SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Evidenčné číslo: FEI-16605-115084

WEB APLIKÁCIA NA SPRÁVU OPC UA SERVEROV

BAKALÁRSKA PRÁCA

2024

Peter Likavec

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Evidenčné číslo: FEI-16605-115084

WEB APLIKÁCIA NA SPRÁVU OPC UA SERVEROV

BAKALÁRSKA PRÁCA

Študijný program :	Aplikovaná informatika
Číslo študijného odboru:	2511
Názov študijného odboru:	9.2.9 Aplikovaná informatika
Školiace pracovisko:	Ústav informatiky a matematiky
Vedúci záverečnej práce:	Ing. Rudolf Pribiš, PhD.
Konzultant ak bol určený:	Meno konzultanta

2024

Peter Likavec

Slovenská technická univerzita v Bratislave
Ústav automobilovej mechatroniky

Fakulta elektrotechniky a informatiky
Akademický rok: 2023/2024
Evidenčné číslo: FEI-16605-115084

STU FEI

ZADANIE BAKALÁRSKEJ PRÁCE

Študent:	**Peter Likavec**
ID študenta:	115084
Študijný program:	aplikovaná informatika
Študijný odbor:	informatika
Vedúci práce:	Ing. Rudolf Pribiš, PhD.
Vedúci pracoviska:	Ing. Ján Cigánek, PhD.
Miesto vypracovania:	ÚAMT

Názov práce: **Web aplikácia na správu OPC UA serverov**

Jazyk, v ktorom sa práca vypracuje: slovenský jazyk

Špecifikácia zadania:

Cieľom práce je návrh a implementácia web aplikácie na správu OPC UA serverov s použitím technológií: ASP .NET/.NET core, MVC, AngularJS, ReactJS, prípadne obdobné technológie.
Úlohy:
1. Analyzujte aktuálny stav pre oblasti Ad-hoc konektivity, digitálny opis aktíva pre I40, infraštruktúry AAS.
2. Navrhnite riešenie zabezpečujúce ad-hoc konektivitu pomocou OPC UA technológie s využitím štandardizovaného digitálneho opisu cez Asset Administration Shell a správou cez webovú aplikáciu.
3. Implementujte navrhnuté riešenie.
4. Overte riešenie nasadením pomocou kontajnerizácie alebo cez IIS.

Rozsah práce: 30 až 40 strán (54 000 až 72 000 znakov)

Zoznam odbornej literatúry:

1. PRIBIŠ, Rudolf; BEŇO, Lukáš; DRAHOŠ, Peter. Asset administration shell design methodology using embedded OPC unified architecture server. *Electronics,* 10. s. 20.
2. PRIBIŠ, Rudolf; DRAHOŠ, Peter. *Digitálne technológie pre sémantickú interoperabilitu v Industry 4.0: dátum obhajoby 22.8.2022.* Dizertačná práca. Bratislava : 2022. 199 s.
3. PRIBIŠ, Rudolf; HAFFNER, Oto; BEŇO, Lukáš; JANECKÝ, Dominik; KUČERA, Erik; PAJPACH, Martin; PAĽA, Adam; HAMRÁK, Michal; ZAJAC, Šimon. Emerging Trends in Education For Industry 4.0 and 5.0 Engineers in Slovakia Using E-learning Web Applications. In: *MoSICom 2023.* Piscataway: IEEE, 2023, ISBN 979-8-3503-9341-5.

Termín odovzdania bakalárskej práce:	31. 05. 2024
Dátum schválenia zadania bakalárskej práce:	20. 02. 2024
Zadanie bakalárskej práce schválil:	prof. Dr. rer. nat. Martin Drozda – garant študijného programu

SÚHRN

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Študijný program :	Aplikovaná informatika
Vyberte typ práce	Web aplikácia na správu OPC UA serverov
Autor:	Peter Likavec
Vedúci záverečnej práce:	Ing. Rudolf Pribiš, PhD.
Konzultant ak bol určený:	Meno konzultanta
Miesto a rok predloženia práce:	Bratislava 2024

Bakalárska práca sa zaoberá získaním analýzou informácií o OPC UA serveroch, administratívnej schránke aktív a ich prepojeniu s Industry 4.0, ktoré sú základným pilierom vedomostí webovej aplikácie. Úvodná časť je zameraná na získanie vedomostí a porozumeniu oblasti ohľadom Internet vecí, administratívnej schránke aktív, OPC UA serverov a Industry 4.0. Nasledujúcou časťou je spraviť analýzu pre možnosti riešenia implementácie a zvoliť vhodné technológie pre návrh a vývoj aplikácie. Cieľom bakalárskej práce je vďaka získaným poznatkom navrhnúť, implementovať a následne nasadiť aplikáciu v IIS s využitím vhodných technológií, ktorá ma slúžiť na prácu s OPC UA servermi, aby mal užívateľ k dispozícii jednoduchý nástroj na prácu s OPC UA servermi. Realizácia práce obsahuje štruktúru webovej aplikácie, ktorá pozostáva z frontendu a backendu. Riešenie zahŕňa detailný opis aplikácie ako aj zoznam a vysvetlenie všetkých použitých technológií, prehľad funkcionality a dôležitých častí, ktoré sú obsiahnuté v užívateľskej príručke. V závere práce sú zdokumentované a zhodnotené výsledky projektu spolu s odporúčaním pre ďalší rozvoj aplikácie.

Kľúčové slová: Webová aplikácia, Industry 4.0, AAS, RAMI 4.0, React.js, ASP .NET

ABSTRACT

SLOVAK UNIVERSITY OF TECHNOLOGY IN BRATISLAVA

FACULTY OF ELECTRICAL ENGINEERING AND INFORMATION TECHNOLOGY

Study Programme:	Applied Informatics
Bachelor Thesis:	Web application for managing OPC UA servers
Autor:	Peter Likavec
Supervisor:	Ing. Rudolf Pribiš PhD.
Consultant:	Meno konzultanta
Place and year of submission:	Bratislava 2024

The bachelor's thesis deals with obtaining and analyzing information about OPC UA servers, Asset Administration Shell and their connection with Industry 4.0, which are the basic pillar of web application knowledge. The introductory part is aimed at gaining the knowledge and understanding of the area regarding the Internet of Things, Asset Administration Shell, OPC UA servers and Industry 4.0. The next part is to make an analysis for the possibilities of the implementation solution an choose suitable technologies for the design and development of the application. The goal of the bachelor's thesis is to design, implement and the deploy and application in IIS using appropriate technologies, which will server to work with OPC UA servers, so user has a simple tool for working and managing OPC UA servers. The implementation of the work include structure of the web application which consist of backend and frontend. The solution includes a detailed description of the application as well as a list of all technologies used, and overview of functionality and important parts. At the end of the work, the results of the project are documented and evaluated together with recommendation for further development of application.

Key words: Web application, Industry 4.0, AAS, RAMI 4.0, React.js, ASP .NET

Vyhlásenie autora

Podpísaný Peter Likavec čestne vyhlasujem, že som Bakalársku prácu Web aplikácia na správu OPC UA serverov vypracoval na základe poznatkov získaných počas štúdia a informácií z dostupnej literatúry uvedenej v práci.

Uvedenú prácu som vypracoval pod vedením Ing. Rudolf Pribiš, PhD..

V Bratislave dňa 06.03.2024

.....

podpis autora

Poďakovanie

Týmto spôsobom by som chcel poďakovať vedúcemu práce Ing. Rudolf Pribiš, Phd. za usmernenie pri písaní práce a odbornú pomoc pri riešení práce.

Obsah

Je potrebné aktualizovať pole obsahu, aby sa zobrazili aktuálne čísla strán.

Zoznam obrázkov a tabuliek

Zoznam skratiek a značiek

AAS	Asset Administration Shell
AASX	Súborový formát balíka pre AAS
API	Application Programming Interface
CSS	Cascading Style Sheets
DOM	Document Object Model
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
IIS	Internet Information Services
IoT	Internet of Things
IIoT	Industrial Internet of Things
JSON	JavaScript Object Notation
NPM	Node Package Manager
OPC UA	Open Platform Communications Unified Architecture
RAMI 4.0	Reference Architectural Model Industrie 4.0
REST API	Representational State Transfer API
SQL	Structured Query Language
XML	Extensible Markup Language
MVC	Model-View-Controller
URN	Uniform Resource Name

Úvod

Industry 4.0, známa aj ako štrvrtá priemyselná revolúcia, predstavuje revolučný posun v oblasti priemyselnej výroby, kde sa inteligentné technológie a kybernetické systémy prelínajú s tradičnými výrobnými procesmi. V tomto novom priemyselnom paradgme zohráva kľúčovú úlohu komunikácia a interoperabilita, čo prinása do popredia technológiu OPC UA.

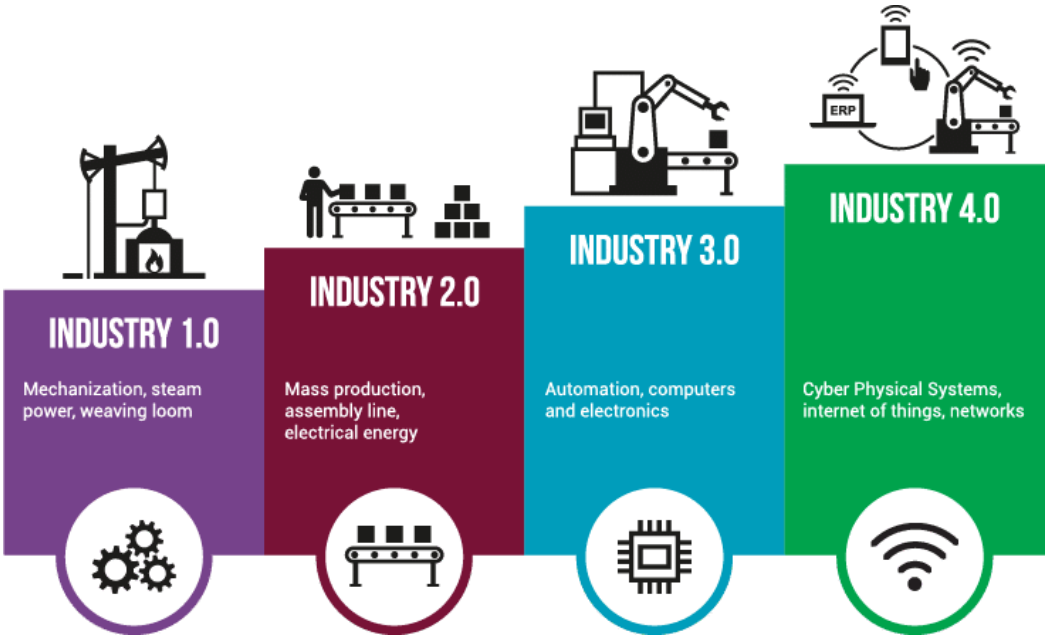
Technológie OPC UA serverov sa stali rozhodujúcim prvkom v kontexte Industry 4.0, poskytujúc spoľahlivý a bezpečný spôsob prenosu dát medzi rôznymi zariadeniami a systémami. Ich využitie umožňuje efektívne riadenie a monitorovanie výrobných procesov, a to nielen v rámci jedného podniku.

