VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA PODNIKATELSKÁ
FACULTY OF BUSINESS AND MANAGEMENT

ÚSTAV INFORMATIKY
INSTITUTE OF INFORMATICS

NÁVRH A IMPLEMENTACE INFORMAČNÍHO SYSTÉMU PRO DOHLED

DESIGN AND IMPLEMENTATION OF THE SURVEILLANCE INFORMATION SYSTEM

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Marek Čelka**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Jan Luhan, Ph.D., MSc**
SUPERVISOR

BRNO 2020

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA PODNIKATELSKÁ

Zadání diplomové práce

Ústav:	Ústav informatiky
Student:	**Bc. Marek Čelka**
Studijní program:	Systémové inženýrství a informatika
Studijní obor:	Informační management
Vedoucí práce:	**Ing. Jan Luhan, Ph.D., MSc**
Akademický rok:	2019/20

Ředitel ústavu Vám v souladu se zákonem č. 111/1998 Sb., o vysokých školách ve znění pozdějších předpisů a se Studijním a zkušebním řádem VUT v Brně zadává diplomovou práci s názvem:

Návrh a implementace informačního systému pro dohled

Charakteristika problematiky úkolu:

Úvod
Cíle práce, metody a postupy zpracování
Teoretická východiska práce
Analýza současného stavu
Vlastní návrhy řešení
Závěr
Seznam použité literatury
Přílohy

Cíle, kterých má být dosaženo:

Navrhnout a implementovat informační systém pro servisní dohled. Výstupem práce bude sofwarový komplet pro automatický servisní dohled pro vybranou společnost.

Základní literární prameny:

BRUCKNER, T., J. VOŘÍŠEK, A. BUCHALCEVOVÁ a kol. Tvorba informačních systémů: Principy, metodiky, architektury. Praha: Grada Publishing, 2012. 360 s. ISBN 978-80-247-4153-6.

DELANEY, K., B. BEAUCHEMIN, C. CUNNINGHAM, J. KEHAYIAS, P. S. RANDAL and B. NEVAREZ. Microsoft SQL Server 2012 Internals (Developer Reference). 1st ed. Sebastopol: Microsoft Press, 2013. 982 p. ISBN 978-0-7356-5856-1.

GRASSEOVÁ, M., R. DUBEC a D. ŘEHÁK. 33 nejpoužívanějších metod strategického řízení. 1. vyd. Brno: Computer Press, 2010. 325 s. ISBN 987-80-251-2621-9.

MOLNÁR, Z. Efektivnost informačních systémů. 1. vyd. Praha: Grada Publishing, 2000. 142 s. ISBN 80-7169-410-X.

SCHWALBE, K. Řízení projektů v IT : Kompletní průvodce. Praha: Computer Press, 2011. 632 s. ISBN 978-80-251-2882-4.

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2019/20

V Brně dne 29.2.2020

L. S.

doc. RNDr. Bedřich Půža, CSc. ředitel	doc. Ing. et Ing. Stanislav Škapa, Ph.D. děkan

Abstrakt

Cieľom práce je návrh a implementácia informačného systému pre servisný dohľad, ktorý bude schopný automatizovane monitorovať požadované zariadenia vybranej spoločnosti. Tomuto cieľu predchádzala strategická analýza pomocou metód PEST a 7S pre zistenie aktuálneho stavu danej spoločnosti. Ďalšou časťou bola analýza efektívnosti servisných procesov pomocou metódy HOS spolu s výberom vhodného dohľadového systému. Zmena servisných procesov bola navrhnutá Lewinovím modelom. Nakoniec bol navrhnutý, implementovaný a otestovaný vybraný informačný systém pre dohľad. Nasadením tohoto systému sa očakáva zjednodušenie dohľadu zariadení pre dispečerov, ušetrenie nákladov a rovnako aj zvýšenie rýchlosti reagovania na vzniknuté situácie.

Abstract

The main goal of the master thesis is design and implementation of information system for monitoring purposes. The implemented monitoring system will be able to automatically monitor devices of a certain company. This goal is preceded by strategic analysis using PEST and 7S methods for examination of current state of the company. Another part is the analysis of the efficiency of service processes using the HOS method together with the selection of a suitable monitoring system. An enhancement of service processes is designed by the Lewin model. Finally, the selected information system for monitoring was designed, implemented and tested. A deployment of the system should provide simplification of devices supervision, to save costs as well as to speed-up the response-time to unexpected situations.

Kľúčové slová

Dohľadové systémy, Informačné systémy, Informačné technológie, Python, Zabbix, Nagios, Datadog, Strategická analýza, McKinsey 7S model, PEST analýza, Lewinov model

Keywords

Monitoring systems, Information systems, Information technologies, Python, Zabbix, Nagios, Datadog, Strategic analysis, Mckinsey 7s model, PEST analysis, Lewin's model

Citácia

ČELKA, Marek. *Návrh a implementace informačního systému pro dohled.* Brno, 2020. Diplomová práce. Vysoké učení technické v Brně, Fakulta podnikatelská. Vedoucí práce Ing. Jan Luhan, Ph.D., MSc

Návrh a implementace informačního systému pro dohled

Prehlásenie

Prehlasujem, že predložená diplomová práca je pôvodná a spracoval som ju samostatne. Prehlasujem, že citácia použitých prameňov je úplná, že som vo svojej práci neporušil autorské práva (v zmysle Zákona č. 121/2000 Sb., o práve autorskom a o právach súvisiacich s právom autorským).

V Brne dňa 15. mája 2020

.....

Marek Čelka

Poďakovanie

Pri tejto príležitosti by som sa chcel poďakovať vedúcemu mojej práce Ing. Jánovi Luhanovi, Ph.D., MSc za trpezlivosť, cenné rady a v neposlednom rade za čas, ktorý mi venoval. Ďalej by som sa chcel poďakovať všetkým, bez ktorých pomoci by nebolo možné vypracovať túto prácu.

Obsah

Kapitola 1

Úvod

Závislosť dnešných spoločností od informačných technológií stále narastá a tak vzrastá aj závislosť na bezpečnej a spoľahlivej prevádzke jednotlivých zariadení tvoriacich informačný systém. Napríklad taký výpadok produkčného systému riadiaceho výrobu znamená aj zastavenie výroby a tým nemalé ekonomické následky.

Cieľom tejto práce je návrh a implementácia informačného systému pre servisný dohľad, ktorý bude schopný automatizovane monitorovať požadované zariadenia vybranej spoločnosti. Nasadením tohoto dohľadového systému sa očakáva zjednodušenie dohľadu zariadení pre dispečerov, ušetrenie nákladov a rovnako aj zvýšenie rýchlosti reagovania na vzniknuté situácie.

V rámci práce bude navrhnutý a implementovaný softwarový komplet, ktorý bude možné použiť pre automatizované dohliadanie požadovaných zariadení. Pri takto zásadnej zmene v stratégií servisných procesov je potrebné najprv vykonať strategickú analýzu firmy a navrhnúť postup implementácie tejto zmeny nevynímajúc riadenie rizík s ňou spojených. Preto ako prvá prebehne analýza vonkajších a vnútorných faktorov ovplyvňujúcich danú firmu. Následne bude navrhnutý postup implementácie zmeny servisných procesov do prostredia firmy. Dôležitou časťou bude identifikovanie možných rizík a navrhnutie opatrení na minimalizáciu ich vplyvu.

Posledná časť práce je zameraná na návrh a implementáciu vybraného informačného systému pre automatizovaný dohľad. Táto časť pozostáva z návrhu grafického rozhrania pre používateľov, návrhu a implementácie jednotlivých konektorov pre monitorované zariadenia a vyhodnotenia testovacej prevádzky systému. V rámci vyhodnotenia budú vyčíslené náklady spojené s nasadením systému ako aj odporúčania k budúcej prevádzke.

Kapitola 2

Cieľ a metodika práce

V súčasnej dobe je všeobecný trend automatizácie rôznych procesov vo firmách, dalo by sa povedať, že firma je taká rýchla ako jej najpomalší zamestnanec. Za automatizáciu môžeme považovať delegovanie opakujúcich sa úloh z človeka na stroj alebo program. Pri snahe o automatizovanie určitej úlohy je potrebné si uvedomiť, či je automatizácia možná a či sa vyplatí. Dôležité je analyzovať, či daná úloha vyžaduje istú kreativitu, koľko času zaberie zamestnancovi a ako často ju opakuje. Posledným krokom je porovnanie ceny za automatizáciu a jej ekonomický prínos pre firmu.

Cieľom tejto práce je návrh a implementácia *automatizovaného dohľadového systému* (ADS). Tento systém má slúžiť na centrálne dohliadanie vybraných zariadení spoločnosti. Systém sa bude skladať z niekoľkých preprocesorov dát, ktoré sú produkované ako výstup alebo stavové informácie sledovaných zariadení. Tieto preprocesory budú spojené do jedného prehľadného celku, ktorý umožní dispečerom dohľad nad všetkými potrebnými informáciami, na všetkých dohliadaných zariadeniach na jednom mieste.

Metodika práce je systematická, začína spracovaním teoretických východísk nasledovaných analýzou, návrhom vlastného riešenia, implementáciou a nasadením do praxe. Neoddeliteľnou súčasťou je taktiež pomocou vhodných nástrojov identifikovať, zhodnotiť a redukovať riziká spojené so zavádzaním nového systému. K analýze využíva rôzne analytické nástroje a formálne modely popísané v ďalšej časti, ktoré sú bežne využívané.

V prvej časti práce bude zhrnutá teória potrebná k porozumeniu danej problematike práce, kde budú vysvetlené základy informačných systémov, ich údržba a životný cyklus. Ďalšia časť práce sa bude venovať analýze stavu vybranej firmy AŽD Praha s.r.o. pomocou analýz vonkajšieho politického, ekonomického, sociálneho a technologického prostredia

firmy (metóda PEST) a vnútorných faktorov firmy pomocou McKinseyho modelu 7S. Efektívnosť celkového servisného systému bude analyzovaná metódou HOS. Taktiež budú analyzované možné riešenia dohľadových informačných systémov používaných na automatizovaný dohľad požadovaných zariadení. V poslednej časti práce prebehne návrh a implementácia najvhodnejšieho dohľadového systému pre danú firmu. Ako posledná bude vytvorená analýza dopadu používania systému pre technikov a jej ekonomický vplyv na danú firmu.

Kapitola 3

Teoretické východiská práce

Informačné systémy (IS) sú v dnešnej dobe neoddeliteľnou súčasťou nášho každodenného života. Informačné technológie patria v súčasnosti medzi najrýchlejšie sa vyvíjajúce odvetvia, preto aj každá firma, ktorá chce byť konkurencie schopná je nútená inovovať a nasadzovať nové informačné systémy. IS ovplyvňujú spôsob práce na rôznych úrovniach firmy, kde je potrebné pracovať s rôznymi typmi informácií, či už ide o vybavovanie objednávok, riešenie mzdových záležitostí alebo monitorovania zariadení a informácií, ktoré zhromažďujú.

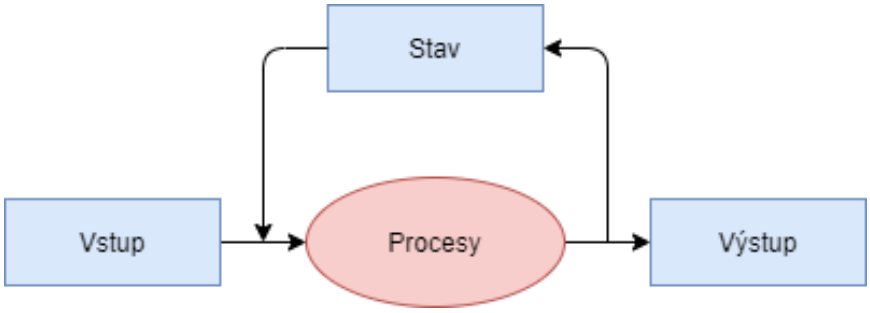
Rôzne IS sú aktuálne vo firme AŽD Praha s.r.o. dobre zavedenou praxou. U zvolenej firmy sa osvedčili v mnohých oblastiach firemnej agendy, od evidencie dochádzky cez IS pre zdieľanie aktuálnych interných informácií a smerníc až po sofistikované systémy pre podporu výrobných procesov. IS preto predstavujú už teraz podstatnú časť podporných prostriedkov riadenia firmy. Cieľom tejto práce je rozšírenie týchto systémov o automatizovaný dohľad. Z tohoto dôvodu bude nasledujúca časť zameraná na teoretický popis IS.

3.1 Informačné systémy

IS je súbor ľudí, metód a technických prostriedkov, ktoré zaisťujú zber, prenos, uchovávanie a spracovanie dát za účelom tvorby a prezentácie informácií pre potreby užívateľov činných v systéme riadenia. [5]

Jednotlivé časti IS sú nasledovné:

- informačné technológie - software spoločne s hardware zabezpečujú požadovanú funkčnosť, prenos, udržiavanie, spracovanie a poskytovanie dát. Na nasledujúcom obrázku je zobrazená obecná schéma IS:



Obr. 3.1: Schéma IS. Dáta uchovávajú **stav** systému a **procesy** realizujú transformácie často vo forme transakcií (sekcia 3.2.1). Zdroj: Vlastná tvorba.

- ľudia - Používatelia IS. Interpretujú dáta a používajú informácie na riadenie, rozhodovanie a sú sami nositeľmi ďalších mnohých informácií.

Základné rozdelenie IS:

- podnikové - IS, ktoré prevádzkujú podniky sami pre seba s takými dátami, ku ktorým majú prístup len akreditovaní zamestnanci napr. kvôli zachovaniu bezpečnosti podniku
- verejné - IS, ktoré pracujú s dátami dostupnými širokej verejnosti, ktoré majú väčšinou len informatívny charakter

Pre správne fungovanie podniku je potrebné efektívne zaobchádzanie s jej zdrojmi a tým zabezpečiť čo najvyššiu produktivitu. Keďže neoddeliteľnou súčasťou podniku sú rôzne IS, je potrebné sledovať efektívnosť aj týchto systémov. Problematiku posudzovania efektívnosti IS môžeme považovať za jeden zo základných pilierov pri výbere správneho informačného systému. Preto v nasledujúcej časti bude popísaná metóda slúžiaca na posudzovanie efektívnosti IS.

3.1.1 Posudzovanie efektívnosti IS metódou HOS

Každému podniku ide o dosahovanie vytýčených strategických cieľov, ktorých dosiahnutie je bez dobre fungujúcich informačných systémov veľmi obtiažne. Efektívnosť môžeme definovať ako stupeň, s ktorým IS poskytuje správne výstupy na správnom mieste, v správnom čase a hlavne za správnu cenu. Molnár v zdroji [21] navrhuje pre jej hodnotenie použiť vzťah 3.1, kde C_d značí stupeň dosiahnutého cieľa a N_d výdaje na dosiahnutie plánovaného cieľa.

$$E = \frac{C_d}{N_d} \tag{3.1}$$

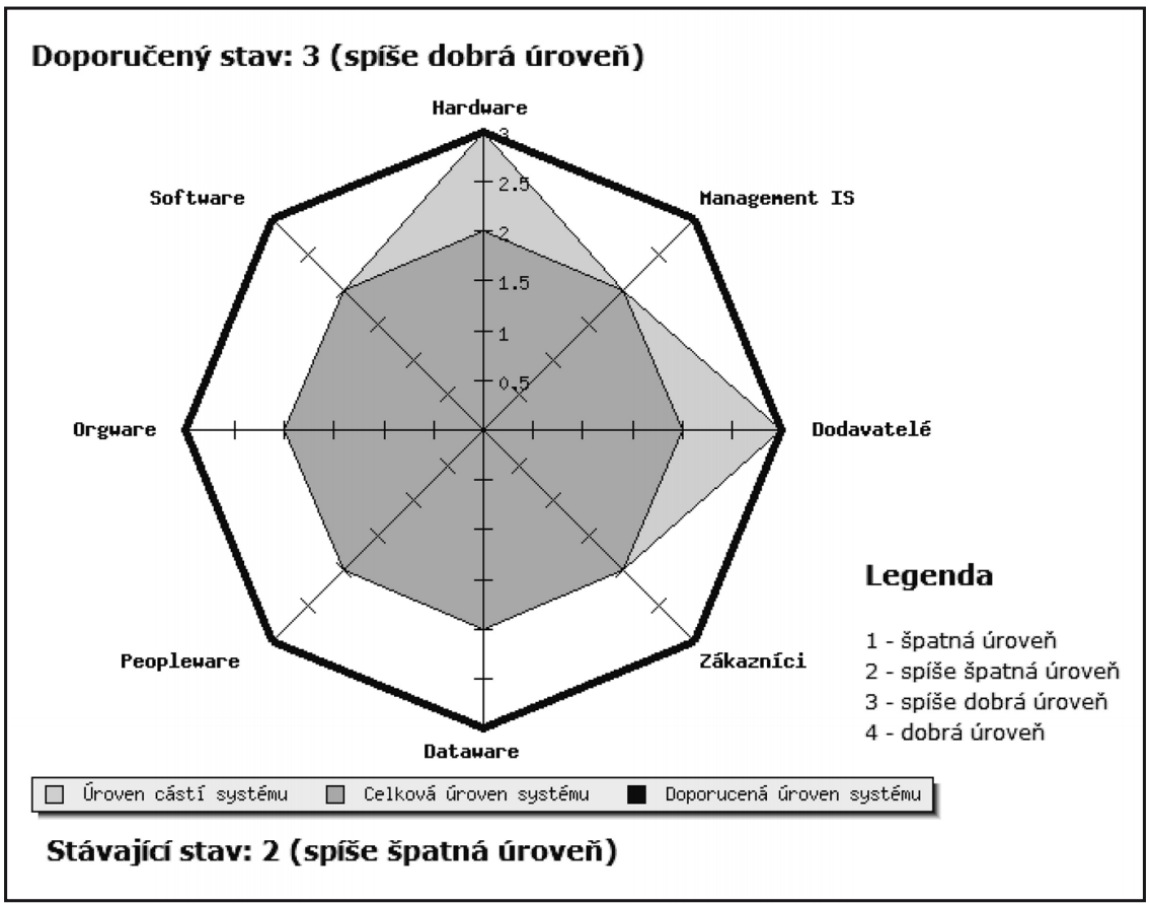
Základnou filozofiou metódy HOS je ohodnotenie IS na základe úrovne ohodnotenia jeho jednotlivých častí, z ktorých sa skladá. Cieľom metódy je nájdenie najhorších zložiek, ktoré negatívne ovplyvňujú IS a posúdenie jednotlivých častí systému, či sú na rovnakej alebo aspoň podobnej úrovni, pretože nevyváženosť vedie k neefektívnosti celého systému.

Na základe dlhodobého výskumu vplyvu jednotlivých častí IS bolo vybraných osem základných častí IS [21]:

- hardware - technické vybavenie firmy
- software - programové vybavenie firmy
- orgware - zahrňuje pravidlá pre prevádzku IS, pracovné postupy a bezpečnostné vo firme
- peopleware - používatelia IS
- dataware - dáta vo vzťahu k ich dostupnosti, správe a bezpečnosti
- zákazníci - zákazníci IS
- dodávatelia - pracovníci zaisťujúci prevádzku IS
- management - riadenie IS vo vzťahu k informačnej stratégií firmy

Pre hodnotenie úrovne jednotlivých častí IS sa používa nasledujúca štvorbodová škála: 1 - zlé, 2 - skôr zlé, 3 - skôr dobré, 4 - dobré. Potom za **vyvážený systém** sa považuje ten, v ktorom všetky časti majú rovnaké hodnotenie alebo najviac tri z nich sa odlišujú najviac o 1 hodnotiaci bod od ostatných. **Nevyvážený systém** je následne taký, ktorý tieto podmienky nespĺňa.

Doporučený stav informačného systému odzrkadľuje, ako dôležitý je pre firmu daný informačný systém. Na ohodnotenie sa používa štvorbodová škála spomenutá vyššie. O čo je daný IS pre firmu dôležitejší, o to väčšia je hodnota doporučeného stavu. Na obrázku 3.2 je znázornenie doporučenej hodnoty vzhľadom na aktuálnu úroveň jednotlivých častí IS. [21] [18]



Obr. 3.2: Znázornenie úrovne jednotlivých častí IS metódy HOS vzhľadom k doporučenej hodnote IS. Z grafu vyplýva nerovnomerná úroveň jednotlivých častí systému a tým pádom nedosiahnutá jeho požadovaná hodnota. Zdroj: [18].

Neoddeliteľnou súčasťou IS sú databázové systémy. Slúžia na uchovávanie, správu a zálohovanie dát, ktoré sú potrebné v rámci IS. Samotná databáza, jej vlastnosti, typ a spôsob, ktorým zaobchádza s dátami, silne ovplyvňuje napojený IS a preto je pre celkový návrh IS a jeho posúdenie kľúčové, aké charakteristiky táto databáza bude mať. Nasledujúca časť popisuje vlastnosti databáze a princípy práce s dátami v nej uložených.

