    "NodeId": "ns=3;s=\"DB_Database_Openers_OA\".\"Openers\"[0]",
    "EndpointUrl": "opc.tcp://opctestserver:4840/",
    "DisplayName": "Openers",
    "Value": {
      "Value": {
        "Timestamp": [ 34, 4, 41, 25, 3, 54, 54, 54 ],
        "Cycle_Time": 2800,
        "Size": 1,
        "Opening_Attempts": 1,
        "Opened_OK": true
      },
      "SourceTimestamp": "2022-04-29T17:03:36.7747917Z"
    }
```

```
  },
  {
    "NodeId": "ns=3;s=\"DB_Consumption\".\"Air\".\"Current_Flow\"",
    "EndpointUrl": "opc.tcp://opctestserver:4840/",
    "DisplayName": "CurrentFlow",
    "Value": {
      "Value": 238,
      "SourceTimestamp": "2022-04-29T17:03:41.7748957Z"
    }
  }
```

# C Vizualizace




Obr. C.1: Vytvořená vizualizace ve webové aplikaci InfluxDB




Obr. C.2: Vytvořená vizualizace ve webové aplikaci Grafana

# D Obsah elektronické přílohy

- /
  - `PrvniReseni` ..... Soubory k prvnímu řešení
    - `IoT Edge` ..... Zdrojový kód modulu filtrování dat, Deploymenty Manifest
    - `Time Series Insights` ..... Konfigurace služby Time Series Insights
  - `DruheReseni` ..... Zdrojové soubory k druhému řešení
    - `IoTEdge` ..... Deployment Manifest
    - `Azure Functions` ..... Zdrojový kód deserializace
    - `Azure konfigurace` ..... Konfigurace nasazených služeb v Azure
    - `Grafana` ..... Konfigurace Grafana
    - `InfluxDB` ..... Konfigurace InfluxDB