Cieľom bakalárskej práce je dostatočné porozumenie materiálov týkajúcich sa témy a následne spracovať nadobudnuté poznatky. Následne ďalšou časťou je získané informácie analyzovať a určiť potrebné technológie pre vývoj aplikácie a vytvoriť si vedomostnú bázu o danej téme. Koncovým bodom bakalárskej práce na základe podmienok navrhnúť aplikáciu. Medzi hlavné požiadavky aplikácie je užívateľsky prijateľné prostredie s funkcionalitou pre manažovanie OPC UA serverov. Hlbšej špecifikácií funkcionality sa budeme venovať v ďalších kapitolách bakalárskej práce.

1 Teoretická časť

1.1 Industry 4.0

Industry 4.0 označuje novú fázu priemyselnej revolúcie, ktorá môže byť definovaná integráciou inteligentných digitálnych technológií do výroby a priemyselných procesov. Industry 4.0 umožňuje firmám a spoločnostiam inteligentnú výrobu a vytvorenie takzvaných inteligentných prevádzok. Cieľom je vylepšenie produktivity, efektívnosti a flexibility. Industry 4.0, ktorá veľmi úzko súvisí s IIoT (Industrial Internet of Things) a zautomatizovanú výrobu spája fyzickú výrobu s operáciami s inteligentnými digitálnymi technológiami, strojovým učením a prácu s dátami aby vytvorili lepší ekosystém pre spoločnosti, ktoré sa zameriavajú na výrobu. Industry 4.0 je založená na 9 technologických pilieroch, pričom skutočná sila Industry 4.0 je dosiahnutá práve použitím technológií spoločne.



1.2 OPC UA

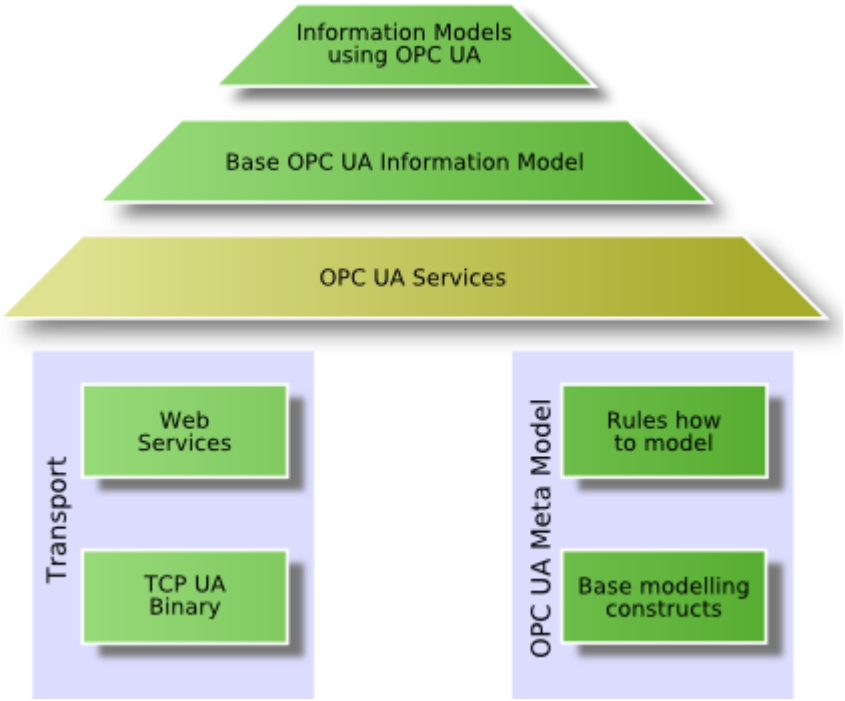
OPC Unified Architecture je priemyslový komunikačný štandard ktorý definovala organizácia OPC Foundation. Od pôvodnej špecifikácie OPC založená firmou Microsoft (funguje iba v OS Windows), je OPC UA technológia založená na obecne používaných

komunikačných štandardoch ako sú TCP/IP, http alebo SOAP. Vďaka tomu môže OPC UA fungovať aj na iných platformách ako je Windows.

OPC UA sa používa v priemyselných doménach, ako sú priemyselné senzory a akčné členy, riadiace systémy, systémy na vykonávanie výroby a systémy plánovania podnikových zdrojov, vrátane priemyselného internetu vecí (IIoT), Machine To Machine (M2M). Tieto systémy sú určené na výmenu informácií a na používanie príkazov a riadenia priemyselných procesov. OPC UA definuje spoločný model infraštruktúry na uľahčenie tejto výmeny informácií. OPC UA špecifikuje nasledovné:

- informačný model reprezentujúci štruktúru, správanie a sémantiku;
- model správy na interakciu medzi aplikáciami;
- komunikačný model na prenos údajov medzi koncovými bodmi;
- model zhody na zaručenie interoperability medzi systémami

https://documentation.unified-automation.com/uasdkcpp/1.5.1/html/L2OpcUaOverview.html



1.2.1 OPC UA a vzťah s Industry 4.0

OPC UA hra kľúčovú rolu v rámci priemyslu 4.0, poskytuje štandardizovaný a platformovo nezávislý komunikačný protokol, na bezproblémovú komunikáciu medzi rôznymi priemyselnými zariadeniami. Podporuje veľké množstvo komunikačných

mechanizmov vrátane výmeny údajov v reálnom čase, oznamovania udalostí,... OPC UA podporuje bezpečnosť bezpečnostnými funkciami ako sú napr. šifrovanie, autentifikácia, autorizácia,...

1.2.2 Informačný model

Informačný model v OPC UA je Štandardizovaná reprezentácia štruktúry, organizácie alebo dát v systéme. Model nám umožňuje definovať ako sú informácie modelované, vymieňané medzi rôznymi zariadeniami a systémami, ktoré komunikujú prostredníctvom OPC UA. Hlavnými komponentami informačného modelu sú:

- Objekty a Nodes – Node obsahuje jednotlivé časti informácie ako sú premenné, metódy alebo eventy, objekty pozostáva z viacerých nodes a formuje skupinu dát
- Atribúty – Nodes v informačnom modely majú atribúty, ktoré definujú dáta (hodnotu premennej, dátový typ,...)
- Referencie – definujú vzťahy medzi jednotlivými nodes
- Dátové typy – OPC UA definuje set štandardizovaných dátových typov, ktoré popisujú dáta a ich hodnoty
- Metódy – Informačný model môže zahŕňať metódy, ktoré reprezentujú funkcie alebo operácie ktoré môžu byt spustené

Informačný model predstavuje framework na popis štruktúr a sémantických dát, zabezpečenie konzistentnosti a interoperability medzi rôznymi zariadeniami.

Aby sme to zhrnuli, informačný model OPC UA je štandardizovaná definícia jedinečných uzlov v adresnom priestore servera OPC UA. Jedinečnosť uzlov zabezpečuje ID uzla. A každý objekt, systém zariadení alebo aj celú továreň je možné reprezentovať pomocou prepojenia uzlov a ich vzťahov v adresnom priestore OPC UA servera je možné reprezentovať pomocou prepojenia uzlov a ich vzťahov v adresnom priestore OPC UA servera.

1.2.3 Adresný priestor OPC UA

Hlavnou úlohou AddressSpace v OPC UA je poskytnúť štandardný spôsob pre servery na reprezentáciu objektov pre klientov. AddressSpace sa chová ako virtuálny priestor kde dáta metódy a ďalšie komponenty sú organizované v hierarchicky. AddressSpace je modelovaný pomocou Nodes, ktoré sú sprístupnené pre klientov pomocou OPC UA

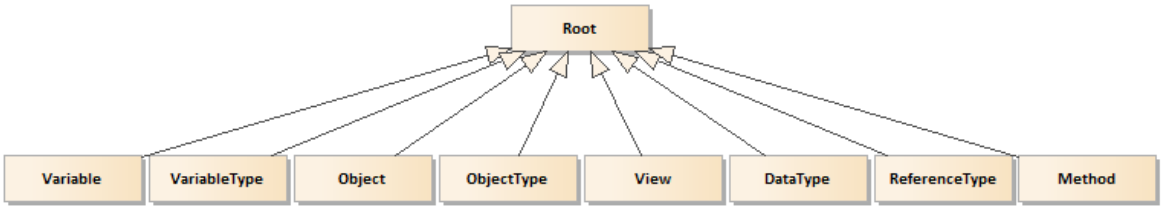
services. Nodes v AddressSpace sa používajú na reprezentáciu reálneho objektu, ich definícii a referencii medzi sebou.

https://commsvr.gitbook.io/ooi/semantic-data-processing/addressspaceaddressspacemodel

https://reference.opcfoundation.org/Core/Part3/v105/docs/4.3

Model adresného priestoru je definovaný nasledovným setom typu Node:

- View: Definuje podmnožinu uzlov v adresnom priestore
- ObjectType: Poskytuje definíciu objektov
- Object: Používa sa na reprezentáciu komponentov, systémov, objektov reálneho sveta a softvérových objektov
- ReferenceType: Používa sa na definovanie vzťahov medzi uzlami (nodes)
- DataType: Používa sa na definovanie jednoduchých a komplexných hodnôt premennej
- VariableType: Používa sa na definovanie typu premennej
- Variable: Používa sa ako úložisko real-time premennej, ktorá obsahuje hodnotu
- Method: Je funkcia ktorej rozsah je ohraničený objektom, kde je definovaná

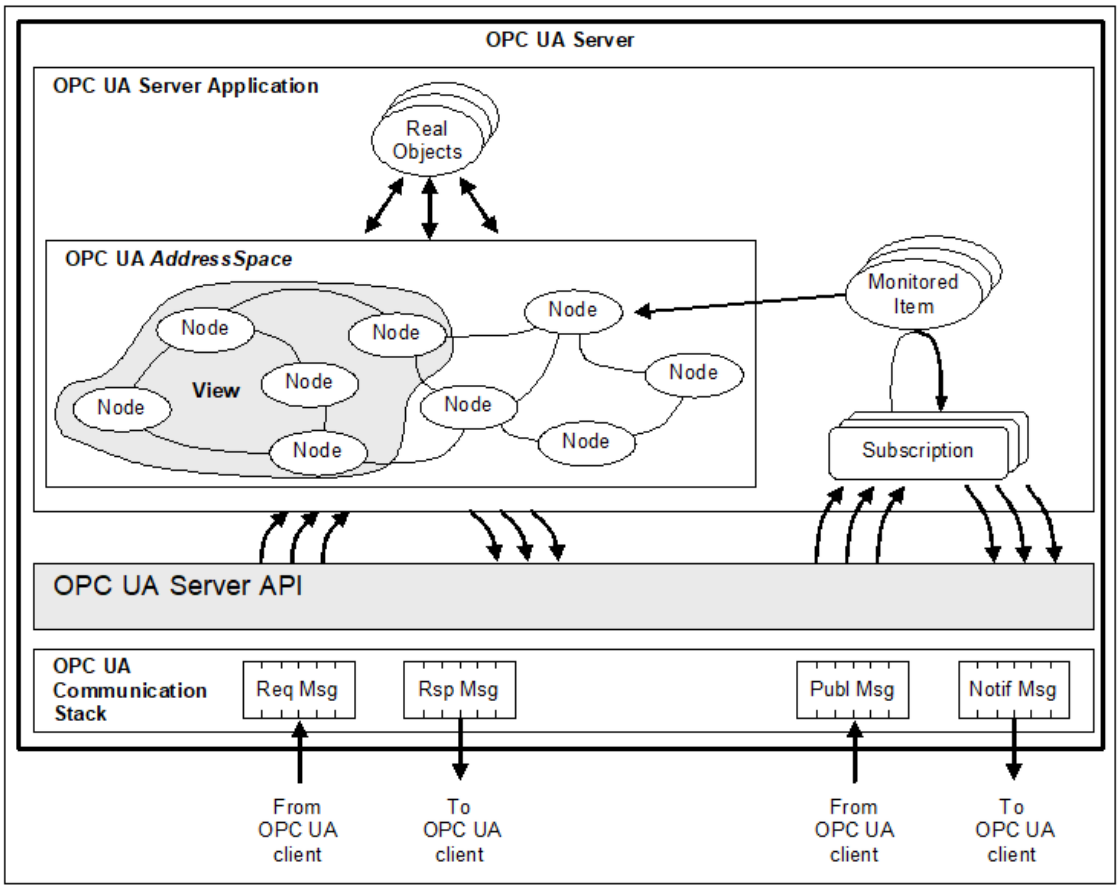


Obr. 1: Adresný priestor model [1]

1.2.4 OPC UA Server

OPC UA server je architektonický model serverového endpointu pre klientske/serverové interakcie.