3.2 Databáza

Databáza je systém súborov obsahujúcich perzistentné dáta s pevnou štruktúrou, ktoré sú využívané aplikačnými systémami. Medzi základné vlastnosti dát uložených v databázy patrí [16]:

- perzistencia - dáta pretrvávajú dlhodobo od jednej operácie k druhej
- správnosť - snaha o odhalenie najrôznejších chýb v dátach
- integrovanosť - zjednotenie niekoľkých požadovaných pohľadov do komplexných dátových štruktúr
- zdieľanie - viacpoužívateľský prístup
- bezpečnosť - jednoduchšie realizovanie práv k dátam
- konzistencia - dáta môžu byť dočasne uložené na viacerých miestach, ale musia mať rovnakú hodnotu

Vznik databázy tak, ako ju poznáme dnes sa datuje do 50. rokov minulého storočia. Potreba uchovávať dáta je však oveľa staršia. Za prvú snahu o uchovávanie dát môžeme považovať knižnicu v meste Ugarit, kde sa našlo pohromade viacej hlinených tabuliek s diplomatickým textom pochádzajúcich z dvanásteho storočia pred naším letopočtom. Avšak za predchodcov počítačových databáz sa považuje kartotéka. Tie mali hlavnú nevýhodu v tom, že manipulovať s ňou musel človek. Preto v roku 1890 vytvoril americký štatistik Herman Hollerith pre potreby štátnych úradov počítací stroj, ktorý používal k uchovaniu dát štítky s dierkami a tak prišlo k prvému elektromechanickému spracovaniu dát. [16]

3.2.1 Databázový systém

Databázový systém, inak nazývaný *systém riadenia báze dát* (Database Management System, DBMS) predstavuje softwarové riešenie operácií nad dátami databázy viď obrázok č. 3.3. Poznáme niekoľko druhov databázových systémov, ktoré sa líšia spôsobom uchovávania dát a práce s nimi.

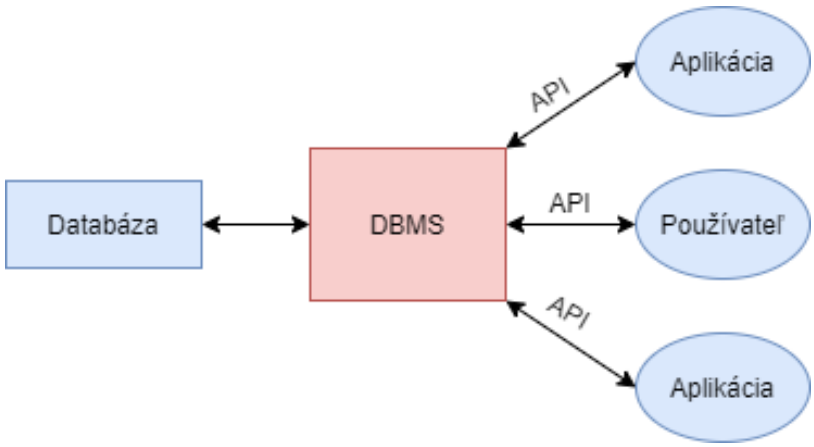
Jedným z druhov DBMS sú relačné databázy, ktoré vychádzajú z relačného modelu dát (relačná algebra). V tomto modely sú dáta usporiadané do tabuliek, kde každá tabuľka zhromažďuje údaje o jednom druhu objektu. Relačné databázy podporujú transakcie. [10]

Medzi základné vlastnosti transakcií zaraďujeme atomicitu, konzistenciu, izolovanosť a trvalosť (Atomicity Consistency Isolation Durability, ACID):

- atomicita - transakcia je ako operácia nedeliteľná (buď sa vykoná ako celok alebo sa nevykoná vôbec)
- konzistencia - pri a po prevedení transakcie neni porušené žiadne integritné obmedzenie
- izolovanosť - operácie v transakcii sú „skryté“ pred vonkajšími operáciami
- trvalosť - zmeny, ktoré sa vykonajú v rámci transakcie sú po jej ukončení uložené do databázy

Medzi najznámejšie relačné databázy patria MS SQL Server, Oracle, PostgreSQL, SQLite alebo MySQL.

Ďalším typom DBMS je noSql databáza, ktorá nevyužíva tabuľky ako je tomu u relačnej. Dôraz je kladený hlavne na výkon pri spracovávaní obrovského množstva dát – cieľom je jednoduchosť aj možnosť horizontálneho aj vertikálneho škálovania. Štruktúra ukladania dát býva hlavne stromová alebo grafová. Na rozdiel od relačných systémov nepodporujú transakčný model ACID. [30]



Obr. 3.3: Znázornenie hierarchií medzi aplikáciou využívajúcou databázu pomocou DBMS. API je programové rozhranie, ktoré predstavuje zbierku dostupných funkcií pre prácu s databázou prostredníctvom DBMS. Zdroj: Vlastná tvorba.

Pri vytváraní nového IS alebo softvéru obecne je potrebné brať v úvahu, že sa nejedná o nejaký v čase konštantný jav, ale že ide o proces vyvíjajúci sa v čase, ktorý môžeme

rozdeliť na rôzne charakterizovateľné fáze, ktoré na seba nadväzujú, môžu sa prelínať alebo opakovať. Ide teda o životný cyklus od vzniku požiadavky na jeho vytvorenie až po údržbu a servis poskytovaný pri jeho finálnom využívaní v reálnej prevádzke. Pohľad niektorých autorov ide až tak ďaleko, že za samostatnú fázu považujú aj likvidáciu [31]. V ďalšej časti budú preto popísané jednotlivé fázy, modely, nástroje a princípy riadenia životného cyklu softvéru.

3.3 Životný cyklus softwaru

Životným cyklusom softwaru sa myslí proces, v ktorom sa potreby používateľa transformujú na požiadavky na software, požiadavky na software sa transformujú na návrh, ktorý sa implementuje, implementácia sa otestuje a nakoniec sa predá používateľovi.

Medzi základné činnosti spojené s vývojom softvéru patria [17]:

- analýza a špecifikácia požiadaviek
- architektonický a podrobný návrh
- implementácia
- integrácia a testovanie
- **prevádzka a údržba**

Ako je možné vidieť v činnostiach spojených s vývojom softvéru, tak podstatná časť vynaloženého času a nákladov je spojená s prevádzkou a údržbou daného systému. Medzi spôsoby ako ušetriť na týchto nákladoch patrí venovať viac úsilia analýze a návrhu riešenia daného systému alebo používanie kvalitných systémov tretích strán, ktoré napomáhajú k efektívnej správe požadovaného systému.

Definovanie prevádzky, údržby a podpory:

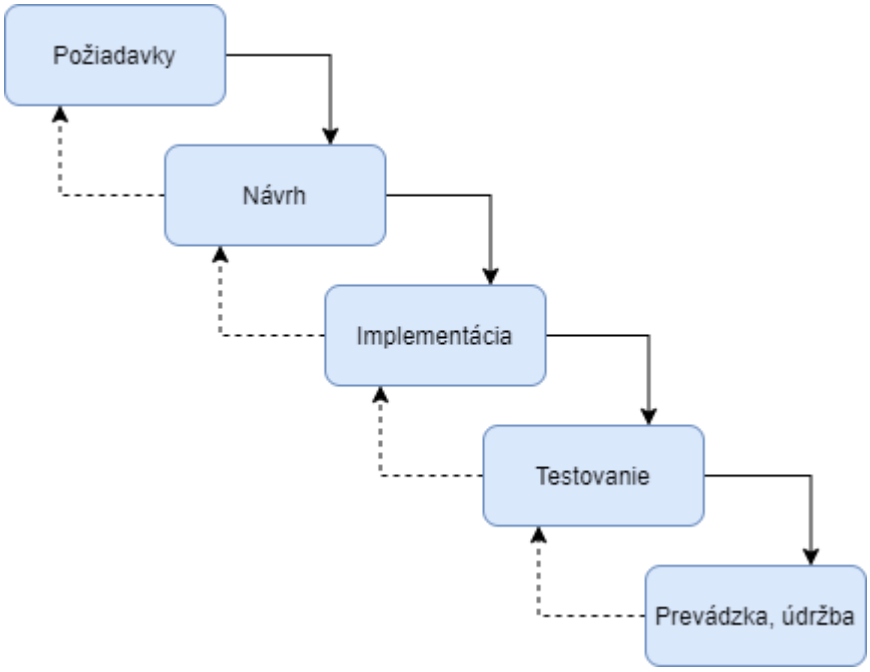
- Zaistenie prevádzky systému, jeho údržbu a rozvoj vzhľadom k novým užívateľským požiadavkám, ktoré sú v súlade so zámermi a cieľmi organizácie.
- **Monitorovanie prevádzky z dôvodu optimalizácie procesov, zistenia využitia aplikácií a prevádzkových chýb.**

- Návrhy zmien a úprav môžu byť funkčného, prevádzkového alebo organizačného charakteru.

3.3.1 Modely životného cyklu softvéru

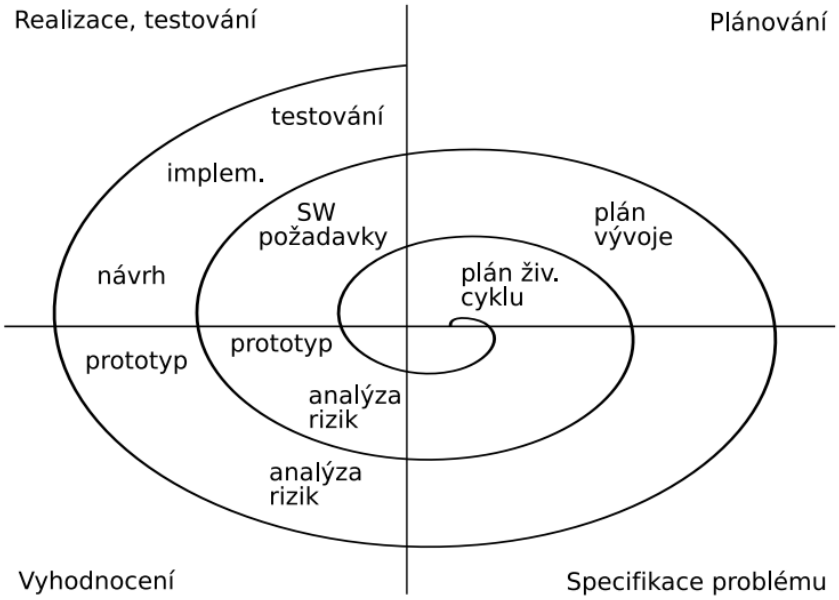
Pre reprezentáciu životného cyklu softwaru sa používajú modely, ktoré definujú jednotlivé kroky (etapy), ktoré je potreba vykonať k úspešnému zvládnutiu daného problému.

Jedným zo základných modelov je vodopádový model 3.4, ktorý je zároveň najstarším modelom. Model bol definovaný v roku 1970 Winstonom W. Roycem [28] ako reakcia na stále zložitejšie projekty v oblasti leteckého priemyslu. Medzi základné fázy modelu patria: Definovanie požiadaviek (systémových a softvérových), analýza a návrh, implementácia, testovanie a prevádzka, kde každá nasledujúca etapa začne až po ukončení predchádzajúcej. Model je možné využiť hlavne v prípade, keď sú dôkladne spracované prvé fázy, keďže odhalenie chyby na začiatku je oveľa lacnejšie ako ich opravovať na konci projektu. Reálne projekty ale väčšinou nesledujú etapy v presnom poradí, užívateľ nie je schopný presne definovať svoje požiadavky na začiatku a vidí reálne výsledky až ku koncu projektu. Ďalšou veľkou nevýhodou je takmer nulová možnosť reagovania na zmenu požiadaviek v priebehu projektu. Problémom je aj samotné testovanie, zahájené až po implementácii, kedy je systém už takmer pripravený na predanie zákazníkovi a každá chyba znamená neúmerné časové aj finančné predraženie projektu. [4] [17]



Obr. 3.4: Schéma vodopádového modelu. Zdroj: Vlastná tvorba.

Iným typom modelu je napr. špirálový model, ktorý je kombináciou prototypovania a analýzy rizík (management). Jednotlivé kroky vo vývoji sa opakujú na vyššom stupni zvládnutej problematiky 3.5. Tento model úspešne pokrýva najväčšie nedostatky vodopádového modelu. To znamená, že prechod do ďalšej fázy je predchádzaný dôslednou analýzou všetkých rizík spojených s projektom. Cyklus modelu môžeme rozdeliť do štyroch základných častí: špecifikovanie problému, vyhodnotenie alternatív, realizácia spojená s testovaním a plánovanie nasledujúcich cyklov. Keďže po každom cykle prebiehajú pravidelné testy, je možné využiť automatické testovanie s drobnými úpravami na základe aktuálneho cyklu projektu. [3]



Obr. 3.5: Schéma špirálového modelu. Zdroj: [17].

3.3.2 Projektové riadenie

Riadenie projektu je riadenie istej vymedzenej sady činností, čo predstavuje organizované úsilie s jasným časovo definovaným cieľom. Jeho účelom je zaistiť efektívne vykonávanie daných činností tak, aby v požadovanom čase za predpokladané náklady bol vytvorený predpokladaný výsledok. Pre dosiahnutie tohoto cieľa je potrebné využívať znalosti, skúsenosti, rôzne nástroje a techniky z oblasti projektového riadenia.

Formou projektu možno realizovať niekoľko druhov produktov, medzi ktoré zaraďujeme napríklad produkty stavebných firiem, podniky zamerané na kusovú výrobu alebo IT firmy, ktoré dodávku realizujú implementovaním rôznych informačných a komunikačných technológií. Tak ako existuje viacero oblastí produkujúcich produkty na základe projektov, tak existuje niekoľko prístupov k jeho riadeniu. Primárne sa delí na tradičný a agilný prístup. [29]

Tradičný prístup

Tradičný prístup je založený na dôkladnom naplánovaní všetkých aktivít na začiatku projektu. Tradičný prístup vyžaduje čo najdetailnejšie popísanie cieľového produktu. Využíva sa hlavne v odvetviach, kde je dopredu jasné, ako bude finálny produkt vyzerať. Jedná

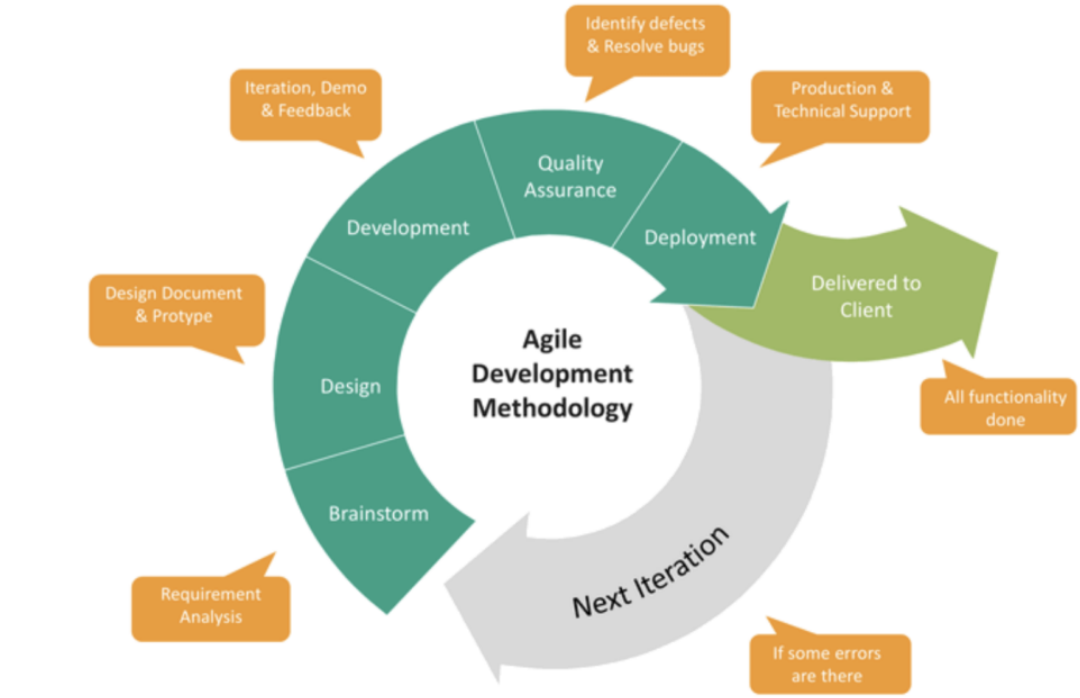
sa napríklad o rôzne stavby alebo pásové vyrábanie súčiastok. Kroky tradičného prístupu môžeme zobecniť do nasledujúcich krokov:

- požiadavka na projekt
- plánovanie a návrh
- realizácia
- monitorovanie
- uzavretie

Agilný prístup

Agilné projektové riadenie je interaktívny spôsob riadenia projektov. Je protikladom tradičného riadenia projektov, tzv. Vodopádového prístupu viď vyššie. Agilné riadenie projektov sa uplatňuje v projektoch, u ktorého je jasný rámcový cieľ, ale z najrôznejších dôvodov je nemožné presne definovať všetky dlhodobé požiadavky. Používa sa teda, keď sa nedá určiť detailný plán projektu vrátane detailných požiadaviek. Agilný prístup je interaktívny, flexibilný a dynamický. Vytyčuje tzv. ihriská pre jednotlivé tímy, na ktorých na základe istých pravidiel majú určitú autonómiu.

Najznámejšou metodikou je metodika SCRUM. Základnými princípmi tejto metodiky sú rozdelenie projektu na krátke kontrolné intervaly - cykly, neustála interakcia so zákazníkom a neformálnosť. Každý cyklus sa skladá z analýzy požiadaviek, návrhu riešenia, implementácie, testovania a predania zákazníkovi viď nasledujúci diagram 3.6 [2]. V každom cykle sú odstraňované nedostatky zistené používaním.



Obr. 3.6: Diagram agilného vývoja. Zdroj: [2].

Softvér pre riadenie projektov

Cieľom projektu je splnenie zadaných úloh, čo najefektívnejším spôsobom. K dosiahnutiu požadovanej efektivity nám môžu dopomôcť rôzne metodiky vedenia projektu alebo široké spektrum nástrojov.

V súčasnosti je na trhu obrovské množstvo nástrojov určených pre sledovanie požiadaviek, ktoré majú byť splnené pre úspešné dokončenie projektu a tak isto poskytujúcich manažérsky prehľad nad projektom. Výber správneho nástroja pre riadenie projektu je pre manažéra kľúčovou úlohou, keďže vybraný nástroj by mal byť súčasťou projektu až do jeho úspešného konca. Prípadná zmena nástroja uprostred vývojového cyklu by mohla znamenať riziko nesplnenia zadaných úloh v stanovenom časovom horizonte.

Vo vývoji softvéru, môže výber správneho nástroja podporujúceho riadenie a sledovanie požiadaviek projektu ovplyvniť rýchlosť procesu vývoja daného softvéru, preto je dôležité vybrať nástroj s vlastnosťami vhodnými pre daný typ projektu.

Medzi základné vlastnosti nástroja pre riadenie projektov, ktoré sú požadované pri vývoji softvéru sú [15]:

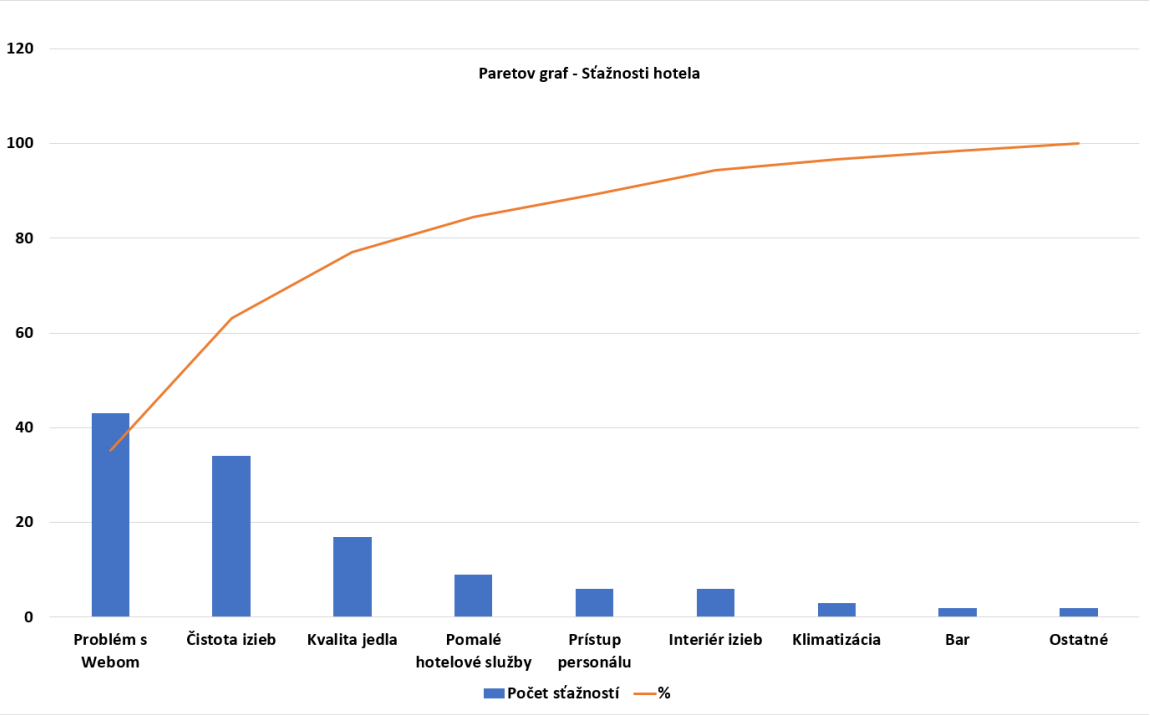
- plán úloh na daný vývojový cyklus - zoznam priradených/nepriradených úloh na daný šprint spolu s ich prioritou
- správa úloh - vytváranie/rozdeľovanie úloh členom tímu
- kalendár - na sledovanie dôležitých termínov
- Ganttov diagram - grafické znázornenie naplánovaných úloh v čase
- wiki stránky - zhromažďovanie rôznych dôležitých informácií a návodov slúžiacich pre jednoduchší a rýchlejší vývoj

Medzi najznámejšie riadiace nástroje patria napríklad: Jira, Redmine, Teamwork Projects, Plutio atd.