- Reálne objekty - sú fyzické alebo softwarové objekty sprístupnené pre serverovú aplikáciu.
- Serverová aplikácia - je kód, ktorý implementuje funkcionalitu serveru, využíva Server API na posielanie a prijímanie správ od klienta. Server API je interný interface ktorý izoluje aplikáciu od OPC UA komunikačného stacku.
- AddressSpace - je modelovaný ako kolekcia Nodes sprístupnené pre klienta použitím OPC UA services.
- AddressSpace View – slúžia na reštrikciu Nodes, ktoré server umožní vidieť klientovi
- MonitoredItems – sú entity v servery vytvorené klientom, ktoré monitorujú Nodes v AddressSpace.



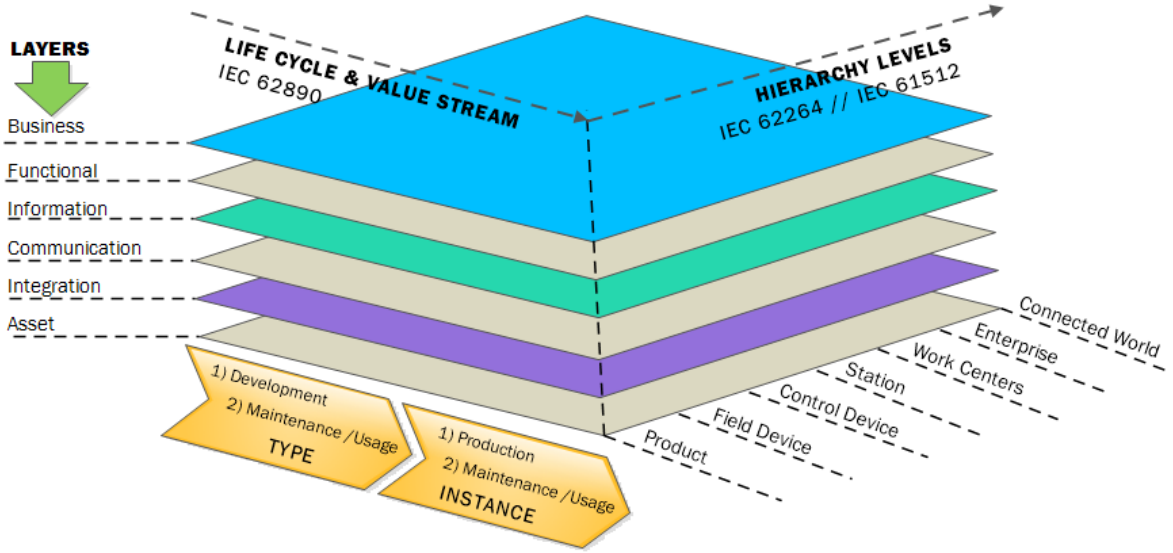
Obr. 2

1.3 RAMI 4.0

https://industry40.co.in/rami-reference-architecture-model-industry-4-0/

RAMI 4.0 alebo Reference Architectural Model Industry 4.0 je trojrozmerná mapa, ktorá ukazuje ako sa vysporiadať s problematikou Industry 4.0 systematickým a štrukturovaným spôsobom. RAMI 4.0 je jednotný model pre všetky komponenty, ktorý zabezpečuje účastníkom zapojeným do ekosystému Industry 4.0 zdieľanie dát a informácii efektívne. RAMI 4.0 mapuje všetkých účastníkov prepojených v priemyselnom odvetví do troch osí definície:

- Os vrstiev architektúry
- Os životného cyklu a hodnotového toku
- Os úrovní hierarchie



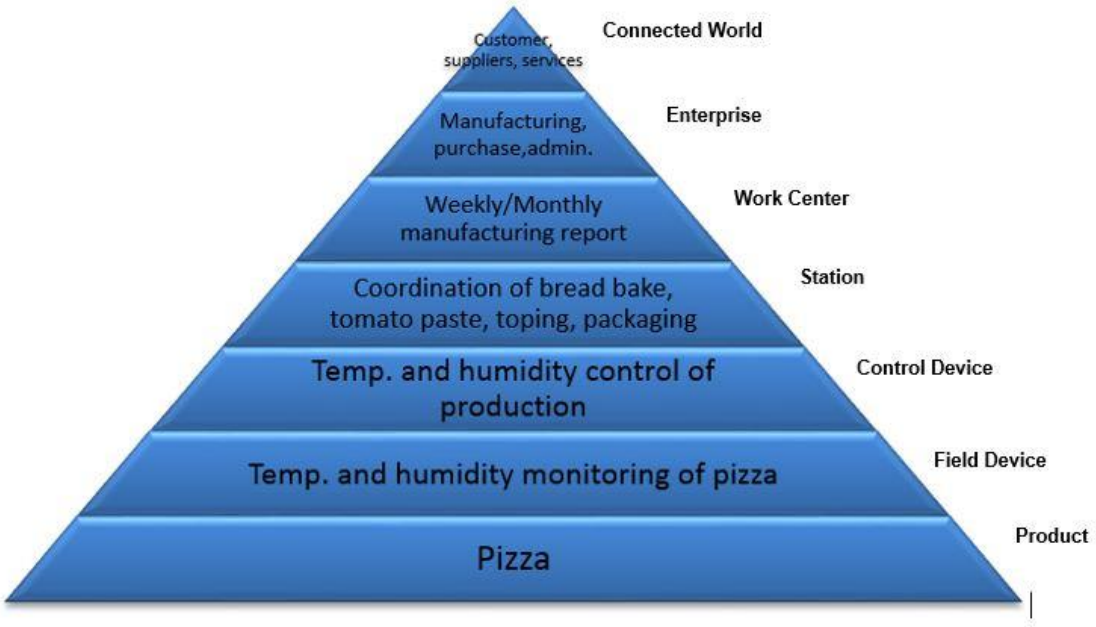
Obr. 3: co

1.3.1 Os úrovní hierarchie

Industry 4.0 architektúra na hierarchickej úrovni ukazuje funkčné priradenie komponentov. Táto os v rámci podniku alebo závodu sa riadi normami IEC 62264 a IEC 61512. Úroveň nad a pod oblasťou noriem IEC predstavuje kroky a popisuje ako skupiny továrni spoluprácu v rámci externých firiem.

Úrovne hierarchie sú:

- Produkt
- Poľné zariadenie
- Riadiace zariadenie
- Stanica
- Pracovné centrum
- Podnik
- Pripojený svet

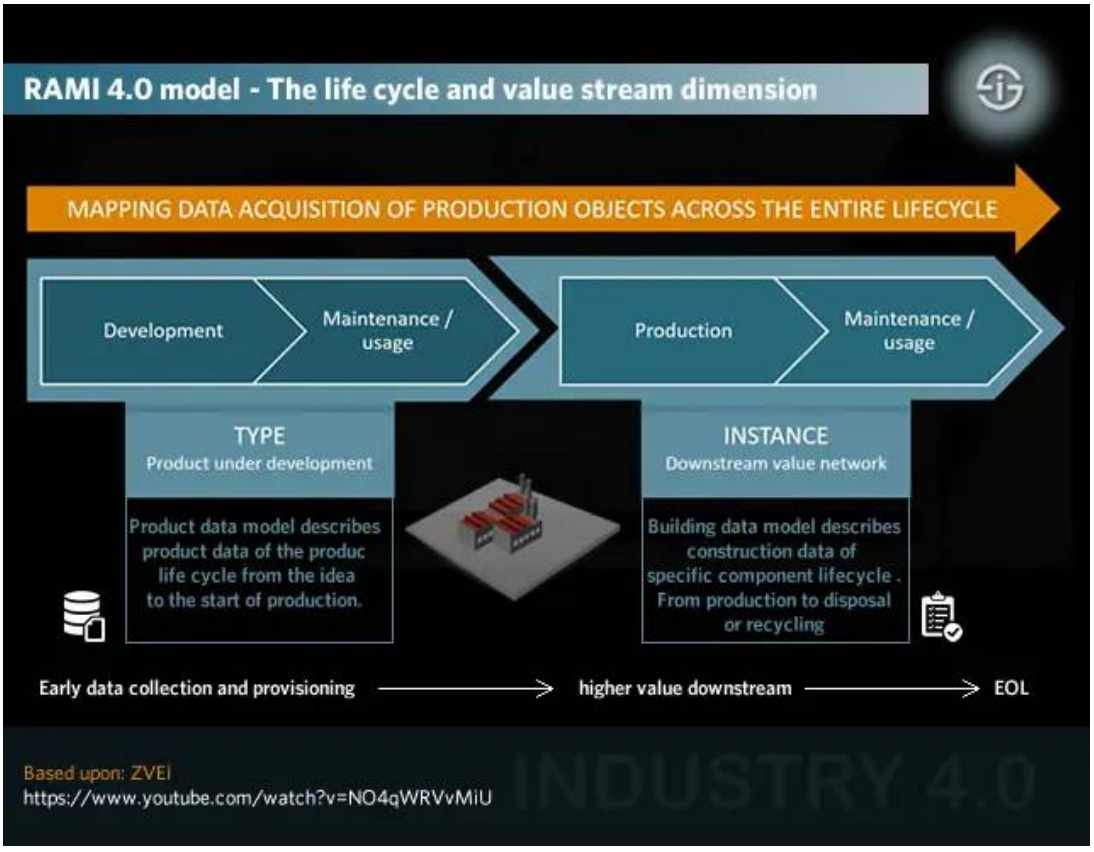


Obr. 4: Opis hierarchického levelu na príkladu výroby pizze

1.3.2 Os životného cyklu a hodnotového toku

Os životného cyklu a hodnotového toku je rozdelená na dve časti a sú nimi typ a inštancia. Typ je rozdelený na vývoj a údržbu/použitie, kým inštancia je rozdelená na produkciu a údržbu/použitie. Typ reprezentuje počiatočnú myšlienku vývoja produktu, kým každý vyrobený produkt reprezentuje inštanciu tohto typu. Inak povedané kým je

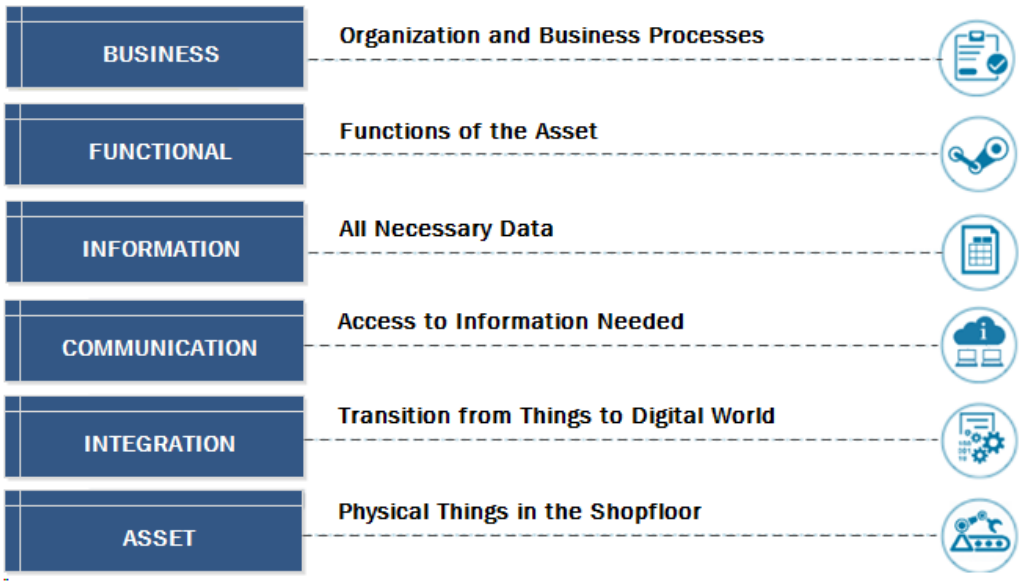
produkt v štádiu vývoja označujeme ho ako “typ”, akonáhle sa presunie do výroby označujeme ho “inštanciou”. Hocikedy kedy je produkt redizajnovaný alebo je pridaná nová vlasnosť, opäť sa stáva typom.



Obr. 5: Mapovanie získavania údajov o výrobe počas celého životného cyklu - https://www.i-scoop.eu/industry-4-0/

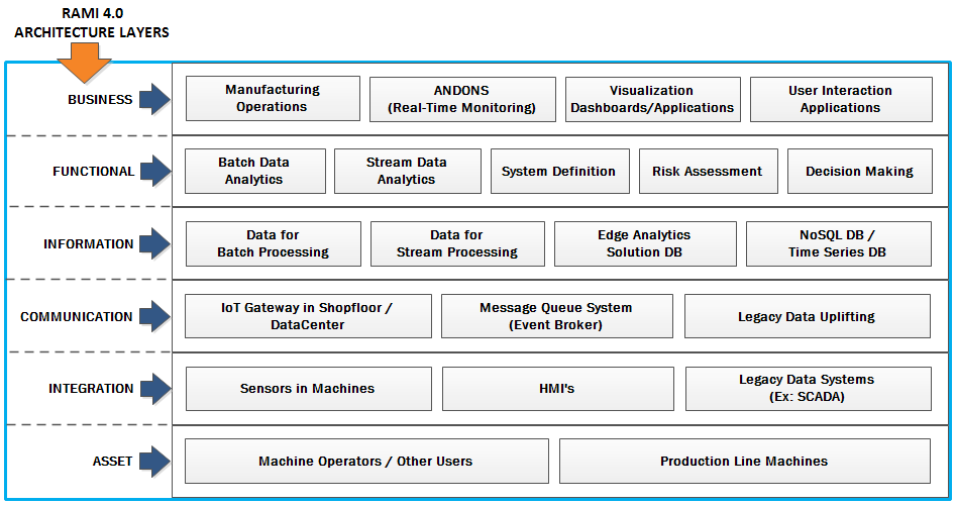
1.3.3 Os vrstiev architektúry

Vrstvy architektúry zahŕňa vrstvy, ktoré umožňujú vývoj Industry 4.0 softvérových riešení konzistentným spôsobom tak, aby boli rôzne a vzájomne závislé výrobné operácie prepojené s prihliadnutím na fyzický a digitálny svet.



Obr. 6: Vrstvy architektúry RAMI 4.0

RAMI 4.0 rozdeľuje komplexné procesy do častí, čo umožňuje jednoduchšie pochopenie a už od začiatku zahŕňa ochranu údajov a bezpečnosť IT. Odpovedá na všetky otázky k problematike ohľadom sémantiky, identifikácie, funkcií, komunikačných štandardoch pre inteligentnú továreň. S RAMI 4.0 architektúrou je továreň nie je vrchnou vrstvou ale sieťou interakcii medzi inteligentnými produktami a svetom.



Obr. 7: Zobrazenie architektúry riešenia analýzy RAMI 4.0

1.4 IoT – Internet of Things

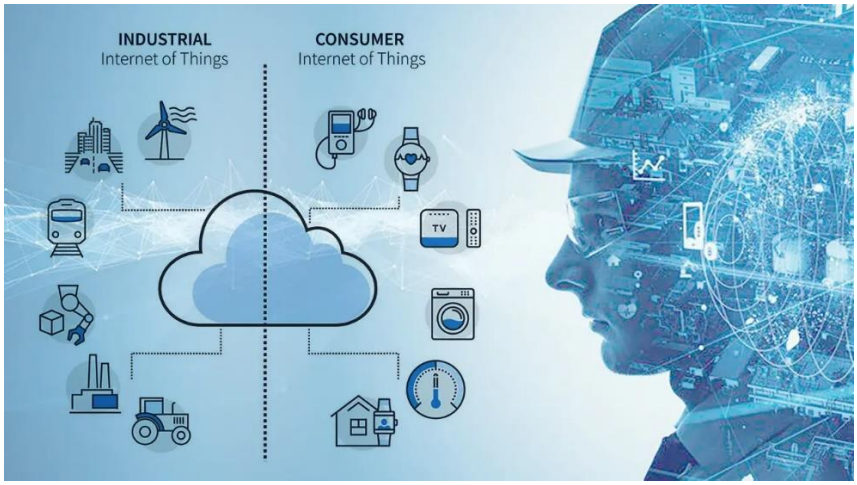
https://www.oracle.com/internet-of-things/what-is-iot/

Internet vecí opisuje site fyzických objektov – zariadení, ktoré obsahujú sensory, softvér a iné technológie na účel komunikácie a výmeny údajov s inými zariadeniami cez internet. Tieto zariadenia siahajú od bežných domácich zariadení až po sofistikované premyslné stroje. Medzi zariadenia patriace do IoT patrí každé zatiadenie, ktoré môže byť monitorované alebo ovládané zo vzdialenej lokácie.

1.5 IIoT – Industrial Internet of Things

https://www.iberdrola.com/innovation/what-is-iiot

Industrial Internet of Things je kolekcia senzorov, nástrojov a autonómnych zariadení pripojených cez internet k priemyselným aplikáciam. Táto sieť umožňuje zhromažďovať dáta, vykonávať analýzy a optimalizovať výrobu, zvyšovať efektivitu a znižovať náklady na výrobný proces a poskytovanie služieb. Priemyselné aplikácie sú kompletné technologické systémy, ktoré prepájajú zariadenia a tie následne s pracovníkmi, ktorí riadia procesy. Súčasné IIoT aplikácie sa predovšetkým sustreďujú na výrobu, dopravu a energetiku. Rozdielom medzi IoT a IIoT je že kým Internet of Things sa zameriava na služby pre zákaznikov (spotrebiteľov), Industrial Internet of Things sa zameriava na zvýšenie bezpečnosti a efektivity vo výrobných prevádzkach. Pre príklad sa spotrebiteľské riešenie zamerali hlavne na riešenia inteligentných zariadení pre domácnosť, virtuálnych asistentov či teplotné senzory... Medzi zariadenie patriace do IIoT sa považujú zariadenia, ktoré musia byť sieťové systémy, ktoré generujú dáta pre analýzu.



Obr. 8 Industrial Internet of Things vs Internet of Things

1.6 Asset Administration Shell

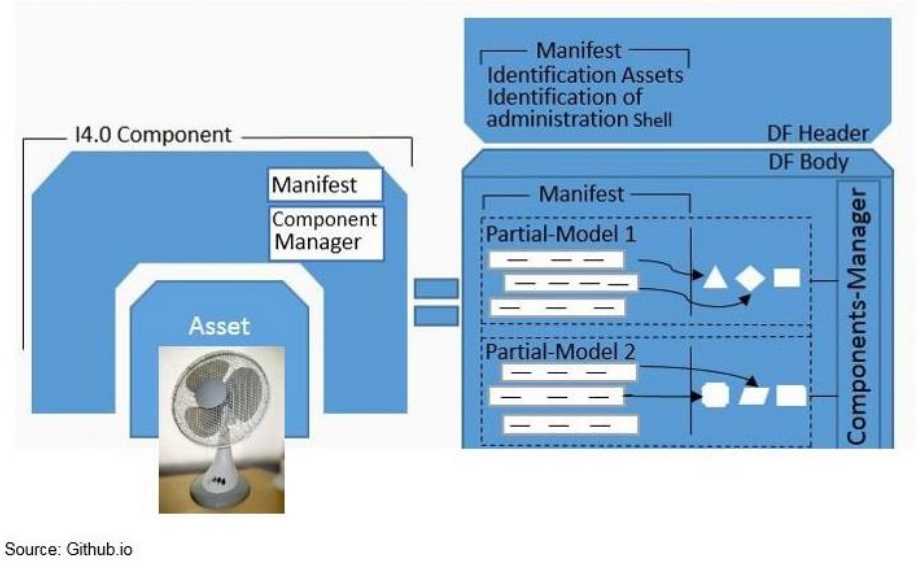
Asset Administration Shell (AAS) je štandardizovaná digitálna reprezentácia aktíva, základným prvokom pre interoperabilitu pre komponenty v Industrie 4.0 systémoch. Industrie 4.0 komponenty sú kombináciou aktíva (senzor, stroj,..) a jej digitálnej reprezentácie, AAS. AAS môže byť logickou reprezentáciou jednoduchého komponentu alebo stroja. AAS pozostáva z počtu submodelov, v ktorých sú opísané všetky informácie funkcionalita daného aktíva, ktorá zahŕňa vlastnosti, charakteristiku, nastavenia, statusy, parametre a namerané dáta. Štruktúra AAS je definová pomocou rôznych technologicky nezávislých modelov a pomocou mapovania ako je XML, JSON alebo OPC UA. Jeho obsah je definovaný prostredníctvom submodelu špecifického pre doménu.