3.3.3 Príčiny a dôsledky zlyhania procesov prevádzky

V každom vývojovom cykle sa stretávame s určitými zlyhaniami a neočakávaným správaním vyvíjaných systémov. Tieto zlyhania sa objavujú takmer vo všetkých fázach vývoja návrhu až po údržbu. Vo veľa prípadoch je zdrojom rôznych problémov rovnaká príčina. O tomto tvrdení hovorí takzvané **Paretovo pravidlo**, ktoré môžeme obecne interpretovať tak, že 20% príčin spôsobuje 80% výsledkov. To znamená, že aplikovaním tohoto pravidla môžeme uvažovať, že identifikovaním a podchytením 20% príčin problémov môžeme odstrániť až 80% problémov, ktoré zapríčiňujú. [19]

Dôležitou úlohou je preto analyzovať/identifikovať čo najlepšie spektrum príčin, ktoré môžu ohroziť prevádzku nasadzovaného informačného systému. Táto analýza sa nazýva **Paretova analýza**. Paretova analýza pracuje na základe štatistického vyhodnocovania príčin. Vychádza z tvorby Paretových diagramov, kde sú dáta zoradené podľa toho aký majú celkový vplyv - na ose **x** sú uvedené jednotlivé kategórie problémových javov a na osy **y** ich početnosť. Na obrázku č. 3.7 je znázornený jednoduchý príklad Paretovho diagramu, kde sú znázornené sťažnosti klientov hotela a je vidno, že keď sa hotel zameria na prvé tri príčiny pokryje zhruba 80% problémov.



Obr. 3.7: Príklad Paretovho diagramu. Zdroj: Vlastná tvorba.

3.3.4 Prístupy k prevádzke systémov

Poslednou fázou životného cyklu softwaru je jeho prevádzka, udržiavanie a podpora, ktorá má zabezpečiť jeho bezproblémový chod. Existuje niekoľko možných prístupov, ako sa postaviť k otázke prevádzkovania systémov.

Jedným z najjednoduchších prístupov k prevádzke systémov je pasívne čakanie na vznik nežiadúcich stavov a ich operatívne vyriešenie bez nejakej bližšej analýzy daného problému. Tento prístup je jednoduchý na implementáciu, pretože "nič" sa nasadzuje jednoducho, avšak pri vzniknutí problému je ťažké naň včas zareagovať a preto jeho riešenie môže trvať neúmerne dlho.

Dosiahnutie zvýšenia efektivity riešenia vzniknutých problémov je možné dosiahnuť, pokiaľ je problém správne popísaný hneď pri jeho prvom výskyte a kontakte s používateľom, kde je potreba čo najpresnejšie popísať kroky k jeho vyriešeniu. Preto je našim cieľom **riadený** prístup k prevádzke systémov, ktorý sa riadi určitými vopred definovanými pravidlami. Pri vytváraní takéhoto štandardu je dobré sa inšpirovať už zabehnutými štandardmi, ktoré úspešne fungujú v iných podnikov po celom svete. Jedným z takýchto celosvetovo

uznávaných štandardov na riadenie a správu IT služieb je štandard **ITIL** (Information Technology Infrastructure Library). [6]

3.3.5 ITIL

ITIL je medzinárodne uznávaný štandard používaný pre riadenie a správu IT služieb. Zameriava sa na neustále meranie a zlepšovanie kvality dodávaných služieb, ako z pohľadu dodávateľa tak z pohľadu zákazníka. ITIL nie je norma, je to skôr odporučenie, ktoré obsahuje súbor praxou overených konceptov a postupov, ktoré umožňujú lepšie plánovať, využívať a skvalitňovať IT služby. [6]

Obr. 3.8: Diagram životného cyklu služby podľa ITIL. Zdroj: [13].

ITIL definuje takzvaný životný cyklus služby (obrázok č. 3.8), ktorý pozostáva z nasledujúcich častí [13]:

- stratégia služieb - Predstavuje správu financií, portfólia služieb a požiadaviek.
- návrh služieb - Predstavuje návrh potrebných služieb pre aktuálne aj možné budúce požiadavky. Zahŕňa správu katalógu a úrovne služieb, kapacít, dostupnosti, bezpečnosti a dodávateľov.

- prechod služieb - Predstavuje prechod služieb do produkčného prostredia. Zahŕňa správu zmien, aktív, nasadení, testovaní a vyhodnotení.
- **prevádzka služieb** - Cieľom prevádzky je dodávanie služieb v požadovanej kvalite. Zahŕňa správu udalostí, incidentov, prevádzky atď.
- neustále zlepšovanie služieb - Zabezpečuje zlepšovanie služieb pomocou výsledkov rôznych meraní.

Pri odovzdávaní služby zákazníkovi je potrebné zároveň definovať, čo daná služba obsahuje, ponúka a všetky ostatné podmienky spojené s užívaním danej služby. Táto definícia znamená zmluvu medzi dodávateľom a zákazníkom označovanú ako **SLA** (Service-Level Agreement).

SLA je dohoda o úrovni poskytovaných služieb. Predstavuje popis služby, ktorú poskytuje dodávateľ zákazníkovi. SLA napríklad definuje garantovanú časovú dostupnosť služby, cenu alebo rýchlosť riešenia vzniknutých problémov.

Po oboznámení sa so životným cyklom softvéru je potrebné zaistiť správu verzií jeho jednotlivých fáz. Či sa už jedná o návrh alebo implementáciu, je potrebné mať kontrolu nad verziami pre potreby porovnávania alebo zálohy starších verzií pre prípadný návrat k predchádzajúcemu riešeniu a podobne. K tomuto účelu slúžia verzovacie systémy popísané v nasledujúcej časti.

3.4 Verzovacie systémy

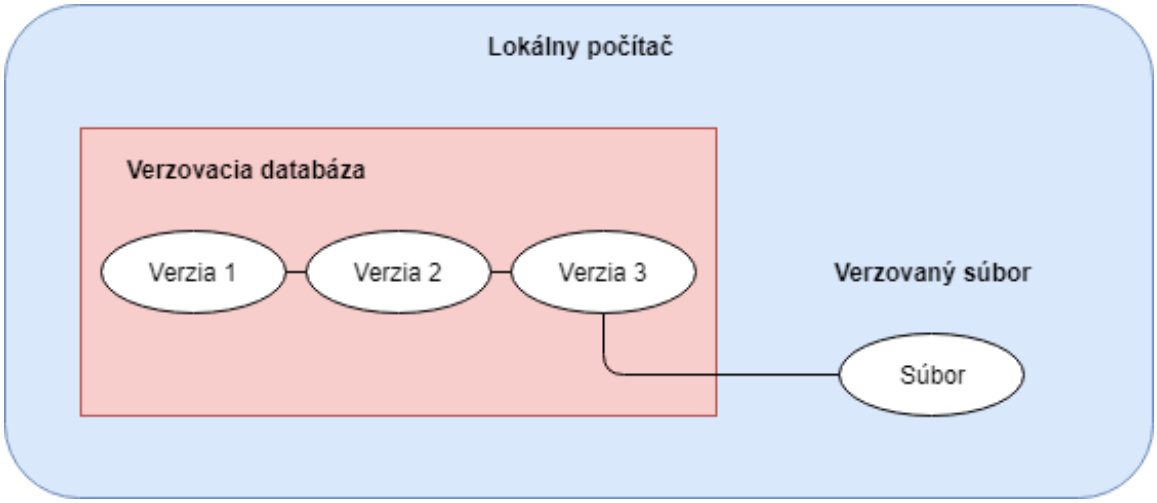
V dnešnej dobe sa žiaden vývojový pracovník nezaobíde bez kvalitného verzovacieho systému. Verzovací systém zaznamenáva zmeny súborov alebo sady súborov v priebehu času. Používateľ je preto schopný obnoviť požadovanú verziu či už súboru, alebo celého projektu. Medzi ďalšie výhody verzovacích systémov patrí možnosť porovnávať zmeny v jednotlivých verziách, zistiť, kto naposledy urobil nejaké zmeny alebo jednoducho obnoviť stratené súbory. Existuje niekoľko typov verzovacích systémov. Podľa spôsobu verzovania ich môžeme rozdeliť nasledovne [7]:

- lokálne systémy správy verzií
- centralizované systémy správy verzií

- distribuované systémy správy verzií

Lokálne systémy správy verzií

Je to jeden zo základných typov verzovania, ktorý používajú väčšinou tí ľudia, ktorí nepoznajú moderné verzovacie systémy. Postup verzovania spočíva v kopírovaní súborov do iného lokálneho adresára. Tento postup je pomerne častý, pretože je jednoduchý a rýchly. Je tu však pomerne veľké riziko chybovosti spočívajúce v nesprávnom/neprehľadnom označení novej verzie a s tým spojená editácia nesprávneho súboru. Z tohoto dôvodu bol vyvinutý lokálny verzovací systém s jednoduchou databázou, v ktorej sa uchováva zoznam zmien medzi jednotlivými verziami v špeciálnom formáte. Príkladom takéhoto systému je systém s názvom res, ktorého princíp je znázornený na obrázku číslo 3.9.

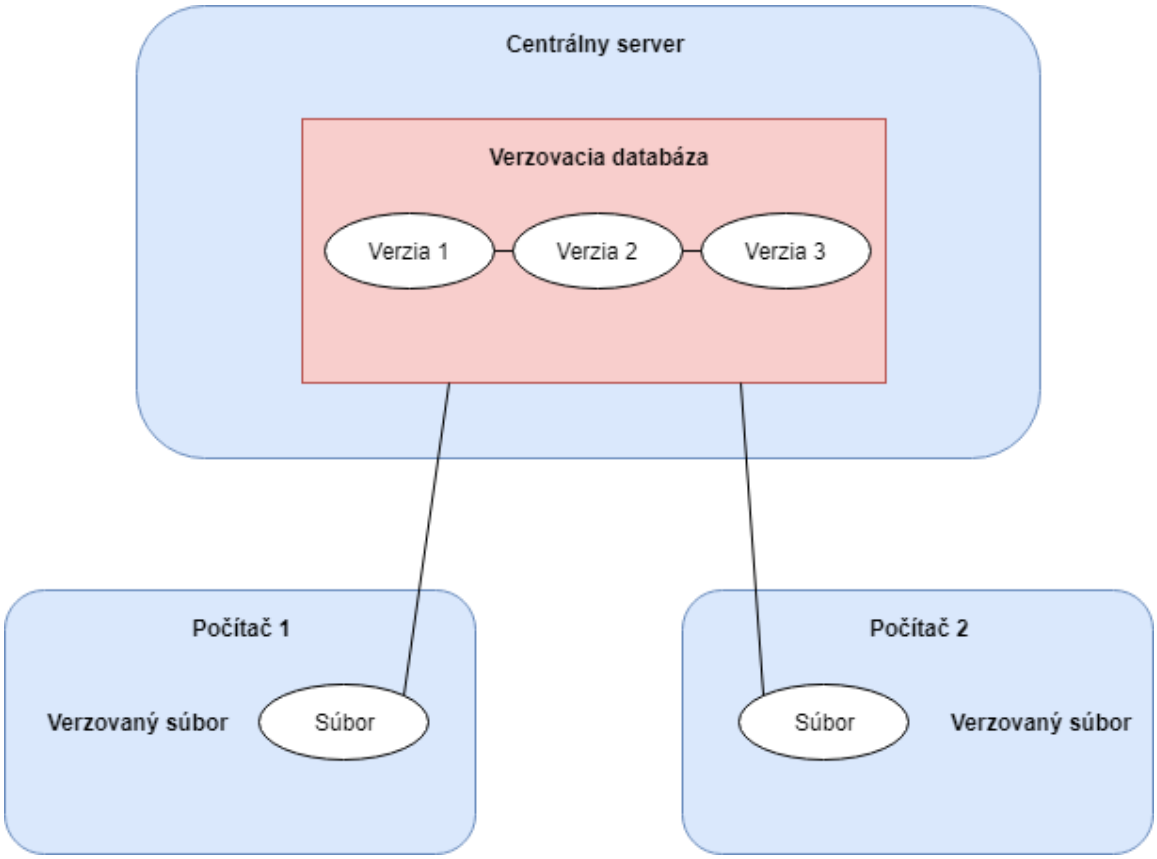


Obr. 3.9: Princíp verzovania lokálnych verzovacích systémov. Zdroj: Vlastná tvorba.

Centralizované systémy správy verzií

Centralizované systémy správy verzií prichádzajú s veľkou výhodou oproti lokálnym systémom. Touto výhodou je možnosť práce všetkých členov tímu s verziami spoločných projektov. Umožnené je to vďaka serverovej časti, ktorá uchováva všetky verzované súbory. Z tohoto centralizovaného miesta majú k nim prístup všetci oprávnení používatelia, ktorí si ich odtiaľ sťahujú. Tento koncept bol dlhé roky štandardom pre správu verzií. Princíp fungovania centralizovaných systémov na správu verzií je znázornený na obrázku číslo 3.10 Hlavným nedostatkom tohoto centralizovaného systému je možnosť výpadku hlavného serveru a tým pádom znemožnenie jeho využívania používateľmi. V prípade výpadku serveru

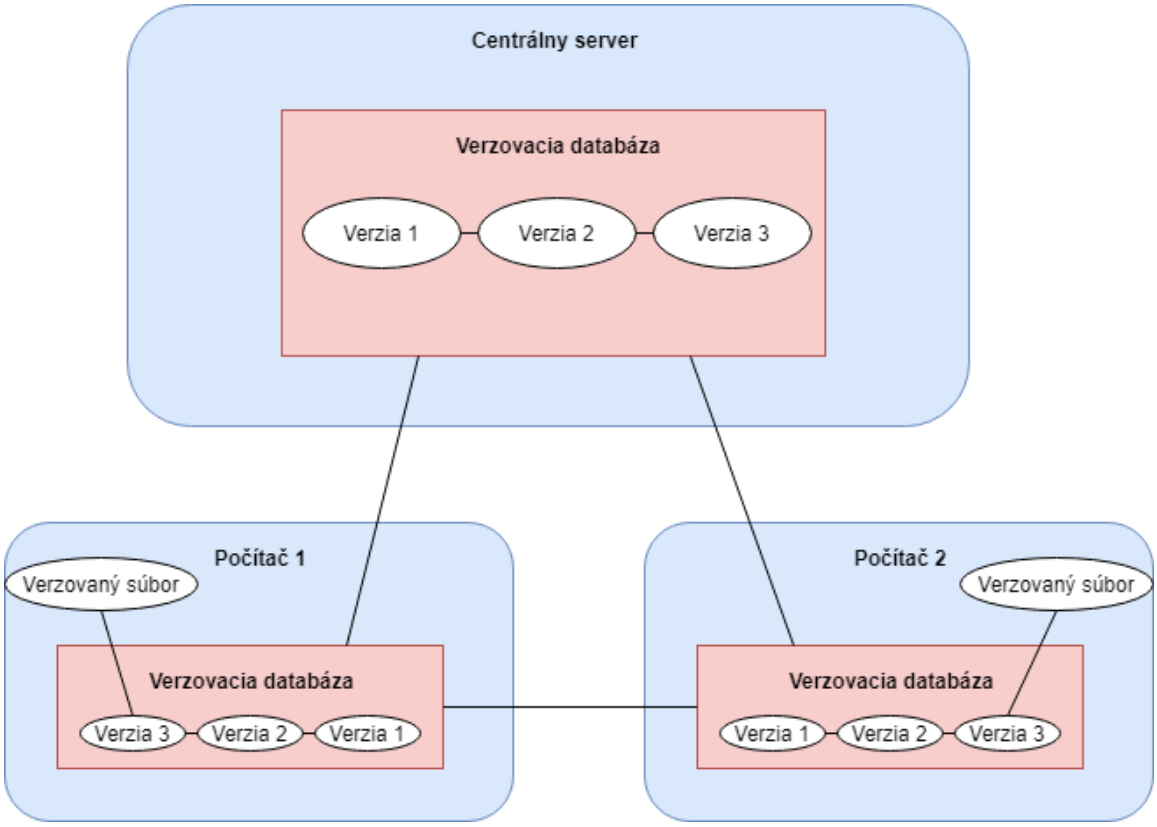
nie je možné ukladať nové verzie projektov, ba dokonca v prípade nezálohovania tohoto serveru môže prísť pri poškodení jeho disku ku strate celej histórie projektu (s výnimkou aktuálnych verzií uložených lokálne v počítačoch používateľov). Medzi najznámejšie systémy pracujúce na centralizovanom princípe patria Subversion alebo Perforce.



Obr. 3.10: Princíp verzovania centralizovaných verzovacích systémov. Zdroj: Vlastná tvorba

Distribuované systémy správy verzií

Pri distribuovaných systémoch správy verzií si používatelia nesťahujú len aktuálnu verziu súborov ale uchovávajú kompletnú kópiu repozitára. Pokiaľ príde v tomto prípade ku zlyhaniu hlavného servera, je možné skopírovať repozitár od ľubovoľného používateľa, ktorý ho má lokálne uložený. Tieto systémy umožňujú prácu s viacerými vzdialenými repozitármi a tak používatelia môžu spolupracovať s viacerými skupinami ľudí súčasne. Najznámejším distribuovaným systémom je GIT. Diagram princípu distribuovaných systémov na správu verzií je zobrazený na obrázku číslo 3.11.



Obr. 3.11: Princíp verzovania distribuovaných verzovacích systémov. Zdroj: Vlastná tvorba.

Ďalším dôležitým krokom je výber správneho programovacieho jazyka pre implementáciu jednotlivých častí dohľadového IS. Dohľadový systém bude slúžiť dispečerom, ktorí nemajú veľa skúseností s programovaním, avšak bude potrebná istá úroveň interakcie so zdrojovými kódmi IS. Preto je potrebné zvoliť programovací jazyk, ktorý bude jednoduchší na pochopenie a zároveň vhodný pre navrhované riešenie IS. Programovací jazyk Python je vďaka veľkému množstvu dostupných knižníc, dobrej čitateľnosti kódu a jednoduchosti implementácie jednou z možností. Spolu s využitím jednoduchého systému webových služieb na komunikáciu jednotlivých častí v rámci IS sa javí ako správna voľba, ktorá bude využitá pri implementácií dohľadového systému.

3.5 Programovací jazyk Python

Python je interpretovaný, objektovo orientovaný programovací jazyk. Vysoko-úrovňové dátové štruktúry kombinované s dynamickým typovaním sú predurčené na rýchle vytváranie prototypov. Ďalšou vlastnosťou je jednoduchá možnosť rozšírenia. Nové zabudované moduly

môžu byť jednoducho napísané v jazyku C či C++. Python je takisto multi-paradigmový jazyk, čo umožňuje používanie viacerých štýlov programovania [11]. Prehľad vybraných potrebných modulov jazyka Python použitých v navrhnutom riešení[12]:

- pyodbc - Modul slúžiaci k jednoduchému pripojeniu ODBC (Open Database Connectivity) databázy. ODBC je štandardizované rozhranie umožňujúce prístup k databázovým systémom. Základným princípom ODBC je prístup k dátam nezávislý na použitom programovacom jazyku. ODBC sa stará o pripojenie k databázy, udržovanie spojenia, získavanie dát a ukončenie spojenia.
- socket - Modul nízko-úrovňového sieťového rozhrania. Môžeme ho definovať ako koncový bod spojenia (endpoint). Je definovaný jedinečnou IP adresou a portom. Na základe použitého transportného protokolu rozlišujeme TCP (spojové) a UDP (nespojové) sockety.
- logging - Modul slúžiaci na záznam informácií do súboru (logovanie).
- json - Modul potrebný pre prácu so súbormi formátu JSON. Umožňuje kódovanie-/dekódovanie a čitateľné zobrazenie JSON formátu.
- win32service - Modul umožňujúci pristupovať k Windows *Service-Control-Managerovi*, vďaka čomu je možné vytvoriť, spustiť/zastaviť službu bežiacu pod systémom Windows.
- httplib - Modul definujúci triedy, ktoré majú za úlohy implementovať klientskú časť HTTP a HTTPS protokolu.