AAS má momentálne 3 používané typy:

- Type 1 Asset Administration Shell – Súbory typu JSON alebo XML, obsahujú statické informácie a môžu byť distribuované ako súbor
- Type 2 Asset Administration Shell – Existujú ako bežiace inštancie, sú hostované na servery, môžu obsahovať statické informácie ale môžu interagovať s ostatnými komponentami. Type 2 AAS poskytujú frontend napríklad pre zariadenia, live dáta zo senzorov...
- Type 3 Asset Administration Shell – Sú nadstavbou Type 2 AAS, avšak navyše dokážu samé od seba začať komunikáciu medzi sebou

1.6.1 Štruktúra AAS

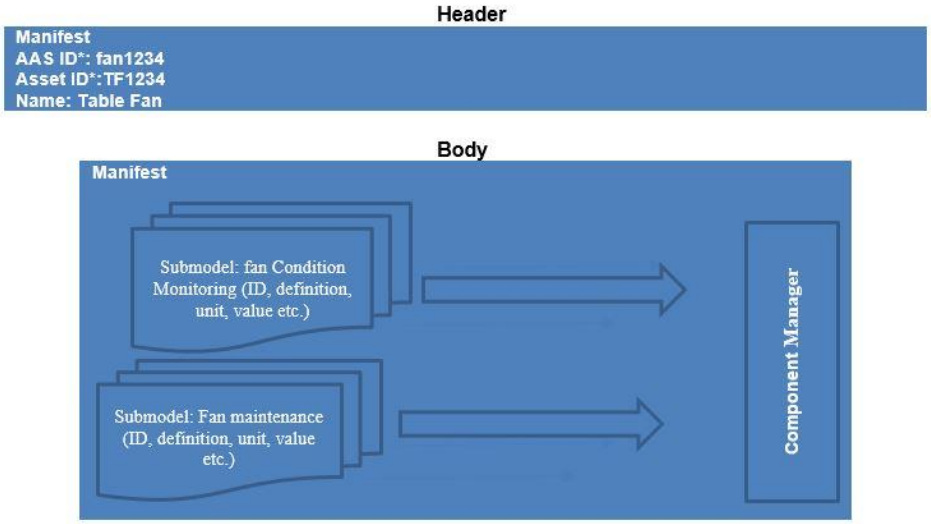
AAS pozostáva z hlavičky a tela. Hlavička obsahuje informáciu pre identifikáciu, administráciu a použitie aktíva, subkomponenty a administration shell ako celok. Tieto informacie su uložené v časti manifest v hlavičke. Telo AAS obsahuje submodely ktoré obsahujú hierarchicky organizované nastavenia aktíva. Tieto nastavenia obsahujú vlastnosti ktoré referujú dátam a metódam ktoré využíva aktívum. Telo AAS obsahuje takisto manifest ktorý pozostáva z listu všetkých submodelov.



Obr. 9: Štruktúra AAS všeobecne

1.6.2 AAS Submodel

AAS obvykle obsahuje viacero submodelov. Submodely definujú vlastnosti a služby (metódy, funkcie). Submodely môžu obsahovať vlastnosti, funkcie, eventy, referencie, vzťahy. Toto umožň1uje poskytovanie veľké množstvo údajov pre submodely. AAS používa striktný for mát ktorý organizuje dáta ako strom vlasností. Rovnaký formát je použitý pre štruktúru vlastností submodelov. AAS a submodely definujú API pre získanie AAS informácii ako aj informácii v AAS submodeloch.



Obr. 10: AAS Submodel príklad

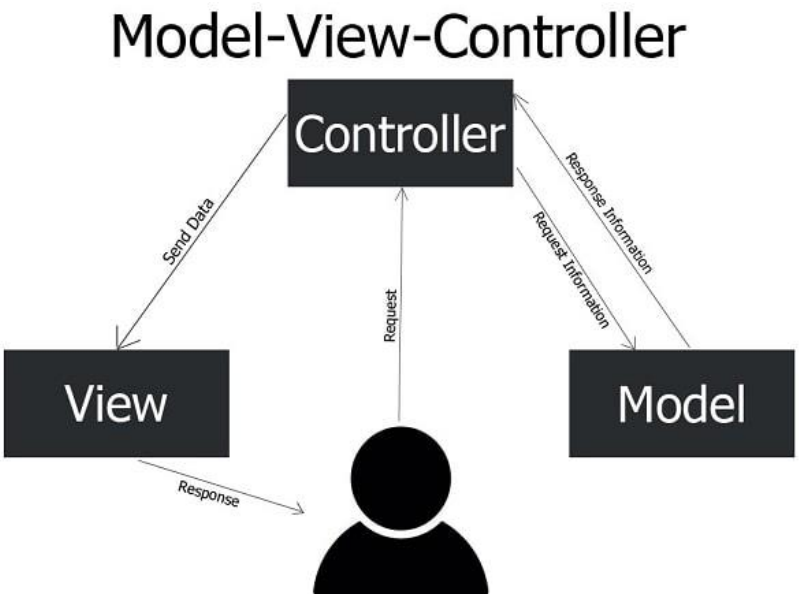
1.7 MVC

Model-view-controller je architektonický vzor, ktorý rozdeľuje aplikáciu do troch hlavných skupín komponentov:

- Model
- View
- Controller

Tento vzor pomáha dosiahnúť separáciu záujmov pri tvorbe web aplikácie. Použitím tohto vzoru používateľské requesty su presmerované do controllera, ktorý je zodpovedný za prácu s modelom aby urobil danú operáciu pre používateľa a vrátil výsledky. Controller následne používa View na zobrazenie výsledkov pre používateľa s výsledkami operácie.

ZDROJ - https://www.geeksforgeeks.org/mvc-design-pattern/



Obr. 11: Model-View-Controller komunikácia

1.7.1 MODEL

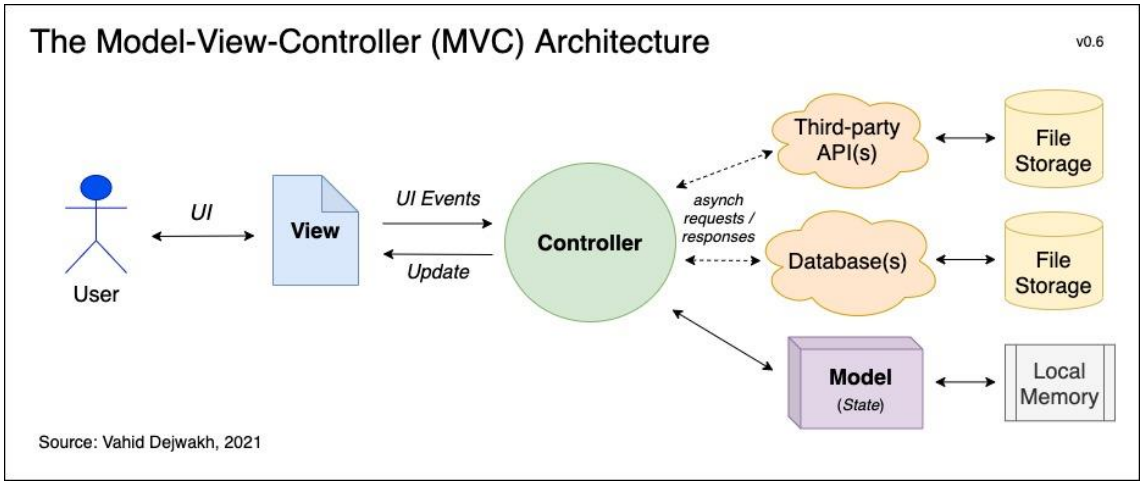
Model v MVC aplikácii reprezentuje spracované dáta a údaje, ktoré aplikácia potrebuje na spustenie, najmä ako in-memory, čiastočná alebo lokálna reprezentácia údajov, ktoré sa nachádzajú v databázach. Metódy, ktoré potrebuje aplikácia na vytvorenie, čítanie, aktualizácie alebo vymazanie (CRUD), by mali byť implementované v modeli, čím je umožnené aplikácii pristupovať k týmto údajom.

1.7.2 VIEW

View je časť aplikácie, ktorú používateľ vidí a interaguje s ňou. Pre front-end aplikácie je to DOM. V prípade rozhrania API alebo inej mikroslužby na strane servera by sme mohli považovať zobrazenie za výstup systému, t. j. odpoveď zo serveraView je zodpovedný za náhľad dát z modelu pre používatela. View vie ako pristúpiť k dátam, ale nevie načo dáta sú ani s nimi nevie manipulovať, view iba reprezentuje zobrazuje dané dáta pre používateľa.

1.7.3 CONTROLLER

Controller je mozgom aplikácie, kde je zabudovaná väčšina logiky aplikácie. Controller je spostredkovateľ, ktorý ťahá dáta z modelu a nasledne ich posielana na zobrazenie (View), aby sa vykreslili na stránke. V opačnom smere Controller prijíma udalosti z používateľského prostredia, spracováva ich a v prípade potreby odosiela údaje do modelu (pridanie údajov z používateľského vstupu do modelu).



Obr. 12: MVC komunikácia

OBR, texty - https://vahid.blog/post/2021-04-16-understanding-the-model-view-controller-mvc-pattern/

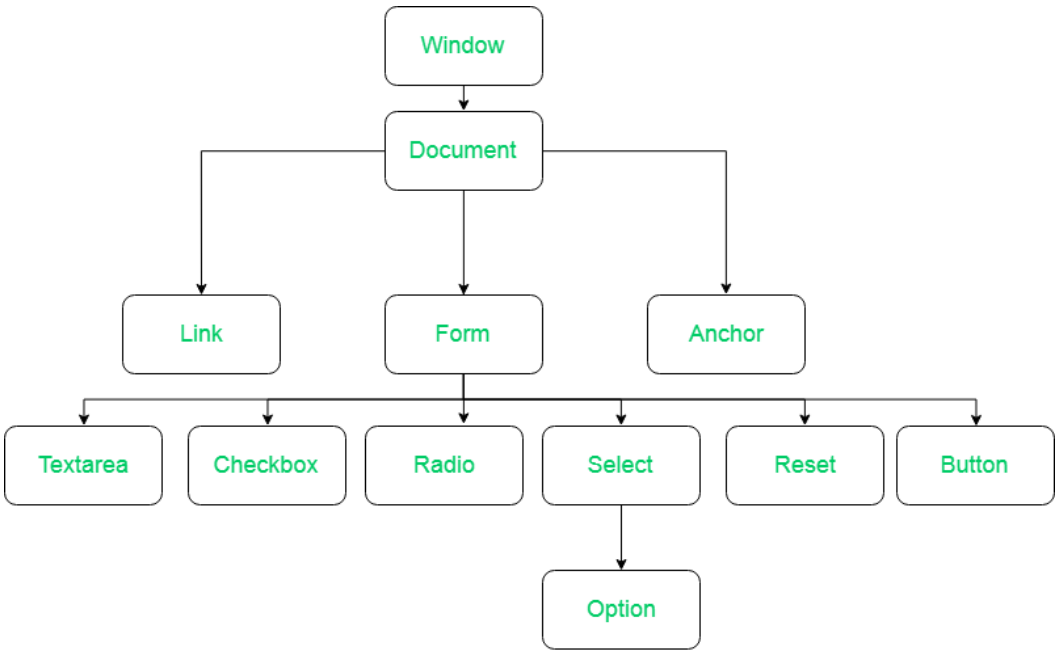
1.8 DOM vs. Virtual DOM

Zdroj - https://www.keitaro.com/insights/2023/07/12/dom-vs-virtual-dom-understanding-the-differences/

1.8.1 Document Object Model

Document Object Model alebo skrátene DOM je programovacie rozhranie pre webové dokumenty. DOM predstavuje štruktúru HTML webovej stránky, ku ktorej je možné pristupovať a manipulovať s ňou pomocou JavaScriptu. Keď sa webová stránka načíta prehliadač vytvorí DOM, ktorý pozostáva zo všetkých prvok HTML komponentov na stránke.

Pomocou JavaScriptu je možné DOM upravovať a aktualizovať webovú stránku. Pre príklad pokiaľ chce developer zmeniť textový obsah tlačidla, pomocou JavaScriptu vie prvok vybrať a následne pomocou DOM jeho textový obsah aktualizovať.



Obr. 13: Reprezentácia DOM

1.8.2 Virtual Document Object Model

Virtual DOM je kľúčový koncept pri využívaní fremeworkoch, ako je React.js, ktorý výrazne zlepšuje výkon aktualizácie skutočného DOM. Keď nastane zmena stavu v komponente, React namiesto priamej aktualizácie skutočného DOM vytvorí "ľahkú" repliku DOM ako Virtu DOM.

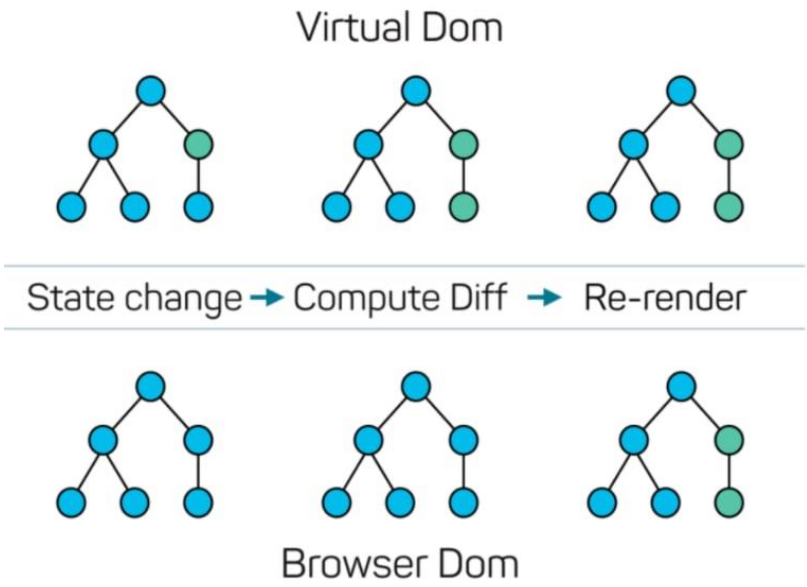
React potom porovná aktualizovaný Virtual DOM s predchádzajúcim, aby detekoval zmeny, ktoré nastali, ktoré je potrebné použiť na skutočný DOM. Akonáhle sú zmeny určené, React aktualizuje skutočný DOM optimalizovaným a efektívnym spôsobom bez zbytočného opätovného vykreslovania.

1.8.3 Rozdiely medzi DOM a Virtual DOM

Najzásadnejší rozdiel je, že DOM reprezentuje aktuálnu HTML štruktúru webovej stránky, zatiaľ čo Virtual DOM je "ľahká" replica DOM.

Tab. 1: Rozdiely medzi DOM a Virtual DOM

DOM	Virtual DOM
Reprezentuje webovú štruktúru HTML kódu.	Slúži ako zjednodušená reprezentácia DOM.
Je možná manipulácia so zobrazenými elementami.	Nie je možná manipulácia elementami zobrazeného na obrazovke.
Modifíkacia v DOM spôsobí aktualizáciu celého DOM stromu.	Modifikácia aktualizuje iba relevantný uzol v strome.
Aktualizácia stránky je pomalá a neefektívna.	Proces aktualizácie je rýchly a efektívny.



Obr. 14: Priebeh od zmeny po vykreslenie

OBR - https://dev.to/adityasharan01/react-virtual-dom-explained-in-simple-english-10j6

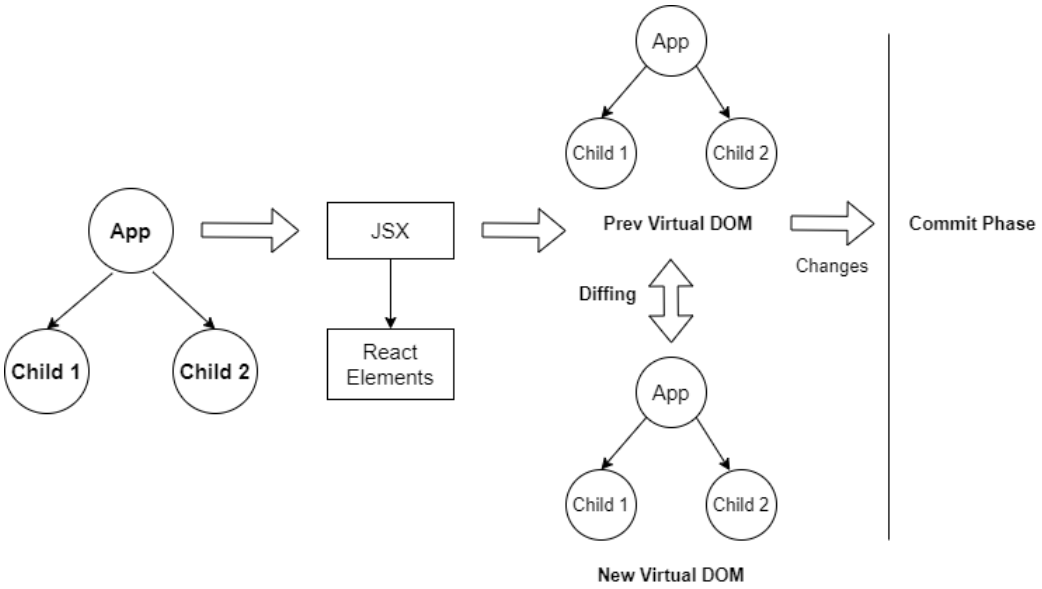
2 Použité technológie

2.9 React.js

React alebo React.js je open-source JavaScript knižnica, vytvorená spoločnosťou Facebook, ktorej cieľom je zjednodušiť proces vytvárania interaktívnych používateľských rozhraní. Používateľské rozhranie je vytvorené pomocou React komponentov, pričom každý zodpovedá za výstup časti HTML kódu, ktorý môže byť opätovane použitý.

Pri vyvíjaní frontendu aplikácie React používa znovu použitelné komponenty, ktoré môžu byť považované za nezávislé Lego bloky. Tieto komponenty sú jednotlivé časti finálneho UI, pričom pri použití viacerých tvoria celkovú frontend aplikáciu.

Hlavnou úlohou Reactu v aplikáciach je spracovať zobrazenie aplikácie ako View pri vzore MVC, tým že poskytuje najlepšie a najefektívnejšie prevedenia vykresľovania na web stránke. Namiesto toho aby sa celé používateľské rozhranie bralo ako jeden celok, React.js umožňuje vývojárom toto UI rozdeliť na viacero jednotlivých používateľských komponentov, ktoré formujú celé používateľské rozhranie. React.js kombinuje rychlosť a efektivitu JavaScriptu s efektívnou metódou manipulácie s DOM objektami na rýchlejšie vykreslovanie webových stránok a vytváranie vysoko dynamických a responzívnych webových aplikácii.



Obr. 15: Priebeh vykreslenia pri React.js

2.10 ASP .NET

Microsoft .NET je open-source cross-platformová iterácia .NET Frameworku. .NET môže byť použitý na vytvorenie rôzneho druhu aplikácia ako sú desktopové aplikácie, mobilné aplikácie, webové aplikácie a IoT zariadenia, Keďže .NET je open-source má výhodu veľkého počtu knižníc, jazykov a editorov. Programátor môže písať .NET aplikácie použitím C#, Visual Basic alebo F#.

2.11 BaSyx

Eclipse BaSyx je platforma s otvoreným zdrojom pre automatizáciu novej generácie. Eclipse BaSyx preto poskytuje bežné a opakovane použiteľné komponenty Industrie 4.0. Eclipse BaSyx umožňuje jednoduché vytváranie nových funkcií okolo oficiálneho HTTP REST rozhrania AAS. Medzi hlavné komponenty patri AAS ako hlavný pilier pre vývoj Industry 4.0 aplikácia.

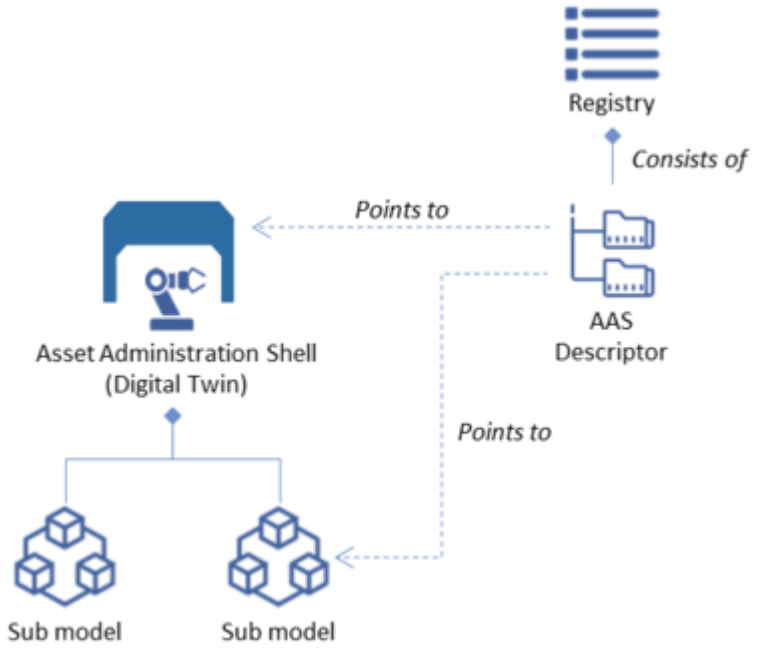
2.11.1 BaSyx AAS Server

AAS server komponent poskytuje prázdny AAS server ktorý môže byť použitý ako host pre viacero AAS alebo Submodelov. BaSyx AAS server je spusti pomocou docker-compose file. Pri spustení AAS servera bez konfigurácie bude vrátený prázdny JSON []. Na konfiguráciu AAS Server pri použití docker kontajnerov slúži aas.properties súbor, kde sú nakonfigurované nastavenia ako napríklad cesta k súboru, ktorý ma byť použitý ako zdroj pre AAS Server.

2.11.2 BaSyx AAS Registry

AAS Registry je centrálny komponent Asset administration shell (AAS) infraštruktúry, ktorý slúži na zobrazenie všetkých dostupných AAS serverov a ich submodelov pomocou jedinečných ID a ukladá dodatočné meta-data pre AAS. BaSyx AAS Registry registruje AAS deskriptory ktoré popisuje Asset Administration Shell a aj submodely AAS. Registry

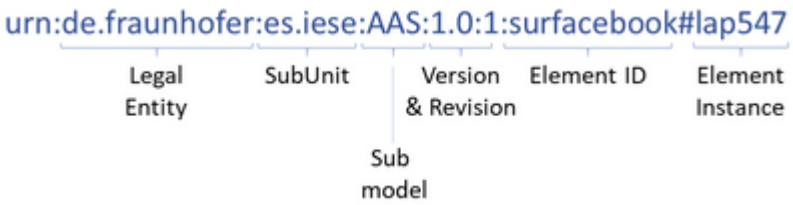
komponent je dostupný ako docker kontajner ako súčasť open-source BaSyx middleware.



Obr. 16: Registry komponent v Eclipse BaSyx

AAS musia byť registrované v Registry komponente aby sa zapezpečilo, že ich je možné nájsť podľa ich ID. Za jeho registráciu je zodpovedný komponent, ktorý pridáva nový AAS komponent.

Registry komponent v Eclipse BaSyx nemá stanovený pevný formát pre ID pre AAS alebo submodel, avšak je doležité aby každé ID bolo jedinečné.