3.6 Webová služba

Webová služba je softvérový systém, ktorý umožňuje komunikáciu zariadení pripojených na sieť. Momentálne rozlišujeme dva typy implementácií webových služieb a prístupu k nim. Jednou z nich je implementácia pomocou jazyka WSDL (Web Services Description Language) a komunikácie prostredníctvom protokolu SOAP (Simple Object Access Protocol) a druhým typom sú takzvané REST (Representational State Transfer) služby. [27]

3.6.1 REST

REST je architektúra rozhrania, navrhnutá pre distribuované prostredie. Rozhranie REST je používané pre jednoduchý prístup k zdrojom, ktorým môžu byť dáta alebo rôzne stavy aplikácie. REST je orientovaný dátovo, na rozdiel od známejších SOAP alebo XML-RPC, ktoré sú orientvané procedurálne. Pre komunikáciu využíva metódy protokolu HTTP - POST, GET, PUT, DELETE, ktoré pre prácu s dátami na serveri postačujú. [27]

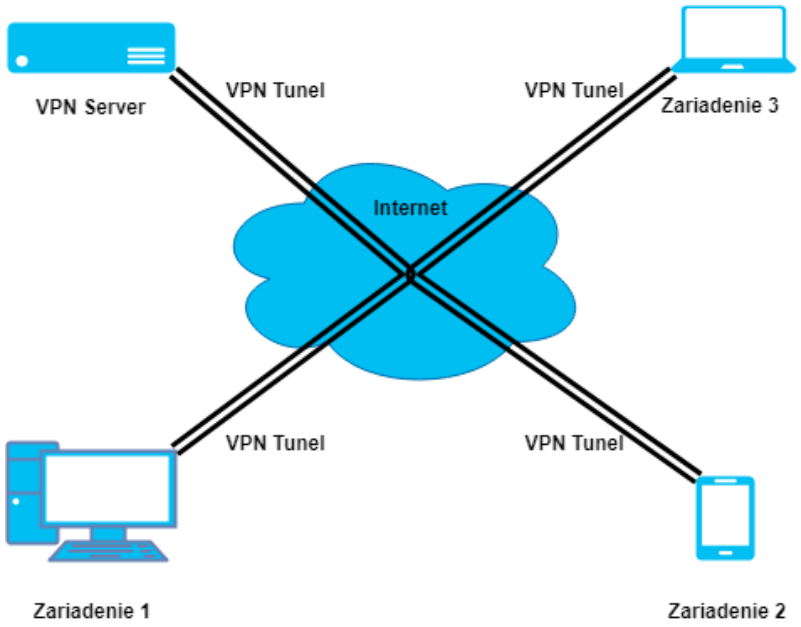
- GET - Metóda slúžiaca na získanie požadovaných zdrojov definovaného v URI (Uniform Resource Identifier) dotazu. Nemení zdroje, preto ju môžeme označiť za bezpečnú metódu. Ďalej musí platiť, že ľubovoľný počet rovnakých dotazov GET vráti vždy rovnaké výsledky pokiaľ neprebehne zmena zdrojov pomocou iných metód (POST, PUT).
- POST - Metóda slúžiaca na vytváranie zdrojov definovaných v URI dotazu. V tele dotazu je definovaná hodnota vytváraného zdroja.
- PUT - Metóda na zmenu zdroja je podobná POST metóde. Rozdiel je v tom, že mení konkrétny zdroj definovaný v URI dotazu a v tele definuje jeho novú hodnotu.
- DELETE - Metóda slúži na vymazanie požadovaného zdroja definovaného v URI dotazu.

Pri každom projekte je potrebné zaistiť jeho bezpečnosť. Pri softvéri sa jedná hlavne o zamedzenie odcudzenia alebo znehodnotenia dát, ktoré by mohlo napáchať veľké škody. Samozrejmosťou je ochrana proti odpočúvaniu prenášaných informácií, tj. šifrovanie, ktoré zaistí nečitateľnosť informácií pre prípadného útočníka disponujúceho možnosťami odpočúvať sieťovú komunikáciu. Jedným zo základných a účinných spôsobov ochrany je použitie virtuálnej privátnej siete. Táto technológia je už teraz široko používaným nástrojom vo vybranej firme (a samozrejme nielen v nej). Virtuálna privátna sieť umožňuje bezpečné spojenie naprieč rôznymi platformami (operačnými systémami) a má veľmi široké využitie. Nasledujúca časť teda popisuje jej základné princípy.

3.7 Virtuálna privátna sieť

V každej časti života je kladený veľký dôraz na bezpečnosť. Inak tomu nie je ani v digitálnom svete, kde pri surfovaní po internete zanechávame digitálne stopy a teda naše súkromné alebo firemné informácie sú v neustálom ohrození. Určitú mieru zabezpečenia poskytuje *virtuálna privátna sieť* (Virtual Private Network, VPN). VPN slúži ako prostriedok, pomocou ktorého je možné bezpečné spojenie viacerých zariadení pomocou nedôveryhodnej siete - napríklad internetu. Vytvorením VPN sa dá dosiahnuť komunikácie medzi zariadeniami za podobných podmienok, ako keby boli prepojené v rámci jednej uzatvorenej siete.

V okamžiku vytvárania spojenia medzi zariadeniami vo VPN je overovaná totožnosť oboch strán pomocou digitálnych certifikátov. V praxi je architektúra VPN tvorená serverom s verejnou adresou aby bol prístupný všetkým požadovaným zariadeniam. Tento server slúži na overenie zariadení a ich certifikátov. Po overení zariadení je medzi nimi vytvorený takzvaný tunel, cez ktorý prebieha bezpečná komunikácia, ktorá je šifrovaná. Diagram príkladu VPN je zobrazený na obrázku číslo 3.12. [8]



Obr. 3.12: Diagram virtuálnej privátnej siete. Zdroj: Vlastná tvorba.

Pre implementáciu a nasadenie nového IS do prostredia firmy je potrebné dôkladne poznať jej aktuálny stav a prostredie, v ktorom pôsobí. K tomuto účelu slúži strategická analýza. Ďalším dôležitým krokom je analýza možných rizík, ktoré sú s nasadením IS spojené

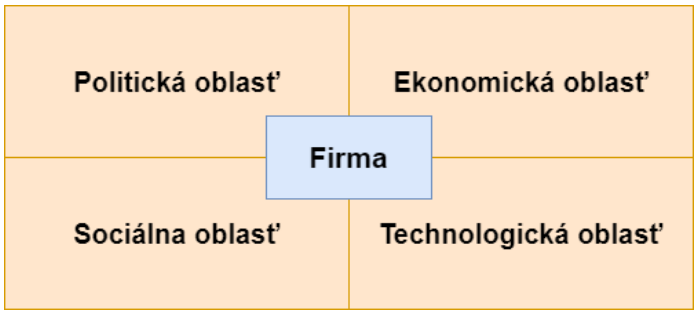
a ich následná eliminácia. Posledným krokom, ktorý je nevyhnutne spojený so zavedením nového IS je navrhnutie spôsobu zavedenia zmeny, k čomu sa hodí využiť tzv. Lewinom model.

3.8 Strategická analýza

Každá zmena vo firme si vyžaduje analyzovať faktory, ktoré ju môžu ovplyvniť a tak narušiť jej hladký priebeh. Faktory, ktoré pôsobia na firmu môžeme najzákladnejšie rozdeliť na vnútorné a vonkajšie. Pre kvalitné analyzovanie stavu firmu je potrebné zohľadniť všetky spomínané faktory. Takáto analýza sa potom nazýva strategická. Jednotlivé metódy podporujúce túto analýzu sú popísané v tejto sekcii.

3.8.1 PEST analýza

PEST analýza slúži na analyzovanie vonkajších faktorov pôsobiacich na firmu v rôznych oblastiach. Skratka PEST reprezentuje tieto oblasti. Sú to politická, ekonomická, sociálna a technologická oblasť. Používa sa hlavne ako súčasť strategického managementu, kedy sa firma rozhoduje podniknúť nejakú strategickú zmenu. Princíp PEST analýzy je ukázaný aj na obrázku číslo 3.13, kde je vidieť postavenie firmy interagujúcej s jednotlivými časťami jej okolia. [14]



Obr. 3.13: PEST analýza. Zdroj: Vlastná tvorba.

3.8.2 McKinseyho model 7S

Pomocou McKinseyho modelu 7S analyzujeme vnútorné faktory firmy. Patrí medzi najpoužívanejšie analýzy strategického managementu. 7S predstavuje 7 oblastí, ktorým sa analýza venuje. Tieto oblasti možno rozdeliť na dve skupiny. Do prvej skupiny zaraďujeme štruktúru

firmy, jej stratégiu a systémy, ktoré využíva. Táto skupina oblastí je dobre známa, pretože je dôkladne popísaná rôznymi firemnými dokumentami. Preto sú aj prípadné zmeny v týchto oblastiach pomerne jednoducho realizovateľné. Do druhej skupiny patrí štýl spoločnosti, spolupracovníci, schopnosti a zdieľané hodnoty. Informácie z týchto oblastí nie sú jednoducho dohľadateľné, čo sťažuje aj ich prípadné zmeny. [1]

3.8.3 SWOT analýza

SWOT analýza sa radí medzi základné metódy strategickej analýzy. Poskytuje potrebné podklady pre formuláciu rozvojových smerov, aktivít a strategických cieľov podniku. Pozostáva z vnútornej analýzy stavu podniku a analýzy situácie vonkajšieho prostredia.

V rámci vnútornej analýzy stavu podniku sú identifikované **silné** a **slabé** stránky, pričom snaha podniku je maximalizovať silné stránky a minimalizovať tie slabé. Z hľadiska vonkajšej analýzy identifikujeme **príležitosti**, ktoré znamenajú možnosť rozvoja podniku a jeho posilnenie na trhu a **hrozby**, ktoré predstavujú ohrozenia dosiahnutia podnikových cieľov. [14]

3.9 Riziková analýza

So zavádzaním každej zmeny je spojené určité riziko. Pre zaistenie úspešnej zmeny je potrebné riziká identifikovať a v čo najväčšej miere eliminovať. V nasledujúcej časti sú popísané metódy znižovania rizík a práce s nimi.

3.9.1 Metódy znižovania rizika

Po správnom identifikovaní rizík je potrebné nájsť vhodný spôsob pre zníženie ich vplyvu. Medzi základné metódy znižovania rizika zaraďujeme [25]:

- retencia rizika - V tomto prípade podstupujeme dané riziko. Toto podstúpenie môže byť buď vedomé alebo nevedomé a čo sa týka rozhodnutia jeho podstúpenia dobrovoľné (riziko je prijaté pretože neexistuje lepší variant) alebo nedobrovoľné (riziku sa nevieme vyhnúť).
- redukcia rizika - Sem zaraďujeme presun rizika na iný subjekt (Defenzívny prístup, napríklad leasing), diverzifikáciu (Rozloženie rizika na čo najväčšiu základňu napríklad investičné nástroje) a metódy operačnej analýzy.

Skórovacia metóda

Skórovacia metóda je rovnako využívaná na analýzu rizík. Táto metóda sa skladá z nasledujúcich troch fáz [25]:

- identifikovanie rizík
- ohodnotenie identifikovaných rizík - Hodnotí sa pravdepodobnosť výskytu a úroveň dopadu daného rizika.
- navrhnutie opatrení na zníženie identifikovaných rizík

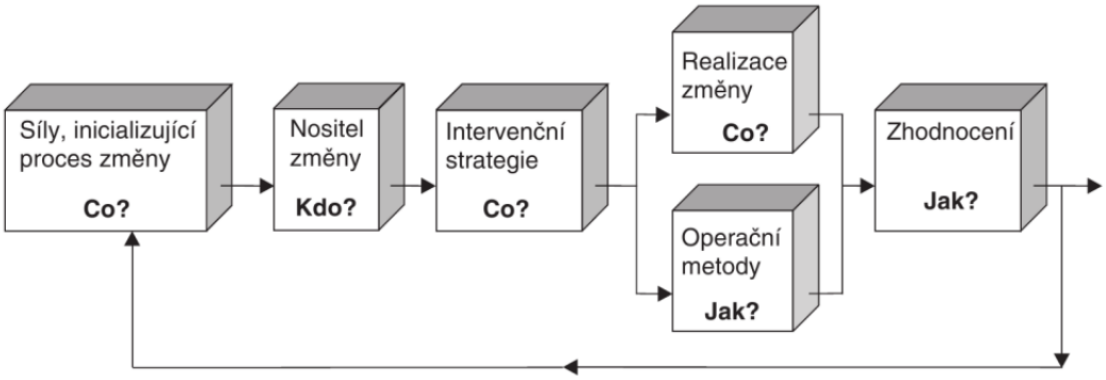
Skórovacia metóda nám pomáha určiť, ktorým rizikám je potrebné venovať väčšiu pozornosť. Identifikované riziká sú na základe ich pravdepodobnosti a výskytu zakreslené do takzvanej mapy rizík, reprezentovanej dvojrozmerným grafickým zobrazením. Mapa sa skladá z troch častí - menej závažné, stredné a kritické riziko. [25]

3.10 Lewinov model

Lewinov model sa používa pri zavádzaní riadenej zmeny v prostredí firmy. Pred zavedením každej zmeny je potrebné poznať odpovede na otázky, ktoré sú zapuzdrené v Lewinovom modely. Ako je zobrazené na obrázku číslo 3.14, je potrebné identifikovať sily inicializujúce zmenu, oblasti, ktoré zmena ovplyvní a osoby zodpovedné za vykonanie zmeny. Ďalej je potrebné definovať ako a kedy bude zmena realizovaná. [25]

Podľa Lewina sa teda úspešná zmena skladá z troch krokov:

- rozmrazenie súčasného stavu firmy
- realizácia zmeny
- zamrazenie stavu po aplikovaní zmeny



Obr. 3.14: Princíp Lewinovho modelu. Zdroj: [25].

Náplň tejto práce si kladie za ciel dať navrhovanej zmene určitý časový rámec. Rozvrhnutie prác a základná identifikácia kľúčových činností v rámci celého tohoto procesu sú pochopiteľne dôležitým faktorom pri posudzovaní a rozhodovaní o nutnosti (či vhodnosti) jeho realizácie. Existujú mnohé spôsoby vytvorenia harmonogramu činností. Jednému z najpoužívanejších spôsobov ako prehľadne zobraziť harmonogram činností, ktorý zároveň vyznačuje činnosti zásadné pre hladký priebeh prác sa venuje tzv. sieťová analýza. Keďže je vhodná aj pre účely tejto práce, obsahuje nasledujúca kapitola jej popis.

3.11 Sieťová analýza

Sieťová analýza je využívaná pri plánovaní projektov, ktoré sa skladajú z viacerých na seba nadväzujúcich činností. Tieto činnosti môžu mať viacero vzájomných vzťahov ako napríklad časová následnosť, nezávislosť, podmienenosť a podobne.

Medzi základné metódy sieťovej analýzy zaraďujeme: [26]

- metóda kritickej cesty CPM (Critical Path Method) - Cieľom metódy CPM je určenie doby trvania projektu na základe takzvanej kritickej cesty, ktorá reprezentuje sled činností s nulovou časovou rezervou. Každý projekt obsahuje minimálne jednu kritickú cestu.
- metóda PERT (Program Evaluation and Review Technique) - Metóda PERT je zovšeobecnením metódy CPM. Používa sa pri projektoch, ktorých činnosti majú stochastický charakter. Trvanie každej činnosti sa chápe ako náhodná premenná s určitým rozdelením pravdepodobnosti.

S uvedenými metódami sieťovej analýzy súvisí pojem sieťový graf, ktorý je grafickou reprezentáciou jednotlivých činností projektu. Zobrazuje následnosť činností spolu s ich časovými údajmi ako možným začiatkom, dobou trvania a možným koncom. Grafy sú buď uzlovo alebo hranovo ohodnotené, kde uzly hrany reprezentujú činnosti a vzťahy medzi nimi. [26]

Kapitola 4

Analýza súčasného stavu

V dnešnej dobe je na trhu veľké množstvo systémov, ktoré sa venujú automatizácií dohľadu nad rôznymi zariadeniami. V tejto kapitole prebehne analýza najvhodnejších kandidátov na nasadenie do zvolenej firmy. Ďalej bude predstavená zvolená firma AŽD Praha s.r.o., jej štruktúra a popis činnosti.

4.1 Základné informácie o spoločnosti AŽD Praha s.r.o.

Názov: AŽD Praha s. r. o.
Sídlo: Záběhlice, Žirovnická 3146/2, 106 00 Praha
IČO: 48029483
Dátum zápisu: 17. 1. 1992

4.1.1 Popis činnosti

Podľa zdroja [22] je *AŽD Praha s.r.o.* (AŽD) významným rýdzo českým dodávateľom a výrobcom zabezpečovacej, telekomunikačnej, informačnej a automatizačnej techniky, predovšetkým zameranej na oblasť koľajovej a cestnej dopravy vrátane telematiky a ďalších technológií. Spoločnosť zaisťuje výskum, vývoj, projektovanie, výrobu, montáž, rekonštrukcie a servis zariadení, systémov a investičných celkov v týchto hlavných oblastiach:

- železničná doprava
- prevádzka metra
- oblasť telekomunikačných, informačných a rádiových systémov

- telematické aplikácie
- cestné, signalizačné a parkovacie systémy
- nové telefónne a rozhlasové systémy pre riadenie železničnej dopravy a pre informovanie cestujúcich

Produkty, ktoré spoločnosť vyrába, zachytávajú najnovšie technické trendy. Vo firme je v súčasnosti vyše 1500 zamestnancov. Tradícia firmy siaha až do roku 1954, kedy firma získala stálu pozíciu na trhu vo svojom obore. V rovnakom obore pôsobia aj nasledujúci významný partneri spoločnosti:

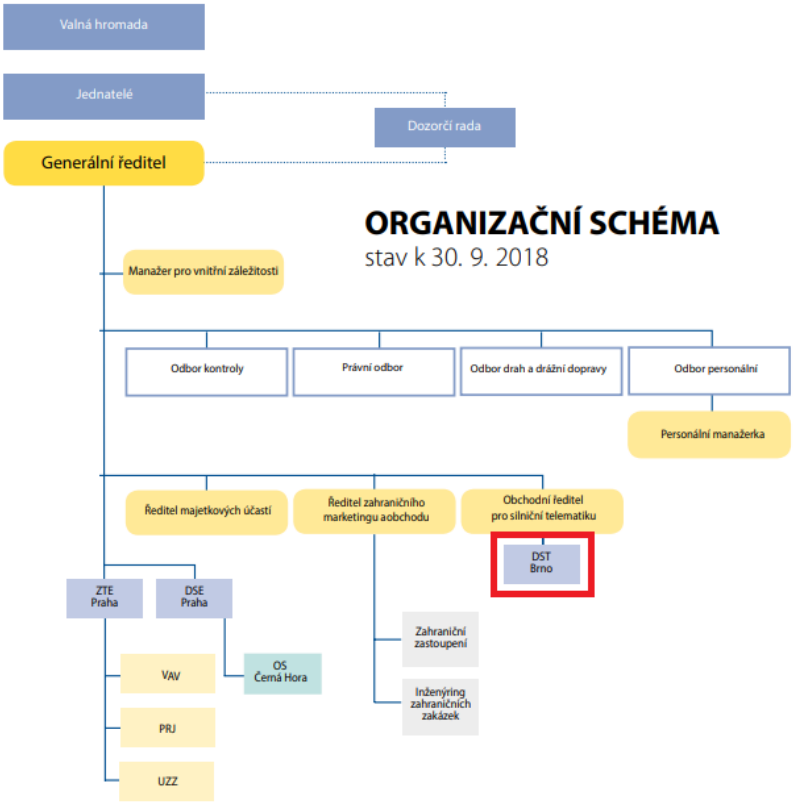
- koľajová doprava - SŽDC, České dráhy, Dopravný podnik Praha
- cestná doprava – Metrostav, SSŽ, PSVS, Skanska, magistráty a mestské úrady miest a obcí ČR

4.1.2 Organizačná štruktúra

Spoločnosť AŽD sa skladá okrem iného z desiatich základných organizačných jednotiek, ktoré zaisťujú vlastnú činnosť firmy [22]:

- Riaditeľstvo spoločnosti
- **Divízia Automatizácie cestnej techniky Brno**
- Divízia Servisu zdelovacej a zabezpečovacej techniky
- Divízia Teleinformatika
- Montážny závod Kolín
- Montážny závod Olomouc
- Výrobný závod Brno
- Výrobný závod Olomouc
- Zásobovací a odbytový závod Olomouc
- Závod Technika

Upravená organizačná štruktúra je znázornená na obrázku číslo 4.1.