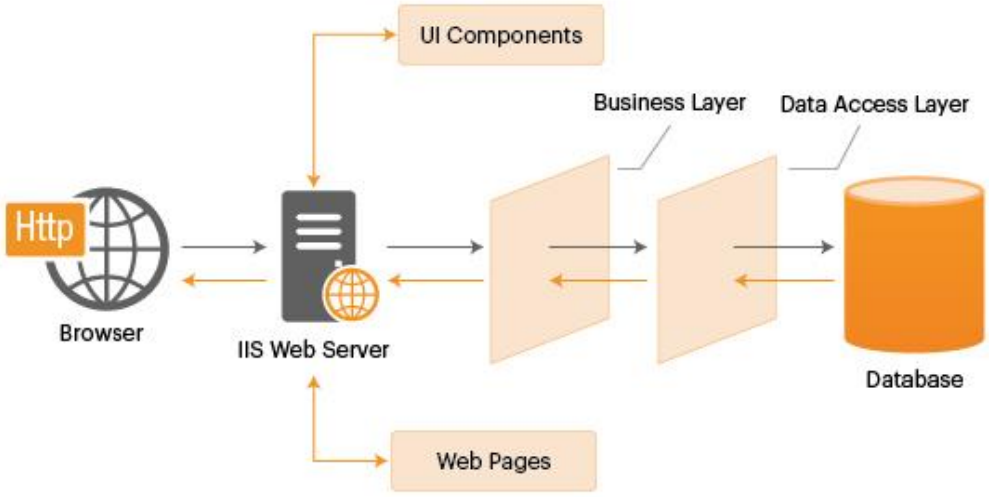
Obr. 17: Príklad pre ID pre AAS v Registry komponente

Eclipse BaSyx navrhuje formát a časti ako by malo ID vyzerať:

- Legal Entity – jedinečný identifikátor entity, ktorý využíva AAS
- SubUnit – odpovedá časti entity, napríklad divízii ktorá zodpovedá za aktívum
- SubModel – definuje submodel, ktorý je referovaný URN, ukazuje na AAS v tomto prípade alebo je to typ submodelu
- Version – definuje verziu AAS
- Revision – mala by byť inkrementovaná každou zmenou AAS alebo submodelu
- Element ID – definuje typ aktíva, ktorý je referovaný AAS alebo submodel
- Element Instance – indentifikuje konkrétne aktívum

2.12 IIS Web Server

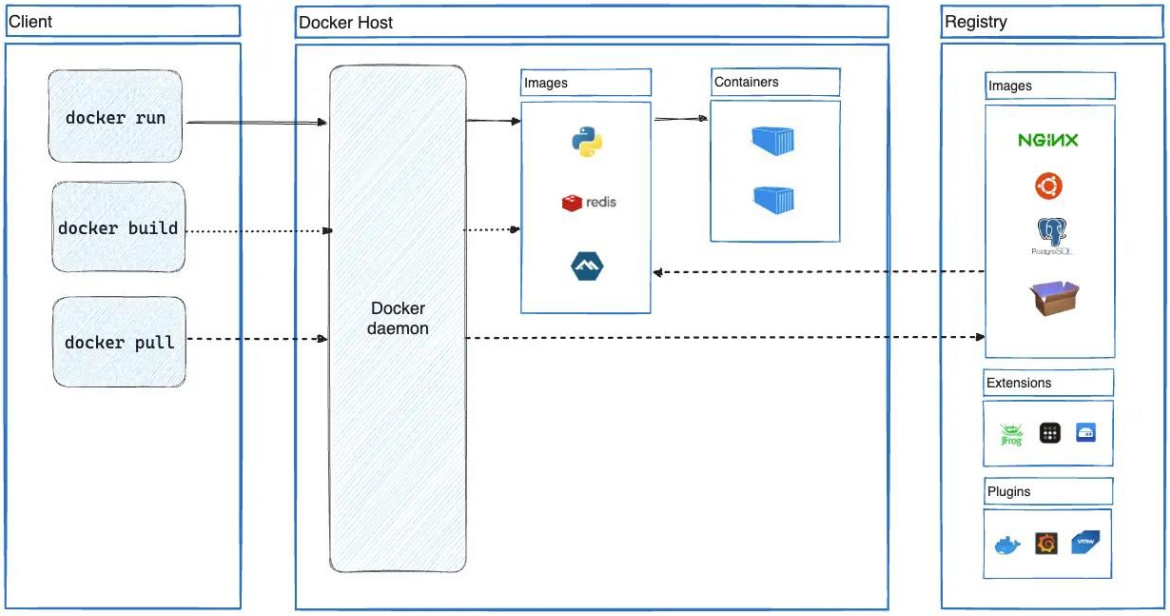
Internet Information Services, alebo inak IIS je webový server od spoločnosti Microsoft, ktorý beží na operačnom systéme Windows a je používaný na výmenu statického a dynamického obsahu s internetovými užívateľmi. IIS web server môže byť použitý na host, deploy alebo správu webových aplikácii, ktoré používajú technológie ako ASP.NET a PHP. IIS používa veľké množstvo protokolov na komunikáciu a dáta vymieňa s klientom alebo počítačom cez protokoly ako sú http, SMTP alebo FTP. Ako hlavný produkt od spoločnosti Windows IIS prišiel s integráciou Windows serveru a beží na Windows OS. IIS web server môže byť použitý aj cez Linux alebo macOS, ale často je málo stabilný a má chabý výkon.



Obr. 18

2.13 Docker

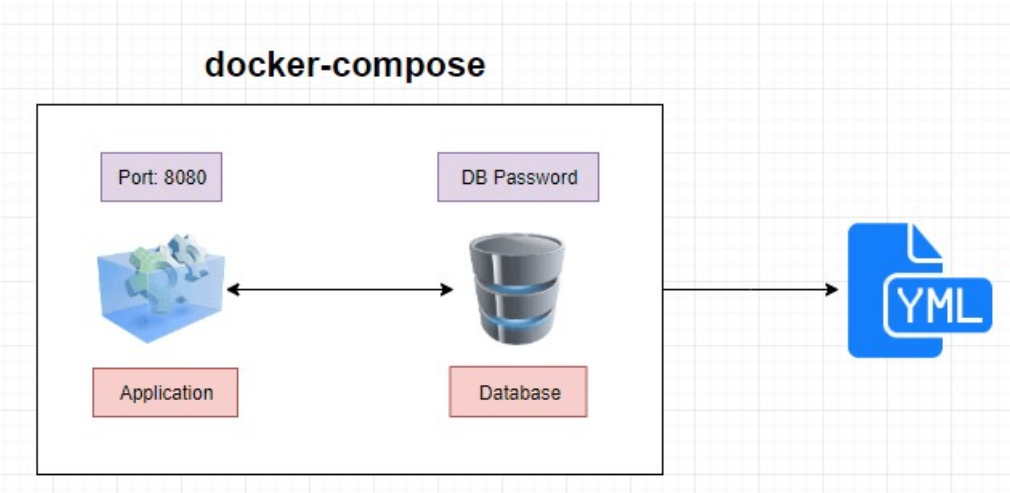
Docker je open-source platforma pre vývoj a spúšťanie aplikácii. Docker vám umožňuje oddeliť aplikácie od štruktúry aby ste mohli rýchlejšie dodávať softvér. Docker poskytuje možnosť zabaliť aplikáciu a spustiť ju v izolovanom prostredí nazývanom Docker Container. Izolácia a bezpečnosť vám umožnia súčasne spúšťať viacero aplikácii cez kontajnery na danom hostiteľovi. Kontajnery nie sú veľkostne náročné a obsahujú všetko čo je potrebné pre spustenie aplikácie, takže sa používateľ nemusí spoliehať na to čo je nainštalovane na hosťujúcom zariadení. Docker klient komunikuje cez Docker daemon, ktorý zabezpečuje build a následne beh Docker kontajneru. Medzi najzákladnejšie prvky Dockeru patrí Docker Image a Docker Container. Docker Image je read-only template (predloha/recept) s inštrukciami pre vytvorenie Docker kontajneru. Docker kontajner je bežiaca inštancia Docker image-u. Jednotlivé kontajnery sú izolované od iných a takisto aj od hosťujúceho zariadenia avšak dokážu aj medzi sebou komunikovať.



Obr. 19: Docker

2.13.1 Docker compose

Docker compose je nástroj, ktorý nám umožňuje definovať a zároveň spustiť viacero Rocker kontajnerov naraz z jedného súboru. Docker-compose využíva YAML súbor na konfiguráciu jednotlivých aplikácii bežiacich v docker kontajneroch.



Obr. 20

2.14 MS SQL

3 Návrh aplikácie

V danej časti je zobrazený návrh a štruktúra aplikácie, ktorá sa odvíja z teoretickej časti a použitých technológií. V danej téme ide o vytvorenie web aplikácie, ktorá pozostáva z klientskej a serverovej časti. Pri návrhu aplikácie je potrebná špecifikácia požiadaviek, aby boli splnené všetky funkcionality, ktoré musí aplikácia poskytovať. Pri klientskej časti je dôležité vytvoriť priateľské užívateľské rozhranie, ktoré nebude náročné na obsluhu, pri vývoji serverovej časti treba vytvoriť API komunikáciu a následne nasadiť aplikáciu v IIS.

3.15 Špecifikácia požiadaviek

Špecifikácia požiadaviek je určená na presné definovanie a opísanie funkcionálnych a nefunkciaonálnych požiadaviek, ktoré budú následne implementované v aplikácií. Cieľom špecifikácie požiadaviek je poskytnutie požiadaviek, ktoré budú jednoznačne opisovať cieľové správanie aplikácie a poskytovať potreby a očakávania používateľa.

3.15.1 Funkcionálne požiadavky

- Aplikácia umožní používateľovi nahrať súbor s AAS serverom za behu aplikácie
- Aplikácia poskytuje používateľovi zoznam AAS serverov
- Používateľ má prístup k detailným údajom AAS servera a takisto aj prístup k jednolivým submodelom
- Systém využíva stromovú štrukutúru na prehladné zobrazenie údajov ohľadom jednotlivých serverov
- Aplikácia podporuje perzistenciu údajov ohľadom AAS serverov
- Aplikácia umožňuje používateľovi vymazať AAS server z databázy
- Pokiaľ AAS server obsahuje stiahnuteľný súbor s AAS serverom, aplikácia umožňuje používateľovi stiahnúť tento súbor, ktorý sa nachádza v submodely
- V zozname serverov je umožnená funkcionalita pre vyhľadávanie konkrétneho servera podľa názvu pomocou vyhľadávacieho poľa
- V zozname serverov je možnosť prepínania medzi zobrazením AAS serverov a OPC UA serverov

- Pri zobrazení OPC UA serverov je možné aplikovať filter pre zobrazenie všetkých/online/offline serverov
- Aplikácia podporuje perzistenciu OPC UA serverov
- Používateľ môže z zoznamu jednotlivé OPC UA servery spustiť, pozastaviť alebo vymazať
- Aplikácia pri stiahnutí OPC UA servera zabezpečí rozipovanie .zip súboru a obsah súbor uloží do ankonfigurovaného priečinka kde bude server uložený a pridá server do databázy dostupných OPC UA server, pokiaľ je už daný subor v databáze prebehne iba stiahnutie súboru bez rozipovania
- Aplikácia umožnuje prístup k informáciam o jednotlivých OPC UA serveroch, ktoré používateľovi zobrazí základné informácie
- Aplikácia umožňuje spustenie OPC UA serverov za behu aplikácie a ich manažovanie

3.15.2 Nefunkcionálne požiadavky

3.15.3 Architektúra aplikácie

3.15.4 Štruktúra dátového modelu

Štruktúra dátového modelu je hlavným prvkom pre organizáciu a správu dát v systéme. Poskytuje jednoznačné definovanie jednotlívých dátových modelov pre správne fungovanie systému. Pri definovaní dátového modelu vo webovej aplikácií sme zakladali na jednoduchosti. Pre správu OPC UA serverov sme použili databázu MySQL kde sme definovali jeden dátový model pre OPC UA server.