Obr. 4.1: Časť organizačnej štruktúry firmy AŽD. Zdroj: [22] (upravený obrázok).

4.1.3 Divízia Automatizácie cestnej techniky Brno

V roku 1993 bola založená *divízia Automatizácie cestnej techniky* (Divize Automatizace Silniční Techniky, DAST) so sídlom v Brne, v ktorom boli sústredené všetky aktivity AŽD v oblasti cestnej techniky. Bol tu vytvorený uzavretý technologický reťazec – vlastný vývoj, projektovanie, výroba a montáž, servisná, poradenská a obchodná činnosť. Zo strategických dôvodov bol v roku 2006 v rámci riaditeľstva spoločnosti vytvorený obchodný úsek pre cestnú telematiku, ktorý je organizačne nadriadený DAST. Tento úsek prevzal zaisťovanie všetkých obchodných aktivít a koncepcií rozvoja činnosti v oblasti cestnej dopravy. [22]

DAST sa už tradične zameriava na výrobu a inštaláciu radičov svetelnej signalizácie. V Českej republike a na Slovensku sú radiče spoločnosti AŽD na viac, ako 100 križovatkách. Ponúka komplexné systémové riešenia v oblasti cestnej telematiky, konkrétne riešenie pre mestskú a medzimestskú dopravu, pre dopravu v tuneloch a riešenie pre verejné osvetlenie.

Divízia tak isto zaisťuje komplexné riešenie pre Detské dopravné ihrisko, od návrhu až po jeho realizáciu. [22]

4.2 Strategická analýza

V nasledujúcej časti sa zameriame na strategickú analýzu firmy AŽD, pomocou analýz vnútorných a vonkajších faktorov vplývajúcich na zmenu stratégie servisných procesov. V závere bude celkové zhodnotenie pomocou SWOT analýzy spolu s odporúčaním na zmenu.

4.2.1 Analýza vonkajších faktorov PEST

Každá firma pôsobí v nejakom prostredí, ktoré rovnako pôsobí na ňu. Firma AŽD nie je výnimkou a z tohoto dôvodu je potrebné analyzovať možné vplyvy z okolia, ktoré môžu výrazne ovplyvniť chod servisných procesov vo firme. Na túto analýzu sa využije metóda PEST, ktorá skúma vonkajšie faktory zoskupené do štyroch základných oblastí: sociálna, ekonomická, politická a technologická. Analýza prebehne v náväznosti na analýzu súčasného stavu 4.4 a pokúsi sa predikovať nasledujúci vývoj v spomínaných oblastiach.

Politická oblasť

Politická oblasť je dôležitou oblasťou pre každú firmu pôsobiacu v štáte. Za najväčšiu hrozbu a súčasne príležitosť môžeme považovať parlamentné voľby. Nasleduje zhodnotenie možných variant zloženia budúcej vlády:

Všeobecne pri zmene orientácie vlády z pravicovej na ľavicovú a opačne možno očakávať zmeny v pracovnoprávnej legislatíve. Pravicová vláda bude mať liberálnejší postoj k zamestnávaniu a bude presadzovať menej štátnej regulácie pre zamestnávateľa. Ľavicová vláda bude mať naopak sociálnejší program zameraný na zamestnancov a ich výhody.

V prípade, že by voľby vyhrala populistická strana, možno očakávať uzavretie sa pred ostatnými štátmi, čo môže spôsobiť výrazný odliv a obmedzenie lacnej zahraničnej pracovnej sily.

Ekonomická oblasť

Situáciu podniku do veľkej miery ovplyvňuje ekonomická situácia. Zlá ekonomická situácia môže výrazne zhoršiť plnenie strategických cieľov firmy. Existuje mnoho ekonomických faktorov, ktoré je potreba sledovať. Medzi tieto faktory zaraďujeme napríklad:

- aktuálny cyklus svetovej a tuzemskej ekonomiky (Recesia, depresia atď.)
- politika (hospodárska, monetárna, fiškálna)
- miera inflácie/deflácie
- situácia na kapitálovom trhu

Momentálne sa nachádzame v období ekonomického rastu a znižovania nezamestnanosti. Postupne sa otvárajú cesty európskym produktom na ázijských trhoch. Zaznamenávame príliv lacnej sily z východu a zemí sužovaných vojnami, čo môže byť v menej kvalifikovaných odvetviach prínosom. Dôležitým ekonomickým prínosom je možnosť čerpať finančné prostriedky z fondov Európskej únie. So zvyšovaním životnej úrovne a hustoty dopravy sa požadujú aj kvalitné zabezpečovacie zariadenia (aj firmy AŽD), ktorých cena sa rokmi osvedčeného používania zvyšuje.

Sociálna oblasť

Sociálna oblasť sa týka ľudí a ľudskej spoločnosti, ktorá môže ovplyvniť vývoj vo firme:

- životná úroveň - Momentálne má životná úroveň rastúcu tendenciu. Tento rast sa predpokladá aj do nasledujúceho obdobia. Z tohoto dôvodu budú zamestnanci na všetkých pozíciách očakávať čoraz vyššie mzdy, aby boli pokryté ich bežné náklady, ktoré zvyšujúcou sa životnou úrovňou rastú. Toto môže byť pre firmu hrozbou, keďže má veľké množstvo zamestnancov.
- flexibilita - Dnes je už samozrejmosťou dopravný prostriedok v každej domácnosti. Dopravná infraštruktúra sa stále vyvíja a ľudia sú čím ďalej ochotnejší za prácou dochádzať aj zo vzdialenejších miest. Veľkou motiváciou je aj dochádzanie z menších miest a obcí do väčších za vyššou mzdou. Ďalšou príležitosťou je stále viac populárna práca z domova (Home office), ktorú na určitých pozíciách môže firma poskytnúť zamestnancom ako benefit. Takáto práca z domova môže prispieť k zníženiu nákladov zamestnanca (dochádzanie) aj zamestnávateľa (viac voľných miest na pracovisku, menši spotreba energie a podobne).

- znalosť cudzích jazykov - Tu je možné rovnako vidieť postupný nárast u ľudí všetkých vekových kategórií, čo umožňuje lepšie uplatnenie na trhu práce v tuzemsku ale aj v zahraničí.

- IT gramotnosť - Postupne sa zvyšujúca gramotnosť ľudí v oblasti IT otvára ľuďom množstvo nových pracovných pozícií

Technologická oblasť

Vývoj technológií vo všeobecnosti postupuje raketovým tempom, čo ovplyvňuje takmer všetky oblasti života. Tak isto je dobre prosperujúca firma odkázaná na prvotriedne technické vybavenie, aby bola schopná konkurencie.

Medzi základné technologické faktory do veľkej miery ovplyvňujúce firmu môžeme zaradiť rozvoj internetu a mobilných sietí. Jedná sa hlavne o rozširujúcu sa dostupnosť služieb a prenosových kapacít. Toto zásadne ovplyvňuje servisné procesy vzdialených zariadení, ktoré firma spravuje. Dôležitým prvkom je potreba zvýšenej úrovne kybernetickej bezpečnosti.

4.2.2 Analýza vnútorných faktorov 7S

Rovnako dôležitým bodom pri návrhu a zavádzaní zmeny je analýza vnútorných faktorov vo firme. Nasledujúca sekcia sa venuje tejto analýze pomocou McKinseyho modelu 7S. Tento model pozostáva z nasledujúcich častí.

Štruktúra

Keďže spoločnosť má pomerne rozsiahlu oblasť pôsobnosti, organizačná štruktúra 4.1 je dosť rozvetvená. Avšak z hľadiska stratégie servisných procesov je vyčlenená divízia DAST popísaná v odstavci 4.1.3, ktorá má na starosti servis dodávaných zariadení a v tejto oblasti je viac menej autonómna. Kompetencie ľudí zodpovedných za servisné procesy sú jasne definované a tak organizačnú štruktúru môžeme považovať za vyhovujúcu.

Stratégia

V tejto časti sa zhodnotí stratégia servisných procesov, u ktorých je navrhovaná zmena. Firma AŽD pôsobí v oblasti zabezpečovacích zariadení, kde má väčšinový podiel na trhu. Z tohoto dôvodu nebol kladený veľký dôraz na zlepšovanie týchto procesov spojených so servisom dodávaných zariadení, pretože zákazníci nemali príliš veľký výber.

Postupom času sa ale objavujú menšie firmy s rovnakým zameraním, ktoré môžu ohroziť postavenie firmy AŽD. Ďalším dôvodom na zmenu je pribúdanie nových zariadení, ktoré

je treba monitorovať a vykonávať ich servis. Vzhľadom na počet zamestnancov venujúcich sa servisu a aktuálnym postupom popísaným v odseku 4.4 je navrhovaná zmena vo forme automatizácie určitých procesov.

Od správnej implementácií automatizácie procesov si firma AŽD sľubuje úspory v oblasti nákladov na dohľad a servis rýchlejším reagovaním na vzniknuté servisné udalosti. Táto zmena je orientovaná dlhodobo na základe pribúdajúcich zariadení, ktoré má firma pod správou. Pri aktuálnom trende nárastu zariadení by firma bez zmeny stratégie potrebovala prijať niekoľko ďalších zamestnancov, čo je pri aktuálnej situácií na trhu práce a z hľadiska finančného nevhodné riešenie.

Systémy

Systémy vo firme AŽD sú na vysokej úrovni. Prenos informácií zo zariadení do systému je rýchly avšak reakcia na vzniknutú situáciu môže trvať dlhšie. Chýba v systéme komunikačný bod, ktorý by urýchlil presun informácií priamo zodpovednej osobe. Týmto systémom by mohol byť ADS na dohľad zariadení, ktorý by v prípade neočakávanej udalosti ihneď kontaktoval vedenie či už zaslaním e-mail, sms správy alebo inou signalizáciou v systéme. Toto riešenie by urýchlilo reakčný čas k vykonaniu potrebných servisných úkonov a tým pádom zníženiu strát, ku ktorým dochádza napríklad nefunkčným zariadením. Nasadenie tohoto systému nevyžaduje najmodernejšie technológie, ktoré sú momentálne vo vývoji (napríklad 5G sieť) a tak je možná okamžitá implementácia s využitím aktuálnych systémov firmy AŽD.

Spolupracovníci

Prijímanie zamestnancov závisí od druhu pozície, na ktorú nastupujú. Firma vyberá pre každú pozíciu len zamestnancov s náležitým vzdelaním a praxou, ktorí prejdú úvodným pohovorom, poprípade vstupným testom. Následne svojich zamestnancov firma AŽD pravidelne vzdeláva formou rôznych školení.

Na druhej strane sú zamestnanci motivovaní k nadštandardným výkonom dobre prepracovaným systémom odmeňovania. Systém odmeňovania je na základe dlhodobejších výsledkov jednotlivých zamestnancov, ktorí sú hodnotení ako ich nadriadenými, tak aj spoločne ako tímy v rámci jednotlivých organizačných zložiek.

Schopnosti

Keďže firma je na trhu vyše 60 rokov, má bohaté skúsenosti vo výrobe, dodávke a servise zariadení. Počas takto dlhého obdobia pôsobnosti sa firma musela vyvíjať spolu

s prostredím, v ktorom fungovala a prekonať niekoľko veľkých zmien. Z tohoto dôvodu myslím, že nadobudla dostatok skúseností a kapacít, aby bola schopná zariadiť aj potrebnú zmenu stratégie servisných procesov.

Tieto dlhoročné skúsenosti sú veľkým plusom, avšak je potrebné dbať na to, aby sa neustále vyvíjali a tým sa zabránilo ich zastarávaniu.

Štýl

Firma má jasne definovanú organizačnú štruktúru a spôsob internej komunikácie. Každé oddelenie má určitý počet skúsených odborníkov ako aj mladých zaúčaných zamestnancov, ktorí navzájom neustále komunikujú, aby bola práca vykonávaná čo najefektívnejšie. Ďalej je definované veľké množstvo noriem, zabezpečujúcich dodržiavanie presných pracovných postupov a podobne. S týmito normami sa musí každý zamestnanec oboznámiť. V pravidelných intervaloch dochádza ku kontrole a aktuálnosti týchto noriem, ktorých prípadné zmeny sú rozposlané firemným e-mailom všetkým zamestnancom na preštudovanie.

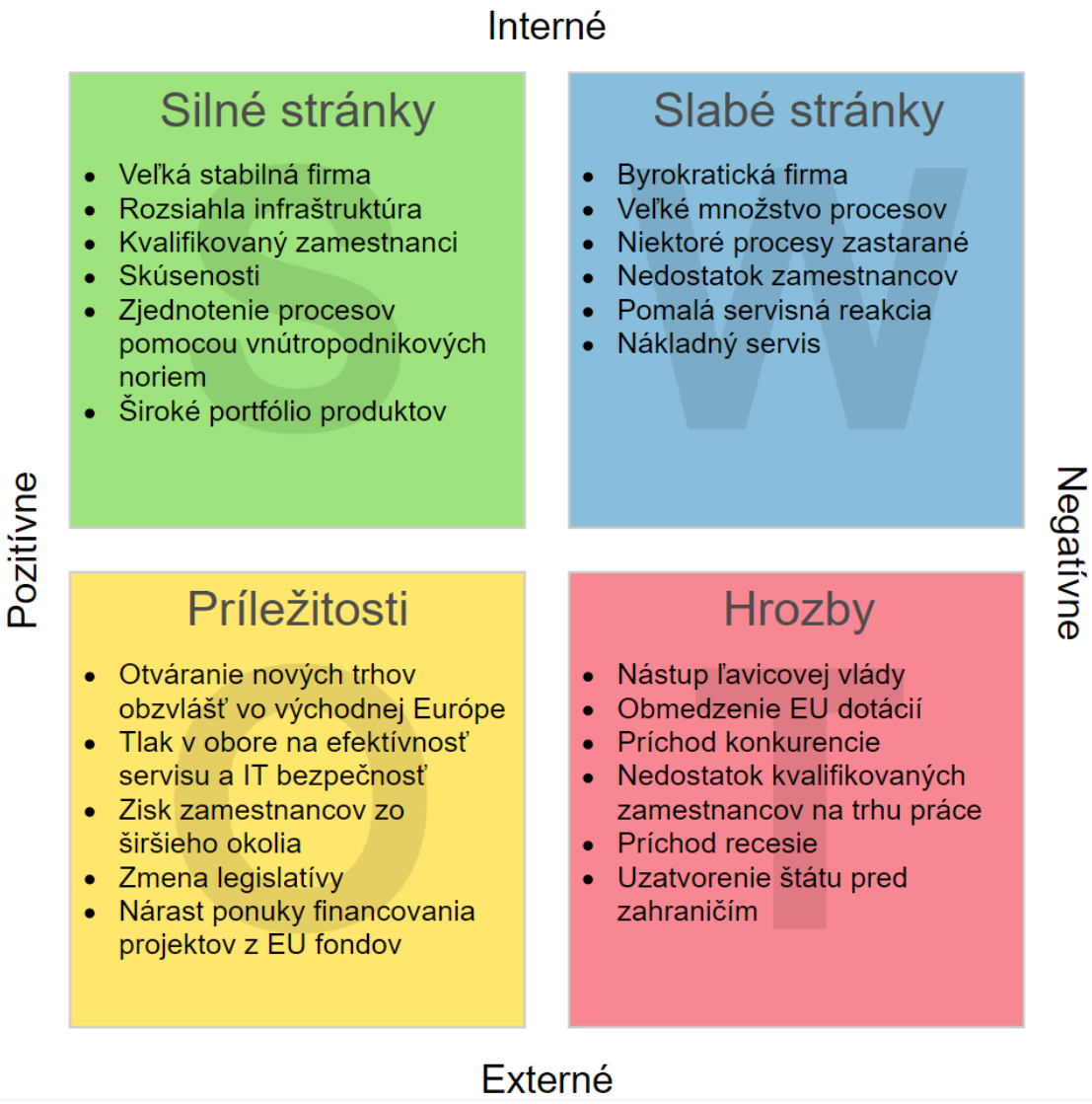
Zdielané hodnoty a ciele

Jeden z hlavných cieľov firmy je napísaný v úvode ich stránok "Bezpečně k cíli". Firma sa snaží poskytovať prvotriedne zabezpečovacie zariadenia, ktorých servisné procesy sú neoddeliteľnou súčasťou. V súčasnej dobe je trend zabezpečovania čoraz väčšieho množstva zariadení a aj s týmto súvisí potreba zmeny stratégie servisných procesov, aby tak prispeli k vyššej bezpečnosti.

V dnešnej dobe, ktorá je spoločensky komplikovaná je nutné mať jasne definovanú víziu a misiu firmy. U spoločnosti AŽD toto nie je v dostatočnom povedomí zamestnancov a chýba to aj na ich webových stránkach.

4.2.3 Vyhodnotenie pomocou SWOT analýzy

Nasleduje vyhodnotenie predchádzajúcich častí pomocou súhrnnej SWOT analýzy. Výsledkom bude rozhodnutie, či má cielená zmena zmysel a či je vhodné ju implementovať do prostredia firmy.



Obr. 4.2: SWOT analýza firmy AŽD Praha s.r.o.. Zdroj: Vlastná tvorba.

Z analýzy SWOT 4.2 vyplýva, silné stránky firmy čerpajú hlavne zo svojho postavenia na domácom trhu a dlhodobej tradície. Firma disponuje veľkou infraštruktúrou, skúsenosťami a širokým portfóliom produktov.

Medzi slabé stránky firmy môžeme zaradiť hlavne zdĺhavé a málo efektívne určité procesy, ktoré vyplývajú z rozvetvenej organizačnej štruktúry firmy. Isté procesy sú už zastaralé a nespĺňajú všetky požiadavky dnešnej modernej rýchlej doby, s čím sú spojené aj dodatočné náklady. Medzi tieto procesy zaraďujeme predovšetkým servisné procesy, ktorých modernizácia je hlavným predmetom tejto práce.

Keďže sa blíži čas volieb do parlamentu, je tu istá hrozba zvolenia vlády, ktorá by nebola orientovaná na potreby zamestnávateľov. S touto zmenou by mohla súvisieť aj väčšia izolovanosť štátu pred zahraničím, čo by firmu ovplyvnilo nepriaznivo. Momentálne je ekonomika v dobrej kondícií, čo spôsobuje nízku nezamestnanosť a obtiažnejšie oslovovanie kvalifikovaných zamestnancov. Ďalšou hrozbou môže byť príchod konkurencie. Existuje množstvo malých firiem, ktoré sa špecializujú len na určité oblasti, ktoré spadajú aj do portfólia firmy AŽD. Tieto menšie firmy majú výhodu väčšej flexibility, keďže nemajú také rozsiahle portfólio produktov ako firma AŽD. Zahraničná konkurencia firiem s podobným zameraním (Deutsche Bahn, Alstom a podobne), predstavuje taktiež isté ohrozenie.

Medzi príležitosti môžeme zaradiť lepšie čerpanie fondov EÚ, zmeny legislatívy či vznik pravicovo orientovanej vlády, zameranej na zamestnávateľov. Príležitosti, ktoré by mohli zásadnejšie eliminovať niektoré z hrozieb sú zlepšenie servisných procesov, čo by mohlo znamenať rýchlejšie reagovanie na neočakávané udalosti. Pre aplikáciu týchto zmien by bola využitá kvalitná rozsiahla infraštruktúra firmy AŽD, skúsenosti z rôznymi už zavedenými zmenami a množstvom kvalifikovaného personálu schopného tieto zmeny učiniť. Táto zmena by mohla byť prínosom aj v oblasti efektívneho využívania zamestnancov. Nasadenie ADS by uvoľnilo určitý počet zamestnancov monitorujúcich zariadenia, ktorí by mohli byť využití pri iných servisných činnostiach.

4.3 Analýza chybovosti zariadení

Ako bolo spomenuté v predchádzajúcej sekcii 4.1.3, tak spoločnosť AŽD spravuje niekoľko desiatok až stoviek riadiacich jednotiek a iných zariadení, ktoré potrebujú byť monitorované. V prvom rade je potrebné analyzovať, aké chyby môžu na daných zariadeniach nastať a či bude možné ich následne monitorovacím systémom detekovať a poslať technikom na spracovanie.