3.15.5 Grafické rozhranie

3.15.6 REST API

Členenie jadra je spravidla nasledovné:

a) Analýza problému

b) Opis riešenia

c) Zhodnotenie

3.16 Analýza problému – Súčasný stav riešenej problematiky

V časti Analýza problému autor uvádza súčasný stav riešenej problematiky doma i v zahraničí, dostupné informácie a poznatky týkajúce sa danej témy. Zdrojom pre spracovanie sú aktuálne publikované práce domácich a zahraničných autorov. Základné definície a formalizmy potrebné na riešenie problematiky.

3.17 Opis riešenia

Časť Opis riešenia jasne, výstižne a presne charakterizuje predmet riešenia. Súčasťou sú aj rozpracované čiastkové ciele, ktoré podmieňujú dosiahnutie hlavného cieľa. Ak je práca implementačná, tak jej súčasťou musí byť aj softvérová špecifikácia požiadaviek, návrh, implementácia, overenie riešenia. Treba podľa možností vychádzať zo známych prístupov. Táto časť práce závisí od konkrétneho zadania. Je dôležité prezentovať návrhové rozhodnutia, alternatívy, ktoré sa zvažovali pri riešení a samotný návrh riešenia zadaného problému. Štruktúra textu by mala vychádzať zo zadanej úlohy, ktorá sa rieši. Najmä v tejto časti študent preukazuje svoj originálny prístup k riešeniu problémov a kritické myslenie.

Súčasťou môže byť metodika práce a metódy skúmania, ktoré spravidla obsahujú:

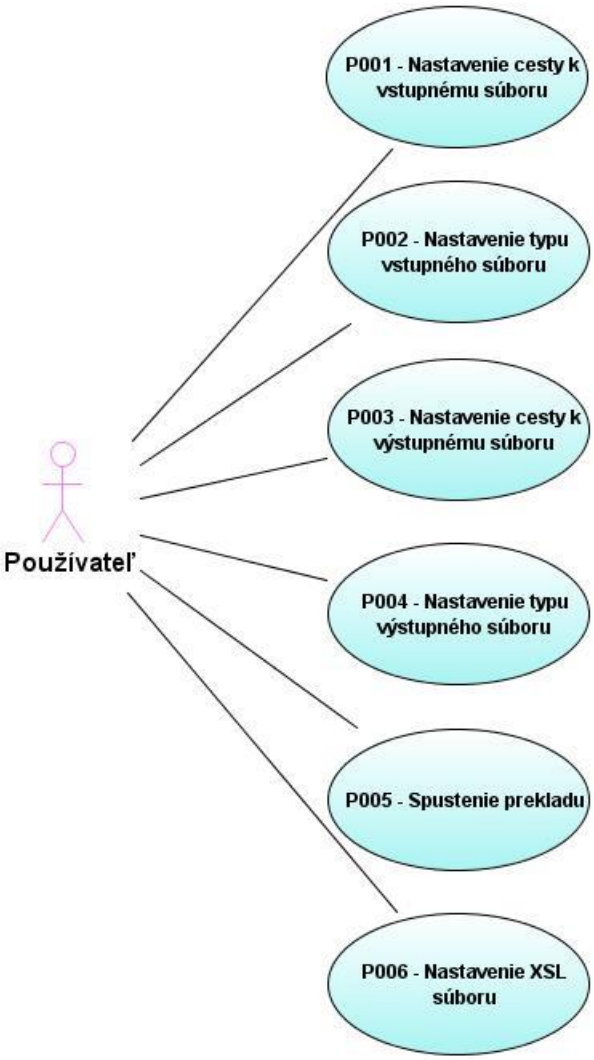
a) charakteristiku objektu skúmania

b) pracovné postupy

c) spôsob získavania údajov a ich zdroje

d) použité metódy ich vyhodnotenia a interpretácie výsledkov

Implementácia musí byť otestovaná. Výsledok musí byť porovnaný s inými riešeniami.



Obrázok 1-1 Use case diagram požiadaviek

3.18 Zhodnotenie

Výsledky (vlastné postoje alebo vlastné riešenie vecných problémov), ku ktorým autor dospel, sa musia logicky usporiadať a pri popisovaní sa musia dostatočne zhodnotiť. Zároveň sa komentujú všetky skutočnosti a poznatky v konfrontácii s výsledkami iných autorov. Ak je to vhodné, výsledky práce a diskusia môžu tvoriť samostatné časti ZP.

3.19 Citácie

Citácia alebo citovanie je skrátené označenie citovaného zdroja (dokumentu) v texte práce podľa niektorej metódy odporučenej normou STN ISO 690. Citácia slúži na spojenie citovaného miesta so záznamom o citovanom dokumente. Tento záznam je potom položkou v zozname bibliografických odkazov. Citácia slúži aj na spojenie citovaného miesta so skráteným záznamom o citovanom dokumente umiestneným napr. pod čiarou na príslušnej strane textu. VZOR CITÁCIE (1)

3.19.1 Postup vkladania citácie

Na karte REFERENCES si zvolíme štýl citácie ISO-690- Numerical references. Následne klikneme na Manage sources a kliknutím na new otvoríme okno na vloženie nového bibliografického odkazu. Z rolovacieho menu vľavo hore vyberieme typ dokumentu, ktorý budeme citovať a vyplníme všetky známe údaje o zdroji citácie. Potvrdíme a vrátime sa späť na písanie dokumentu. Na karte references klikneme na Insert Citation a vyberieme citovaný zdroj. V dokumente sa objaví číselný odkaz. VZOR CITÁCIE (1)

3.20 Špeciálne požiadavky

Ak je súčasťou vašej práce vytvorenie softvéru je nutné k tomuto softvéru vytvoriť dokumentáciu (technickú dokumentáciu, užívateľskú príručku) a pripojiť ju ku práci vo forme prílohy. Ak je dokumentácia rozsiahla, je vhodnejšie ju pridať ako prílohu na CD/DVD. Ak je kratšia, tak je vhodné ju pridať aj v tlačenej forme.

4 Popis šablóny

V šablóne sú použité viaceré druhy polí. Pevné polia nedovoľujú prepisovať ich obsah. Naopak polia, ktorých text je vyznačený červeným písmom musí byť zmenený, alebo vymazaný. V šablóne sa nachádzajú selektívne polia, ktoré umožňujú výber z viacerých variant. S poľami sa dá pracovať na karte vývojár, ktorú je možné vložiť v nastaveniach.

4.1 Popis nastavenia strany

OKRAJE: hore 3cm, dole 3cm, vľavo 3cm, vpravo 2,5cm, orientácia: na výšku
PAPIER: typ: A4, šírka: 21cm, výška:29,7cm,
ROZLOŽENIE: hlavička: 1,5cm, päta: 1,5cm, zvislé zarovnanie: hore

4.2 Popis nastavenia štýlov

NADPIS 1.ÚROVNE: založiť na: žiadnom, štýl nasledujúceho odseku: základný, Písmo: Times New Roman, 22 b, Tučné, Vľavo, Riadkovanie: jednoduché, Medzera Za: 16b, Kontrola osamotených riadkov, Zlom strany pred odsekom, Zviazať s nasledujúcim, Zviazať riadky dohromady, Viacúrovňové + Úroveň: 1 + Štýl číslovania: 1, 2, 3, … + Číslovať od: 1 + Zarovnanie: Vľavo + Zarovnať na: 0 cm + Zarážka: 1,27 cm, Štýl: Prepojené, Automaticky aktualizovať, Zobraziť v galérii štýlov

NADPIS 2.ÚROVNE: založiť na: žiadnom, štýl nasledujúceho odseku: základný, Písmo: Times New Roman, 16 b, Tučné, Zarážka: Vľavo: 0 cm, Opakovaná zarážka: 0,6 cm, Vľavo, Riadkovanie: jednoduché, Medzera Pred: 16 b, Za: 14 b, Kontrola osamotených riadkov, Zviazať s nasledujúcim, Zviazať riadky dohromady, Viacúrovňové + Úroveň: 2 + Štýl číslovania: 1, 2, 3, … + Číslovať od: 1 + Zarovnanie: Vľavo + Zarovnať na: 0 cm + Zarážka: 0,6 cm, Štýl: Prepojené, Automaticky aktualizovať, Zobraziť v galérii štýlov

NADPIS 3.ÚROVNE: založiť na: žiadnom, štýl nasledujúceho odseku: základný, Písmo: Times New Roman, 14 b, Tučné, Zarážka: Vľavo: 0 cm Opakovaná zarážka: 0,6 cm, Vľavo, Riadkovanie: jednoduché, Medzera Pred: 14 b Za:14 b, Kontrola osamotených riadkov, Zviazať s nasledujúcim, Zviazať riadky dohromady, Viacúrovňové + Úroveň: 3 +

Štýl číslovania: 1, 2, 3, … + Číslovať od: 1 + Zarovnanie: Vľavo + Zarovnať na: 0 cm + Zarážka: 0,6 cm, Štýl: Prepojené, Zobraziť v galérii štýlov

ZÁKLADNÝ: založiť na: žiadnom, štýl nasledujúceho odseku: základný, Písmo: Times New Roman, 12 b, Zarážka: Prvý riadok: 0,8 cm, Podľa okraja, Riadkovanie: 1,5 riadka, Kontrola osamotených riadkov, Štýl: Prepojené, Automaticky aktualizovať, Zobraziť v galérii štýlov

POPIS: Písmo: 9 b, Kurzíva, Farba písma: Text, Riadkovanie: jednoduché, Medzera

Za: 10 b, Štýl: Skryť, kým nie je použité, Zobraziť v galérii štýlov, Priorita: 36, Podľa: Normálny

Použitie: na popis obrázkov, tabuliek a grafov

NADPIS NEČÍSLOVANÝ: Bez odrážok a číslovania, Štýl: Prepojené, Zobraziť v galérii štýlov Podľa: Nadpis 1.úrovne

Záver

V závere je potrebné v stručnosti zhrnúť dosiahnuté výsledky vo vzťahu k stanoveným cieľom.

Zoznam použitej literatúry

1. **Prata, Stephen.** *Mistrovství v C++*. [prekl.] Vozák David, Beroun Libor, Dokoupil Petr, Ptáček Lubomír Sokol Boris. 3. Praha : Computer Press, 2007. s. 1119. ISBN: 8025117491.

Prílohy

Prílohy sú „číslované“ písmenami A, B, C...

Príloha A: Štruktúra elektronického nosiča

Štruktúra elektronického nosiča (CD, DVD, atď.) s kompletnou digitálnou verziou tlačenej formy práce, vrátane príloh, funkčných zdrojových kódov, programov (aplikácií) pripravených na inštalovanie a iných, vo všeobecnosti ťažko opísateľných ale potrebných častí. Elektronický nosič musí mať obal, pomocou ktorého sa pevne pripevní do práce. Nosič musí mať popis obsahu a meno autora.