Táto sekcia bude preto zameraná na analyzovanie štruktúry chýb sledovaných zariadení, ich riešení a dopadu na celkový systém. Nasledujúca analýza bola vytvorená v rámci jednej oblasti pozostávajúcej z 14 zariadení za obdobie 1 roku.

Typ závady	Riešenie	Počet
Závada výpadku vozidlového detektoru	vzdialený reštart chybného detektoru	3
Závada riadiacej jednotky	výmena vadného zdroja, reštart modulu GPS, vzdialený reštart riadiacej jednotky	16
Údržba/Kalibrácia	predstavuje pravidelné servisné úkony	4
Zlý export z radaru	predanie diagnostike na vývojové oddelenie	7
Porucha napájania	riešenie chyby v spolupráci s poskytovateľom dátových a energetických služieb	5
Zlé spojenie so serverom	úprava nastevení cez vzdialený prístup a upgrade firewallu	2
Problém s meničom	výmena meničov/batérií	4
Ostatné	podla typu závady vzdialene alebo servisným výjazdom	4
Závady celkom		**45**

Tabuľka 4.1: Tabuľka zobrazujúca chybovosť sledovaných zariadení za obdobie 1.10.2017 - 30.9.2018. Zdroj: Vlastná tvorba.

Z tabuľky 4.1 vyplýva, že minimálne 3/4 chýb vyskytujúcich sa na sledovaných zariadeniach by bolo možné monitorovať centralizovane pomocou inteligentného dohľadového systému, ktorý dokáže včas upozorniť na blížiaci sa problém na danom zariadení.

4.4 Súčastný dohľad zariadení

V súčastnosti v sledovanej spoločnosti AŽD prebieha monitorovanie nasadených zariadení v rámci divízie DAST (4.1.3) pomocou tímu dispečerov, ktorí sa postupne "ručne" pripájajú k jednotlivým zariadeniam pomocou IP adresy a skontrolujú všetky potrebné ukazovatele. V prípade neočakávaného stavu zapíšu záznam chyby a pošlú ho tímu technikov, ktorý daný problém vyrieši buď priamo na mieste alebo vzdialene cez počítač. Záznamy o vzniku a riešení porúch, ako aj štatistiky stavov sledovaných zariadení sú vedené zvlášť v súboroch typu excel.

Na základe tohoto neefektívneho zdĺhavého procesu monitorovania a vyhodnocovania stavu zariadení vznikla potreba zo strany divízie DAST na vytvorenie ADS. Nový systém by mal byť centralizovaný - zhromažďovať všetky dôležité dáta pre monitoring na jednom mieste, ktorý by posielal notifikácie pri vzniku nejakého problému.

4.5 Dohľadové systémy

Závislosť dnešných spoločností od informačných technológií stále narastá a tak vzrastá aj závislosť na bezpečnej a spoľahlivej prevádzke jednotlivých zariadení tvoriacich informačný systém. Napríklad taký výpadok produkčného systému riadiaceho výrobu znamená aj zastavenie výroby a tým nemalé ekonomické následky.

Úlohou dohľadového systému je zbierať informácie z požadovaných zariadení, tieto informácie spracovávať a včas, správnym spôsobom interpretovať dohľadovým technikom, aby boli schopní v prípade potreby promptne zareagovať a tým minimalizovať škody vzniknuté napríklad výpadkom spojenia so zariadeniami alebo ich nesprávnou funkciou.

Ako bolo spomínané v predchádzajúcej sekcii 4.1.3, spomínaná spoločnosť vyrába, distribuuje a spravuje niekoľko typov zariadení (signalizačných radičov, verejného osvetlenia atď.), ktoré je potrebné efektívne monitorovať. V nasledujúcej časti budú popísané a zhodnotené použiteľné dohľadové systémy.

4.6 Analýza efektívnosti servisných procesov pomocou metódy HOS

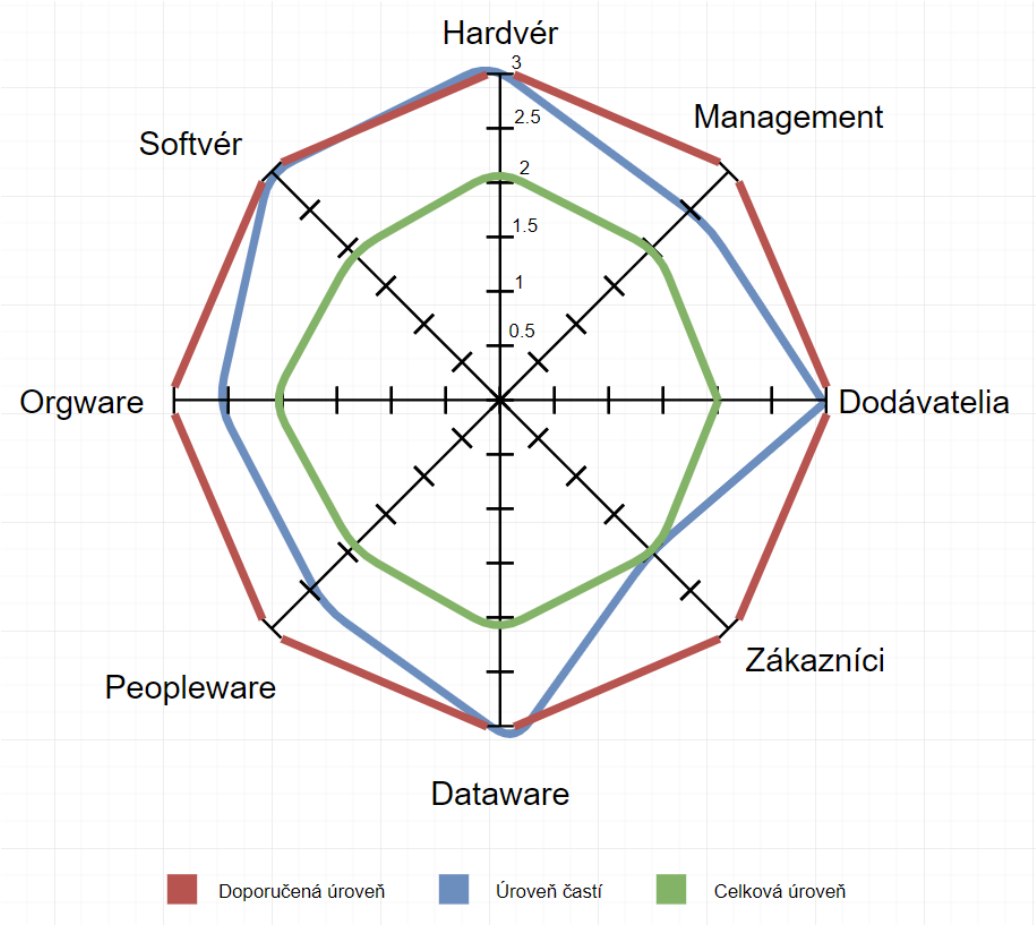
Dôležitým krokom pri výbere alebo nasadení ľubovoľného systému v prostredí firmy je analýza súčasného stavu, identifikácia nedostatkov, ich odstránenie a zabezpečenie požadovanej úrovni efektívnosti danej firmy. Z tohoto dôvodu prebehla analýza efektívnosti servisných procesov firmy AŽD, ktorá je predmetom nasadenia dohľadového informačného systému. Analýza bola vytvorená na základe metódy HOS popísanej v kapitole 3.1.1.

V prvej fáze analýzy prebehli stretnutia s kľúčovými osobami servisných procesov, ktoré majú na celý systém najväčší vplyv podľa HOS. Informácie o dodávateľoch a zákazníkoch boli získané od osoby zodpovednej za prevádzku a servis zariadení divízie. Hardware, software, orgware a dataware bol prekonzultovaný s vedúcim oddelenia informačných technológií a posledné dve časti peopleware a management s hlavným projektovým manažérom divízie.

Medzi hlavné nedostatky môžeme zaradiť istú mieru nespokojnosti zákazníkov [1], vyplývajúcu predovšetkým z pomalejších reakcií na servisné udalosti. Toto môže zvýšiť užitočnú hodnotu produktu, ktorého nedeliteľnou súčasťou je servisná podpora. Čo sa týka hardvéru

[1] Na základe interných štatistík vedených divíziou DAST.

a softvéru je divízia vybavená nadštandardne, čo môže byť menej efektívne vzhľadom na nedostatok servisných technikov a ďalších zamestnancov IT oddelenia. Dostupnosť dát, ich správa a bezpečnosť je na veľmi vysokej úrovni, čomu zodpovedajú aj štatistiky vedené ohľadne dostupnosti internetového pripojenia a služieb či zabezpečenia dát. Firma taktiež disponuje rozsiahlym súborom vnútorných organizačných noriem pokrývajúcich väčšinu činností a bezpečnosti pri nich. Firma má dlhoročnú tradíciu medzi poskytovateľmi zabezpečovacích zariadení a to sa podpísalo aj na aktuálnej skostnatenej byrokratickej štruktúre firmy a procesoch potrebných pri každej činnosti, či sa jedná už o nový produkt alebo nejakú zmenu vo firme. Čiže z manažérskeho pohľadu nejaké smerovanie divízie, určovanie stratégií alebo reagovanie na nové trendy nie je úplne jednoduché a priamočiare, vyžaduje si veľa rôznych procesov a byrokracie.



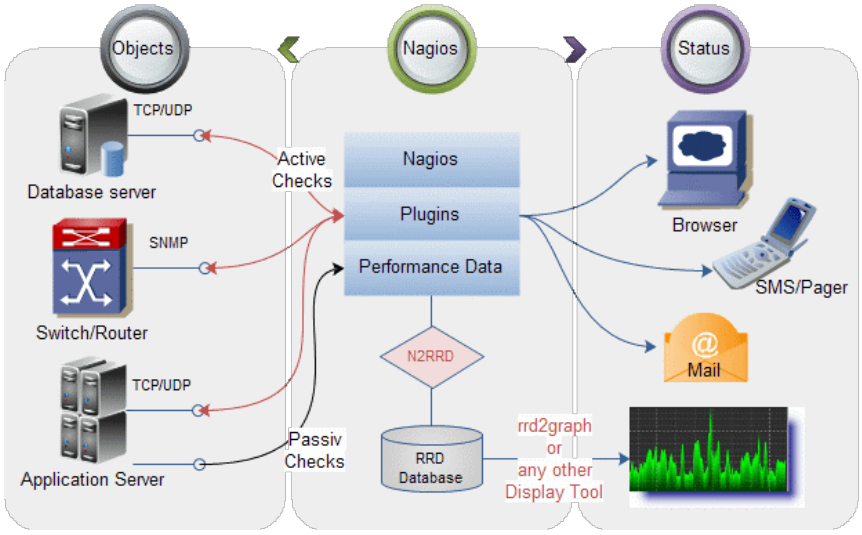
Obr. 4.3: Graf posúdenia efektívnosti divízie DAST podľa metódy HOS. Zdroj: Vlastná tvorba.

Na základe získaných informácií bol vytvorený kvalifikovaný odhad efektívnosti jednotlivých domén HOS metódy. Tieto výsledky boli zakreslené do grafu číslo 4.3. Ako je možné vidieť, jednou z hlavných oblastí, kde je priestor na zlepšenie je oblasť zákazníkov. Túto oblasť môžeme pozitívne ovplyvniť nasadením ADS, ktorý nám umožní zrýchliť reakcie na vzniknuté servisné problémy, ktoré môžu technici rýchlejšie odstrániť už pri ich zárodku. Ďalšou výhodou môže byť úspora zamestnancov aktuálne sa venujúcich dohľadu a využiť ich na chýbajúcich pozíciách v rámci divízie. Čo sa týka orgware, je tu priestor pre aktualizáciu organizačných noriem týkajúcich sa servisu. Všetky tieto spomínané opatrenia nám môžu pomôcť zefektívniť celkové fungovanie servisných procesov.

V nasledujúcej časti sú predstavené rôzne alternatívne riešenia ADS. Tieto systémy boli vybrané na základe analýzy vhodných dostupných dohľadových systémov, ktoré sú overené používateľmi, majú dobre spracovanú dokumentáciu a jedná sa o riešenia, ktoré sú na trhu dostupné dlhodobo, takže existuje vysoká miera pravdepodobnosti ich budúceho rozvoja a udržania sa na vysokej úrovni poskytovaných služieb. Jeden z nich bude vybraný na integráciu v prostredí firmy. Pri konečnom výbere bude braná do úvahy ich technologická úroveň, vhodnosť pre použitie, ktoré je cieľom tejto práce, a v neposlednej rade taktiež náklady spojené s ich nasadením a prevádzkou. Nasledujúce kapitoly teda charakterizujú vybrané automatizované dohľadové systémy.

4.6.1 Nagios XI

Nagios je rozsiahly nástroj na monitorovanie všetkých kritických častí IT infraštruktúry – aplikácie, služby, operačné systémy a iné. Poskytuje centrálny pohľad na jednotlivé sledované zariadenia pomocou dashboardu. Webové rozhranie poskytuje veľké množstvo rôznych pohľadov, sumárov a grafov pre zobrazenie informácií o monitorovaných zariadeniach. Systém Nagios ponúka možnosť pasívneho čakania na dáta zo sledovaných zariadení alebo ich aktívny dopyt pomocou Nagios agenta nainštalovaného na jednotlivých zariadeniach. [23]



Obr. 4.4: Štruktúra systému Nagios XI. Zdroj: [32].

Na obrázku číslo 4.4 je zobrazená základná štruktúra systému Nagios XI. Systém je možné logicky rozdeliť na nasledujúce časti:

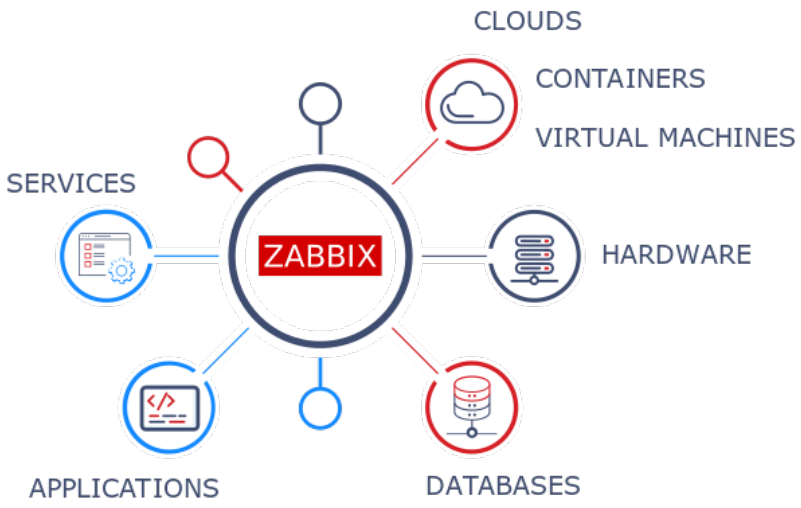
- objects - Monitorované objekty systémom Nagios (DB, Routers, Switches, Apps a podobne). Podporované sú rôzne protokoly ako napríklad TCP, UDP, SNMP. Monitorovanie prebieha buď pasívne, kedy sú dáta v pravidelných intervaloch odosielané na server alebo aktívne, kde si ich server priamo vyžaduje.
- Nagios - Hlavná časť, ktorá riadi systém, zhromažďuje výkonnostné dáta a ukladá ich do RRD (Round Robin Databáza) databáze pomocou N2RRD (prenos Nagios dát do RRD) nástroja.
- status - Koncové zariadenia, na ktorých je možné sledovať interpretované dáta. Patria sem prehliadače, mobilné zariadenia, sms alebo e-mailové notifikácie na udalosti atď.

Systémové požiadavky systému Nagios sú nasledovné:
HDD - 20GB
RAM - 2GB
CPU - Dual core 2.4 GHz
OS - CentOS, Redhat, Ubuntu, Debian
DB - MySQL/MariaDB

Cena: Zdarma do 7 monitorovaných zariadení (hostov), inak od 1 995$

4.6.2 Zabbix

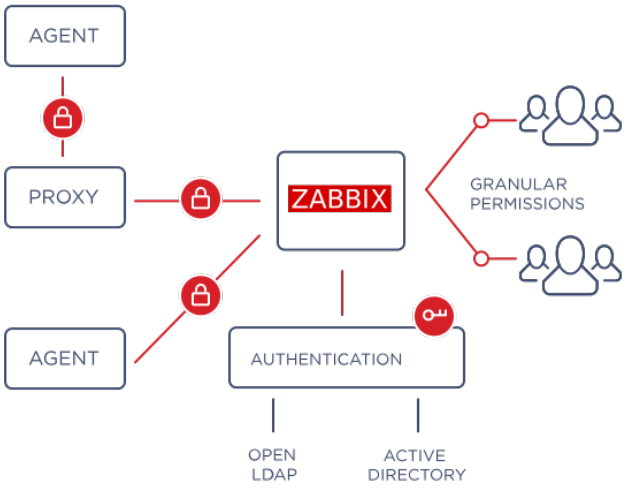
Zabbix je vysoko výkonný softvér navrhnutý na monitorovanie dostupnosti a výkonu všetkých prvkov sieťovej infraštruktúry. Zabbix umožňuje zber neobmedzených typov dát zo sieti. Jeho vysoký výkon umožňuje v reálnom čase monitorovať desaťtisíce serverov, virtuálnych PC a ostatných sieťových prvkov simultánne. Zabbix taktiež ponúka popri možnosti zbierania dát aj ich vizualizáciu. Ďalšou z možností je použitie Zabbixu pre monitorovanie webu. Licencia systému Zabbix spadá pod „open source“, preto sa stáva dostupným pre každého zadarmo.



Obr. 4.5: Výber možností využitia Zabbix systému. Zdroj: [20].

.

Na obrázku číslo 4.5 možno vidieť širokú škálu monitorovateľných zariadení. Zabbix umožňuje dohľad od hardvéru až po konečné aplikácie. Samozrejmosťou je dohľad databáz, virtuálnych strojov, cloudových riešení a rôznych senzorov.



Obr. 4.6: Zabezpečenie systému Zabbix. Zdroj: [20].

Diagram zabezpečenia systému Zabbix je zobrazený na obrázku číslo 4.6. Systém Zabbix umožňuje vytvoriť užívateľov alebo skupiny užívateľov s rôznymi právami prístupu a tak zamedziť alebo povoliť prístup k citlivým dátam. Na prihlásenie sa do systému je k dispozícií niekoľko autentifikačných metód akými sú LDAP (Lightweight Directory Access Protocol), HTTP alebo interná databáza. Samozrejmosťou je silné šifrovanie dát preposielaných medzi komponentami Zabbix systému. [24]

Význam dôležitých termínov používaných v Zabbixe

- Host - Sieťové zariadenie monitorované Zabbixom
- Item - Konkrétny údaj, ktorý Zabbix prijíma z hostu
- Trigger - Logický výraz, ktorý definuje hranicu problému a používa sa na vyhodnotenie dát získaných z itemov
- Graph - Slúži na vizualizáciu dát z itemov
- Screen - Slúži na zoskupovanie informácií (grafy, textové informácie, . . .) z viacerých zdrojov
- Map - Vizualizácie infraštruktúry danej siete

Štruktúra systému Zabbix

Systém Zabbix je tvorený z viacerých častí. Tými hlavnými sú Zabbix server, Zabbix agenti a webové rozhranie Zabbixu.

Zabbix server

Jedná sa o hlavný proces, ktorý všetko riadi, vyžaduje, odchytáva, zhromažďuje a spracováva dáta. Tento server je centrálnym úložiskom, v ktorom sú uložené všetky konfigurácie, štatistické a prevádzkové dáta. Všetky konfiguračné dáta sú uložené v mysql databázy

Zabbix agent

Je to klientský proces. Zabbix agent je implementovaný v jazyku C a môže bežať na rôznych platformách (napr. Linux, Unix alebo Windows). Zbiera miestne dáta (info o pevných diskoch, CPU, ...) z hostov a posiela ich na Zabbix server. Podporované sú dva prístupy vzhľadom na získavanie dát:

- pasívne (polling - server žiada o dáta agenta)
- aktívne (trapping - agent periodicky posiela aktuálne hodnoty serveru)

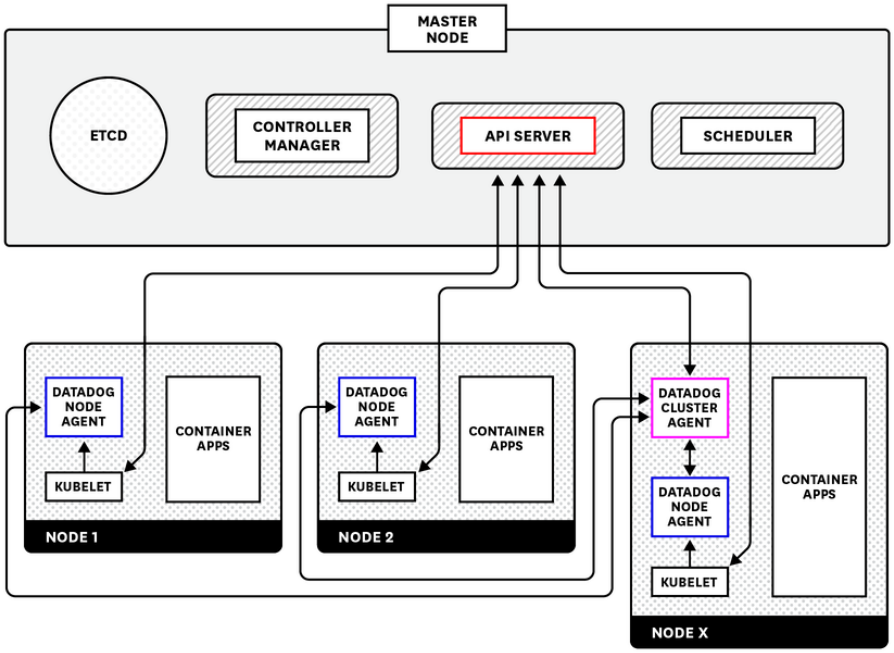
Systémové požiadavky systému Zabbix sú nasledovné (príklad pre monitorovanie cca 200 zariadení):
RAM - 2GB
CPU - Dual core
OS - Linux, IBM AIX, FreeBSD, NetBSD, OpenBSD, HP-UX, Mac OS X, Solaris
DB - MySQL, PostgreSQL, Oracle, IBM DB2, SQLite

Cena: Zabbix server a agent je distribuovaný pod licenciou GNU (General Public License), čiže je jeho inštalácia a používanie zdarma. [24]

4.6.3 Datadog

Datagog monitorovací systém je založený na cloudovom riešení serverovej časti a datadog agentoch na sledovaných zariadeniach. Systém podporuje monitorovanie širokej škály zariadení, aplikácií a služieb. Agenti podporujú odosielanie logov aplikácií alebo služieb na server, kde je možné tieto údaje filtrovať a analyzovať. Na strane cloudu prebieha vyhodnocovanie dát z agentov zaistené pomocou moderných technológií – prijaté správy sú uchovávané pomocou systému Kafka, z ktorého sú realtime spracovávané pomocou map-reduce operácií a

ukladané do noSQL databázy Cassandra. Na nasledujúcom obrázku je znázornený príklad architektúry použitia datadog agenta[9]:



Obr. 4.7: Architektúra využívajúca cluster datadog agenta komunikujúceho s priradenými node agentami. Na hlavnom stroji sú potom prístupné všetky *Kubernetes cluster* (Set strojov, na ktorých bežia kontajnerové aplikácie.) informácie. Zdroj: [9].

Systémové požiadavky systému Datadog nie sú žiadne, keďže je riešený formou cloudu na strane dodávateľa.
Cena: od 15$ za host na mesiac

4.6.4 Komparatívna analýza vybraných dohľadových systémov

V predchádzajúcej časti práce bolo spomenutých niekoľko známych systémov podporujúcich automatizovaný dohľad nad rozličnými zariadeniami. Pre dosiahnutie vytýčených cieľov je dôležitým krokom výber toho správneho dohľadového systému. Výberu jedného z analyzovaných systémov predchádza nasledujúca komparatívna analýza inšpirovaná SWOT analýzou dohľadových systémov.

Silné stránky

Pre všetky porovnávané systémy platí, že sa jedná o overené systémy veľkým počtom užívateľov. Ďalšou silnou stránkou týchto riešení je, že sú multiplatformné a teda môžu byť spustené takmer na ľubovoľnej platforme a to s veľmi nízkymi nárokmi na systémové požiadavky. Výborne je taktiež spracovaná užívateľská dokumentácia.

Silnou stránkou systému Datadog je možnosť efektívnej práce s obrovským množstvom logovacích súborov.

Silnou stránkou systému Zabbix je jeho nulová cena za používanie. Veľmi dôležitou silnou stránkou systému Zabbix je existujúca skúsenosť technikov firmy AŽD s týmto systémom, čo môže výrazne znížiť náklady na zaškolenie budúcich používateľov dohľadového systému.

Slabé stránky

Slabšou stránkou systémov Nagios XI a Datadog sú nutné náklady spojené s ich nasadením v rámci infraštruktúry firmy, keďže nie sú zadarmo a doterajšie skúsenosti zamestnancov s ich nasadením a prevádzkou sú nulové.

Medzi slabé stránky vybraných dohľadových systémov môžeme zaradiť skutočnosť, že sa jedná o relatívne mladé systémy a z toho vyplývajúce náklady na rôzne zmeny a reakcie na ich vývoj.

Príležitosti

Všetky porovnávané systémy ponúkajú zvýšenie spoľahlivosti dohliadaných zariadení. To nám umožňuje dosiahnuť rýchlejšiu reakciu na neočakávané udalosti akými môže byť výpadok energie, chyba zariadení alebo ich odcudzenie.

Medzi príležitosti systému Zabbix možno zaradiť skúsenosť inej divízie AŽD s týmto systémom, čo by bolo možné využiť pre hladké nasadenie a rýchle školenie budúcich používateľov dohľadového systému Zabbix.

Hrozby

Keďže sa jedná o často používané systémy, je tu istá hrozba útoku na tieto systémy, z čoho vyplýva tlak na ich zabezpečenie. Pri systéme Datadog môže byť toto nebezpečenstvo umocnené skutočnosťou, že Datadog využíva cloudové riešenie dohľadového systému. Ďalej je potrebné získať kvalifikovaných technikov na obsluhu dohľadového systému. Nutnosťou je efektívne riešenie a odfiltrovanie falošných správ zo systému.

Kapitola 5

Návrh a implementácia vlastného riešenia

V nasledujúcej časti bude navrhnutá strategická zmena servisných procesov (konkrétne dohľadu zariadení) pomocou Lewinovho modelu. Prebehne identifikácia rizík a návrhy opatrení na ich elimináciu.

V ďalšej časti bude predstavený návrh riešenia ADS. Najprv bude navrhnutá architektúra celého systému. Nasledovať bude výber vhodných technológií pre realizáciu navrhnutého systému. Poslednou časťou bude implementácia navrhnutého riešenia pomocou zvolených technológií a postupov.

5.1 Lewinov model zavedenia systému automatizovaného dohľadu

Lewinov model dopomáha k riadenej zmene vo firme. Podľa Lewina sa proces realizácie úspešnej zmeny skladá z nasledujúcich troch fáz.

5.1.1 Fáza rozmrazenia

V prvej fáze rozmrazenia je dôležité si uvedomiť, že súčasná situácia je ďalej neúnosná a vyžaduje zmenu. Vo fáze rozmrazenia je potrebné identifikovať sily, ktoré túto zmenu iniciujú a rovnako sily, ktoré bránia vykonaniu zmeny.

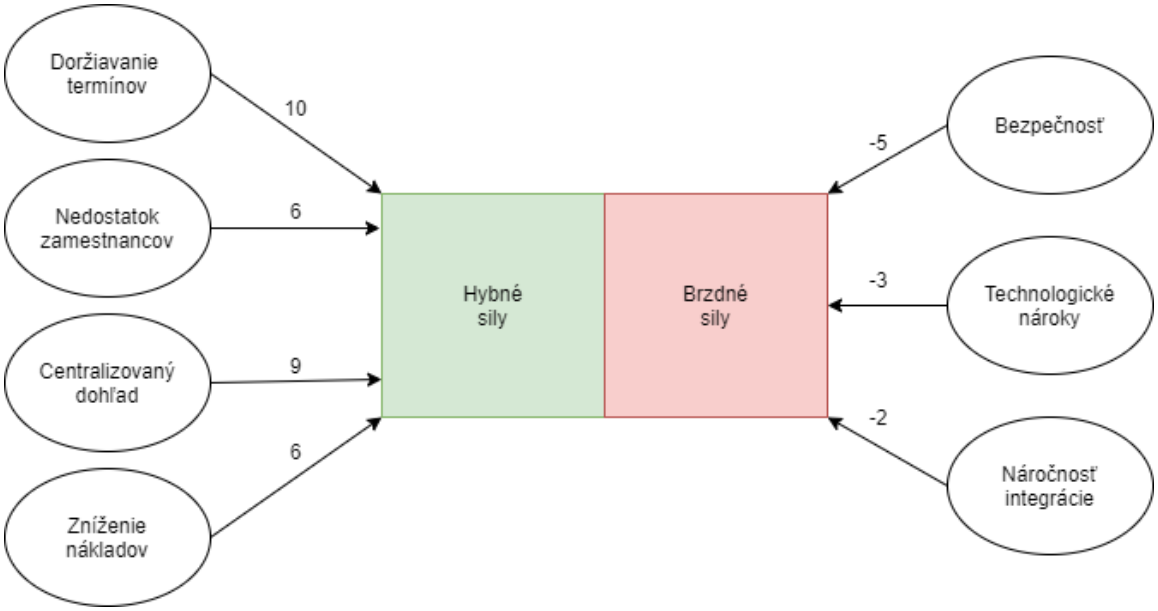
Hybné sily

- Náklady spojené s dodržiavaním zazmluvnených servisných termínov - Túto silu môžeme nazvať hlavnou hybnou silou. Príliš veľké množstvo zariadení je spravované nedostatkom zamestnancov, čo spôsobuje dlhšie reakčné časy na servisné udalosti. Čím je rekčná doba dlhšia, tým sa zvyšujú náklady spojené s nápravou.
- Nedostatok kvalifikovaných servisných zamestnancov - Ekonomika napreduje, nezamestnanosť klesá na historické minimá a osloviť kvalifikovaných zamestnancov je čím ďalej náročnejšie.
- Centralizovaný dohľad nad zariadeniami - Rôzne typy zariadení sa dohliadajú inými spôsobmi, čo vytvára priestor pre chaos hlavne novo zaúčaných pracovníkov. Centralizovaný pohľad na všetky typy zariadení uľahčí prácu zamestnancom zodpovedným za dohľad a zníži chybovosť.
- Zníženie nákladov na dohľad - Menej zamestnancov potrebných na dohľad, znamená aj ušetrenie nákladov na ich mzdy a je tu možnosť ich využitia v iných servisných procesoch, kde je rovnako nedostatok zamestnancov.

Brzdné sily

- Obhájenie nákladov na zmenu (nový server, implementácia a podobne).
- Bezpečnosť systému - Jednou z hlavných brzdných síl je obava vedenia z narušenia bezpečnosti nasadeného dohľadového systému.
- Osvojenie nových postupov, ktoré zamestnanci neradi menia.
- Náročnosť integrácie do prostredia firmy.

Na nasledujúcom obrázku 5.1 sú zobrazené všetky sily pôsobiace na zmenu servisných procesov. Po rozhovore s kľúčovými osobami boli na základe kvalifikovaného odhadu priradené váhy jednotlivých síl z intervalu <-10, 10> určujúce vplyv na zavedenie/nezavedenie zmeny, kde 10 znamená najvyššiu podporu pre zmenu a -10 najväčší odpor voči zmene.



Obr. 5.1: Zobrazenie síl pôsobiacich pre zmenu servisných procesov. Zdroj: Vlastná tvorba.

Ako je možné vidieť na obrázku síl 5.1, tak po sčítaní váh jednotlivých síl jasne prevládajú hybné sily, ktoré podporujú návrh zmeny stratégie určitých servisných krokov, medzi ktoré patrí hlavne dohľad zariadení.

Ďalším krokom podporujúcim rozmrazenie je podpora hnacích a eliminácia brzdných síl. Aj keď hnacie sily jasne vedú nad brzdnými, vypracovaním štúdie možných riešení zabezpečenia požadovaného ADS, môžeme aj tieto minimálne brzdiace sily oslabiť až odstrániť. Stačí navrhnúť a predstaviť systém, ktorý spĺňa vysoké bezpečnostné požiadavky, jeho integrácia nie je náročná a možno využiť dostupné technické vybavenie firmy. Toto vybavenie by bolo vylepšené postupom času, kedy by sa systém u vedenia firmy osvedčil a bolo požadované jeho rozšírenie aj na ďalšie procesy firmy.

Nositelia zmeny

Po vysvetlení potreby zmeny a teda umocnení hybných síl (a eliminácie brzdných) je potreba určiť nositeľov zmeny - pracovníkov zodpovedných za realizáciu definovanej zmeny:

- agent zmeny - Agentom zmeny bude v našom prípade vedúci IT oddelenia príslušnej organizačnej jednotky (OJ), ktorý má dostatok kvalifikovaných zamestnancov na realizáciu požadovanej zmeny. Programátori z IT oddelenia budú mať na starosti samotnú implementáciu daného riešenia, s ktorým im budú pomáhať dohľadoví dis-

pečeri servisného oddelenia. Títo dispečeri majú potrebné informácie o dohliadaných zariadeniach, ktoré budú programátori potrebovať na integráciu do nového ADS.

- projektant zmeny - Architekt z radov IT oddelenia OJ, ktorý bude mať na starosti tvorenie návrhov jednotlivých dielov dohľadového systému. Podlieha priamo agentovi zmeny, ktorý následne dáva prednosť jednotlivým návrhom projektanta.

- sponzor zmeny - Sponzorom zmeny bude technický riaditeľ, pod ktorého spadá IT oddelenie (Agent zmeny). Bude mať na starosti schvaľovanie nákupu potrebných technológií.

- advokát zmeny - Advokátom zmeny bude oddelenie servisných technikov, ktorí majú rovnako záujem na zriadení centralizovaného dohľadu, ktorý im uľahčí servisné výjazdy.

Intervenčné oblasti

V nasledujúcej časti sa zameriame na oblasti, ktoré budú zmenou ovplyvnené. Zmena sa aplikuje na časť servisných úkonov, konkrétne dohliadania servisovaných zariadení firmy. Zmena bude znamenať nahradenie súčasného spôsobu dohľadu 4.4 systémom ADS, ktorý bude implementovať IT oddelenie firmy. Čo sa týka technického vybavenia, v prvej etape zmeny bude použitá prevažne dostupná infraštruktúra (Hardvér) firmy, ktorá by mala postačovať na dohľad súčasných zariadení. Zmena ovplyvní najviac pracovníkov dohľadu (dispečerov), ktorí budú mať centralizovaný prehľad nad všetkými monitorovanými zariadeniami, čo im ušetrí čas a umožní rýchlejšie reagovať na vzniknuté situácie.

Ďalšou skupinou ľudí, ktorým zmena prinesie uľahčenie práce je technické oddelenie, ktoré má na starosti servisné výjazdy k zariadeniam. Včasným zareagovaním a možnou predikciou stavu zariadenia sa optimalizuje vykonávanie týchto servisných výjazdov.

5.1.2 Fáza zmeny

Po úspešnom rozmrazení aktuálneho stavu servisných procesov prichádza na rad fáza zmeny. Táto fáza zahrňuje všetko od počiatočného návrhu zmeny až po jej finálnu realizáciu, čiže nasadenie ADS do prostredia firmy. Súčasťou je časový harmonogram zavedenia odprezentovaný v ďalšej časti práce.

5.1.3 Fáza zmrazenia

Poslednou fázou je zmrazenie stavu po implementovaní požadovanej zmeny, ktorý je udržateľný v priebehu času. Účelom zamrazenia je teda stabilizovanie aktuálnej situácie dosiahnutej zavedením zmeny. Aby bola zmena udržateľná je potrebné neustále vzdelávanie zamestnancov spolu s celou organizáciou. Zavedená zmena je reflektovaná v aktualizovaných smerniciach, normách a nariadeniach, ktoré sú spolu so všetkými relevantnými dokumentami odoslané zamestnancom na oboznámenie a následne formou školenia prezentované.

5.2 Riziková politika

S každou zmenou je spojené určité riziko. Kľúčovou úlohou pri implementovaní zmeny je toto riziko v čo najväčšej miere eliminovať. V tejto kapitole sa preto zameriame na identifikovanie rizík, ich zakreslenie do mapy a navrhneme možné spôsoby ich znižovania.

5.2.1 Posúdenie rizík pomocou skórovacej metódy

Dopad identifikovaných rizík spojených so zmenou stratégie servisných procesov ohodnotíme pomocou skórovacej metódy. V nasledujúcej časti budú definované riziká v jednotlivých oblastiach. Za každý rizikom bude odhadnutá miera pravdepodobnosti výskytu rizika (P), dopadu rizika (D) a celková hodnota rizika (H), ktorá vznikne vynásobením H = P*D.

Technologická oblasť

- chyba softvéru ADS P=2, D=4, H=8
- nekompatibilita ADS s požadovanými zariadeniami na monitorovanie P=4, D=1, H=4
- výpadok hlavného serveru ADS P=4, D=4, H=16
- poškodenie/strata monitorovaných dát P=2, D=3, H=6
- narušenie bezpečnosti systému ADS (z vonka aj z vnútra) P=3, D=4, H=12
- nedostupnosť monitorovacích služieb P=3, D=3, H=9

Finančná oblasť

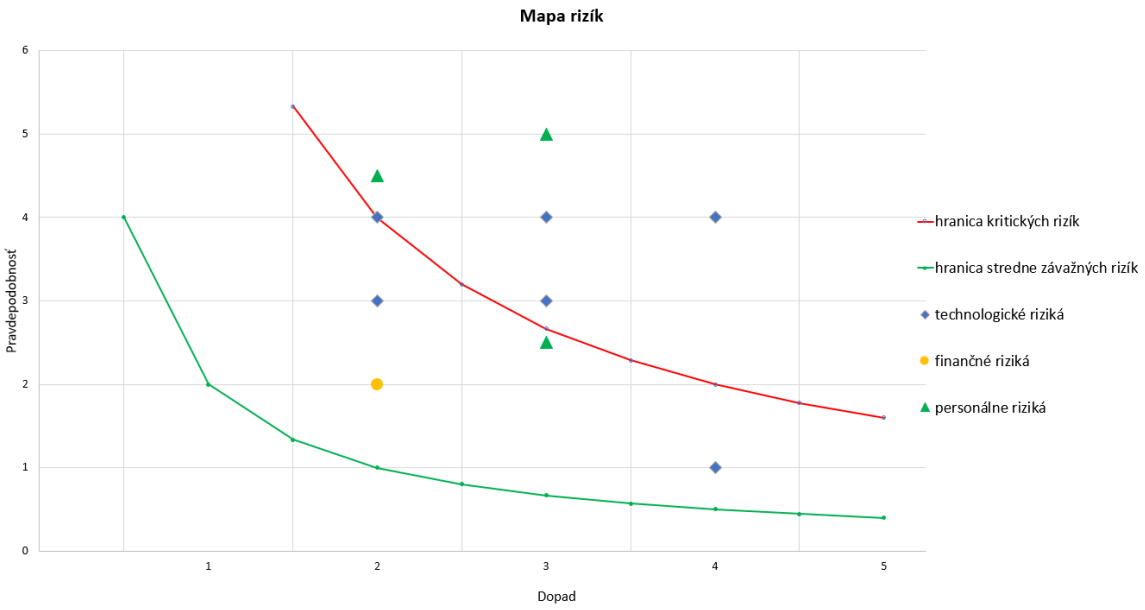
- nárast neočakávaných nákladov P=2, D=2, H=4

Personálna oblasť

- nedostatok kvalifikovaných zamestnancov P=3, D=5, H=15
- znížená pozornosť zamestnancov P=2, D=4.5, H=9
- počiatočná averzia z nového systému pre dohľad P=3, D=2.5, H=7.5

Z identifikovaných rizík v predchádzajúcej časti zostrojíme mapu rizík 5.2. Mapa je rozdelená na nasledovné časti:

- kritická (nad červenou krivkou) - Riziká v tejto kategórií závažným spôsobom ohrozujú systém a je nutné prijať opatrenia na ich zníženie.
- stredne závažná (medzi zelenou a červenou krivkou) - Sem patria riziká, ktorých výskyt spolu s dopadom nepredstavuje bezprostredné ohrozenie systému ale určitým spôsobom ho ovplyvňuje a preto je potrebné prijať opatrenia na zníženie týchto rizík.
- menej závažná (pod zelenou krivkou) - Najmenej závažné riziká, ktoré neohrozujú chod systému ale evidujeme ich.



Obr. 5.2: Mapa rizík. Zdroj: Vlastná tvorba.

5.2.2 Znižovanie rizík

V nasledujúcej časti budú popísané identifikované riziká a návrhy na ich znižovanie.

Chyba softvéru

Môže sa stať, že v programe nastane neočakávaná chyba, ktorá znefunkční časť alebo celý systém. Tento problém môže nastať napríklad po aktualizácií alebo nesprávnej konfigurácii systému. Ďalším možným zdrojom chybovosti môže byť program zodpovedný za napojenia jednotlivých typov dohliadaných zariadení.

Toto riziko možno eliminovať dôslednejšou kontrolou programátorov, ktorí majú implementáciu a nasadenie na starosti. Veľký dôraz bude kladený na testovanie softvéru od počiatočných fáz implementácie až po koniec jeho životného cyklu. Aktualizácia systému bude nasadená až po určitom čase, kedy používatelia potvrdia spoľahlivosť systému.

Nekompatibilita

V priebehu času môže nastať situácia, že v nejakej časti systému napríklad napojenia na monitorované zariadenie nastane nekompatibilita, ktorú je možno eliminovať používaním vhodných technológií, nad ktorými treba pouvažovať pred samotnou implementáciou systému. Ďalším prípadom môže byť požiadavka na monitorovanie zariadenia, ktorého výstup nie je kompatibilný s množinou typov podporovaných v dohľadvom systéme. Riešením bude implementácia konektora, ktorý vhodným spôsobom interpretuje daný typ dát.

Výpadok serveru

Riziko výpadku serveru je zo všetkých rizík najväčšie a jeho dopad je veľký, keďže server zaisťuje chod celého systému. Výpadok serveru môže nastať z rôznych príčin. Najčastejšou príčinou je výpadok elektrickej energie, medzi menej časté zaraďujeme rôzne chyby hardvéru alebo softvéru. Zabezpečenie alternatívneho zdroja energie je kľúčovým opatrením k zníženiu tohoto rizika. Na elimináciu rizika chybovosti hardvéru/softvéru je možnosť zaobstarania náhradného serveru s rovnakou konfiguráciou, ktorý je nasadený v prípade poruchy primárneho (Failover). Dopad nefunkčnosti serveru narastá s časom, počas ktorého je nedostupný, preto je potrebné zaistiť nahodenie sekundárneho zdroja/serveru v čo najkratšom možnom čase.

Poškodenie/strata dát

Strata dát patrí medzi stredne závažné riziká. Toto riziko súvisí hlavne z predchádzajúcim rizikom výpadku serveru, čiže znížením tohoto rizika sa priamo úmerne znižuje aj riziko straty dát. Ďalším účinným opatrením je zálohovanie dát v pravidelných intervaloch.

Narušenie bezpečnosti

Hrozba narušenia bezpečnosti je pomerne vysoká, preto je potrebné ju znížiť v oboch smeroch (P aj D). Pravdepodobnosť výskytu znížime zavedením povinnosťou bezpečnejších hesiel používateľov a použitím virtuálnej privátnej siete, v ktorej celý systém pobeží. Dopad zredukujeme spätným overením potreby zásahu (výjazdu technikov) a to starým postupom overenia funkčnosti zariadení, ktorý sme nahradili ADS.

Nedostupnosť služieb

Nedostupnosť služieb súvisí s výpadkom jednotlivých služieb na monitorovanom zariadení alebo celého zariadenia. Pravdepodobnosť tohoto rizika sa nedá veľmi znížiť, pretože aj kvôli týmto výpadkom uvažujeme nad zmenou servisných procesov. Avizovaná zmena môže dopomôcť k zníženiu dopadu tohoto výpadku pretože bude odhalený skôr a umožní tak technikom včas zareagovať.

Nárast nákladov

Toto riziko radíme medzi menej závažné. Nie je možné ho eliminovať, keďže súvisí hlavne s externými vplyvmi. Môžeme byť naň iba pripravení a to dostatočnou finančnou rezervou na pokrytie týchto nákladov. Tieto náklady by pre firmu nemali znamenať nejaké väčšie riziko.

Nedostatok kvalifikovaných zamestnancov

Nedostatok kvalifikovaných zamestnancov znamená výrazný problém, ktorý narúša chod celého systému. Firma musí byť schopná osloviť požadovaný počet zamestnancov rôznymi benefitmi a primeraným finančným ohodnotením. Jedným z cieľov implementácie tejto zmeny servisných procesov je aj zníženie počtu potrebných zamestnancov na dohľad a tak vhodným výberom systému a jeho nasadenia môžeme aj toto riziko nedostatku zamestnancov výrazne znížiť.

Znížená pozornosť zamestnancov

Zavedenie ADS môže viesť k zníženej pozornosti zamestnancov, ktorí boli zvyknutí všetko kontrolovať ručne. Je potrebné aby boli poriadne preškolení a vedomí si dôležitosti dozoru aj nad ADS. Zníženie pravdepodobnosti tohoto rizika je možné zavedením neohlásených falošných (kontrolných) problémov, ktoré preveria rýchlosť reakcie zamestnancov. Toto opatrenie je potrebné robiť v rozumnej miere, aby neovplyvnili reagovanie na skutočné problémy v negatívnom smere. Ďalším predpokladom zvýšenej pozornosti je správne

nastavenie užívateľského rozhrania, odosielanie notifikácií vo forme e-mailov, sms správ a podobne.

Počiatočná averzia zamestnancov k novému systému

Väčšina zamestnancov je zvyknutá na starý zabehnutý systém správy a monitorovania zariadení. Môže medzi nimi panovať z počiatku averzia k používaniu nového systému. Toto budeme schopní eliminovať počiatočným školením a vysvetlením toho, že ide hlavne o uľahčenie ich práce.

Napadnutie systému z vnútra

Systém môže byť terčom útoku aj z radov zamestnancov. Toto riziko sa týka všetkých už spomenutých rizík od výpadku serveru, cez poškodenie a stratu dát až po nárast nákladov na nápravu. Na elimináciu tohoto rizika je potrebné obmedziť prístup do ADS len autorizovaným používateľom, ktorých kroky v systéme sú podrobne logované. K tomuto opatreniu patrí povinnosť silného hesla, za ktoré nesie zodpovednosť zamestnanec, cez ktorého by k prípadnému narušeniu bezpečnosti prišlo.

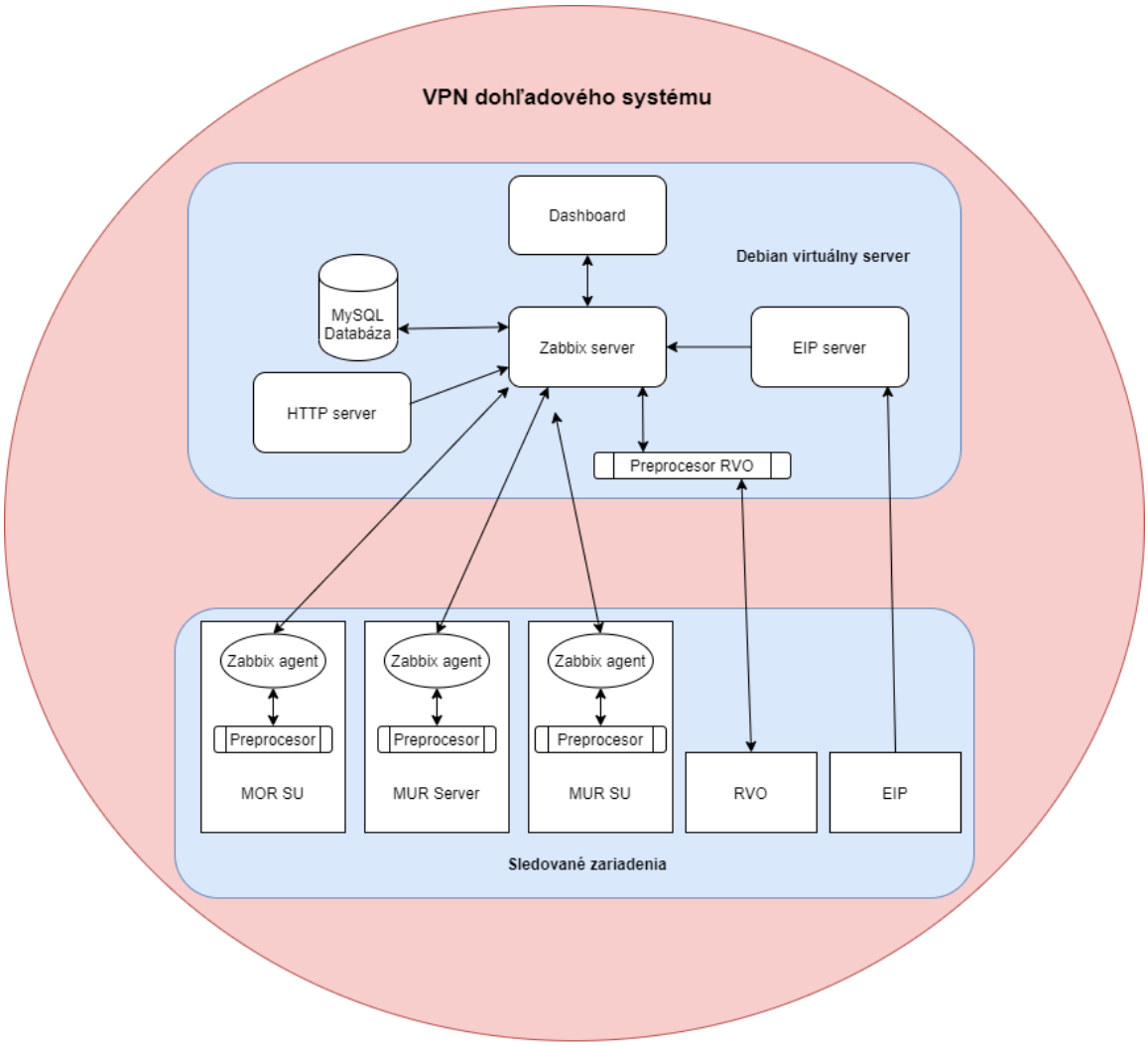
5.3 Návrh architektúry systému automatizovaného dohľadu

Základom funkčného a efektívneho informačného systému je dobrý návrh jeho architektúry. V tejto sekcii bude predstavený návrh architektúry ADS. Navrhnutú architektúru je možné vidieť na obrázku číslo 5.3.

Srdcom ADS bude monitorovací systém Zabbix, ktorý funguje na princípe architektúry klient-server, kde server bude umiestnený na našom virtuálnom serveri a klient (agent) nasadený na jednotlivých sledovaných zariadeniach. Na virtuálnom serveri bude ďalej bežať HTTP server, ktorý bude odchytávať chybové stavy posielané zariadeniami *riadiacimi verejné osvetlenie* (RVO). Bude tu umiestnená databáza MySQL na ukladanie dát zo sledovaných zariadení. Jednou z posledných hlavných častí virtuálneho serveru bude TCP server, ktorý bude prijímať, spracovávať a odosielať dáta z *elektronických informačných panelov* (EIP) na Zabbix server.

Dáta z jednotlivých sledovaných zariadení, ktoré budú zhromažďované na serveri, budú rôznych typov a budú získavané rozdielnymi spôsobmi. Z tohoto dôvodu bude potrebné navrhnúť a implementovať preprocesory, ktoré dané dáta premapujú do požadovaných formátov. Tieto preprocesory budú slúžiť ako medzivrstva medzi surovými (raw) dátami na

zariadeniach a Zabbix agentami, ktorí spracované dáta budú posielať na Zabbix server. Na zariadenia typu RVO nie je možné nainštalovať Zabbix agenta, preto je potrebné navrhnúť, implementovať a nasadiť RVO preprocesor na virtuálny server, ktorý bude priamo komunikovať s RVO jednotkami (pomocou Zabbix agenta virtuálneho serveru).



Obr. 5.3: Návrh architektúry ADS. Zdroj: Vlastná tvorba.

5.4 Návrh technológií potrebných pre systém automatizovaného dohľadu

V predchádzajúcej podkapitole 5.3 bola navrhnutá základná architektúra ADS. V tejto podkapitole sa preto budeme venovať výberu vhodných technológií pre realizáciu navrhnutého systému.

Základom ADS bude monitorovací systém Zabbix, ktorý sa skladá z dvoch základných častí - server a klient. Pre Zabbix server bol vybraný operačný systém Linux (Distribúcia Debian), na základe jeho priamej podpory systémom Zabbix formou inštalačného balíku. V prospech linuxového operačného systému Debian patrila taktiež jeho dostupnosť (používanie je zadarmo), podpora a ponúkaný výkon. Ďalšou potrebnou súčasťou systému Zabbix je databáza, ktorá bude obsahovať všetky dohliadané dáta. Na základe podpory Zabbixom a predchádzajúcimi skúsenosťami bola zvolená databáza MySQL.

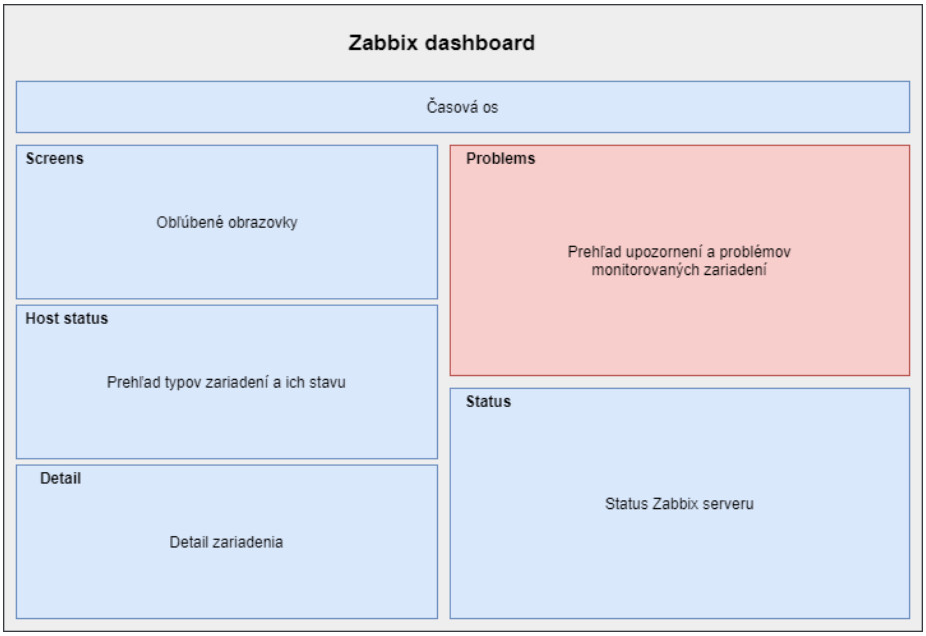
Z návrhu architektúry je zrejmé, že každé dohliadané zariadenie produkuje informácie iného typu, či formátu a získanie týchto dát je rovnako rozdielne. Preto je potrebné navrhnúť a implementovať preprocesory a servery schopné tieto dáta získať a previesť do požadovaného formátu, ktorému systém Zabbix rozumie a je schopný ho interpretovať používateľom. Na implementáciu týchto programov bol zvolený skriptovací jazyk Python. Tento jazyk bol zvolený hlavne pre jeho univerzálnosť a širokú škálu využitia. Keďže je dosť pravdepodobné, že do styku s preprocesormi prídu aj technici, ktorí nie sú programátormi, je potrebné aby bol zvolený jazyk jednoduchý a zrozumiteľný, čo je ďalším dôležitým špecifikom programovacieho jazyka Python.

Medzi komponentami dohľadového systému a monitorovanými zariadeniami prebieha predávanie citlivých dát. Hlavne z toho dôvodu je potrebné zabezpečenie proti odcudzeniu týchto informácií. Pre robustné zabezpečenie systému bolo na výber niekoľko možností ako firewall (softwarový alebo hardwarový), fyzicky oddelená sieť alebo virtuálna sieť. Na zabezpečenie systému bol zvolený koncept virtuálnej privátnej siete pomocou voľne dostupného softvéru OpenVPN, ako je možné vidieť na obrázku číslo 5.3. Pomocou VPN je medzi zariadeniami v dohľadovom systéme vytvorený takzvaný tunel, ktorým si zariadenia predávajú dáta v šifrovanej podobe. Ďalšie identifikované riziko možnej nekompatibility bude znížené použitím vhodného systému pre nasadzovanie nových verzií softvéru.

5.5 Návrh webového rozhrania systému

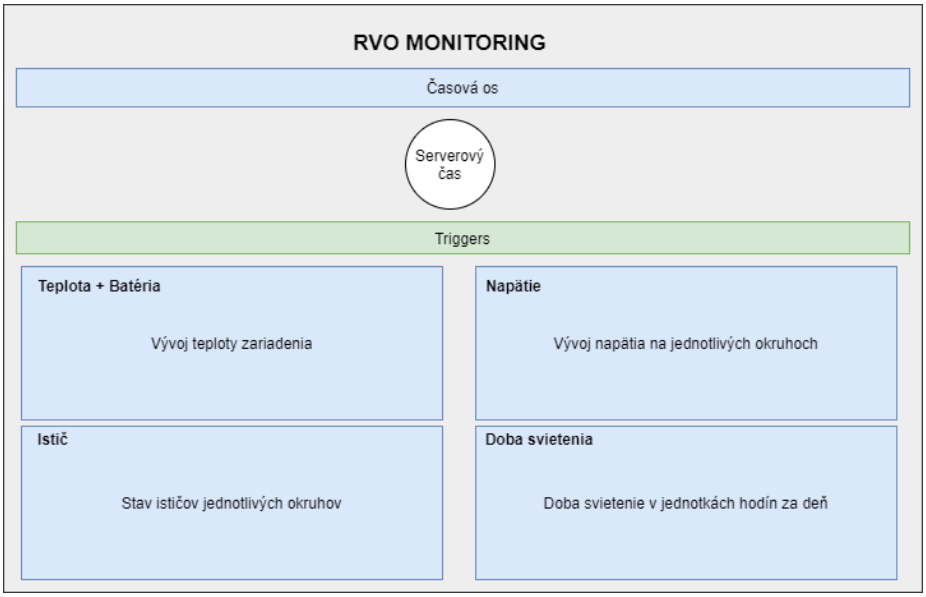
Ďalším potrebným krokom je návrh webového rozhrania dohľadového systému, ktoré bude slúžiť dispečerom na prehľad nad celým systémom. Základnou časťou webového rozhrania bude hlavná stránka nazývaná Dashboard. Dashboard slúži na zobrazenie dôležitých prevádzkových údajov o Zabbix serveri a vybraných informácií o dohliadaných zariadeniach. Na základe dostupných informácií zo Zabbix dokumentácie bol vytvorený v spolupráci s dispečermi návrh tejto stránky, ktorý je zobrazený na obrázku číslo 5.4. Dashboard je rozdelený na nasledujúce časti:

- problems - Hlavná časť dashboardu slúžiaca pre zobrazenie aktuálnych problémov na zariadeniach až do ich vyriešenia. Poskytuje informácie o čase vzniknutia problému, jeho popisu a odkazu na zariadenie, kde je možné získať podrobnejšie informácie o probléme. Ďalej informuje o vzniknutých udalostiach, ktoré boli vyvolané vznikom problému, medzi čo zaraďujeme napríklad odoslanie e-mailu nastaveným užívateľom.
- status - Obsahuje informácie o stave Zabbix serveru. Zobrazuje počet monitorovaných zariadení, celkový počet triggrov a ich stavov, počet online užívateľov a počet prijatých dát za jednotku času.
- screens - Slúži na jednoduché prechádzanie uložených obrazoviek.
- host status - Je zoznam pripojených typov zariadení, ich počet a aktuálny stav.
- detail - Sem je možné vložiť detail (graf) vybraného zariadenia, u ktorého je potrebné sledovať vývoj dát v čase.
- časová os - Slúži na zobrazovanie dát v zadanom časovom intervale a tak poskytuje možnosť prezerať aj historické dáta.



Obr. 5.4: Návrh dashboard obrazovky dohľadového systému. Zdroj: Vlastná tvorba.

Ďalej bolo potrebné navrhnúť obrazovku pre monitorovanie zariadení typu RVO. Výsledok návrhu je zobrazený na obrázku číslo 5.5.

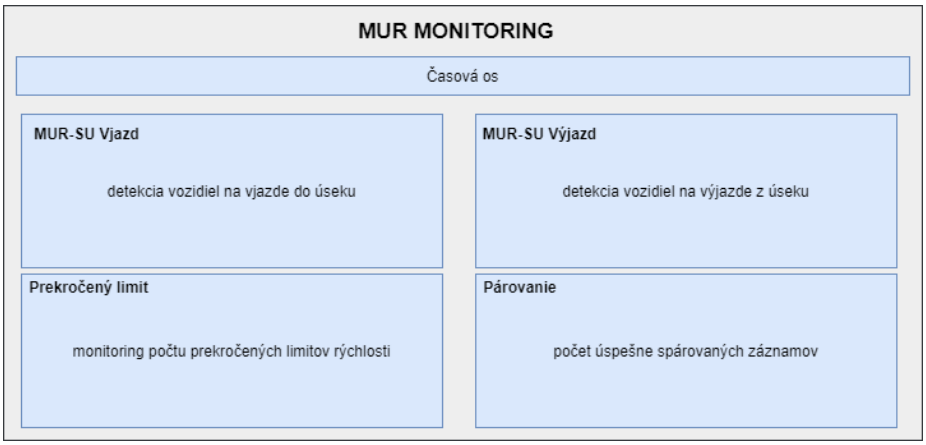


Obr. 5.5: Návrh obrazovky pre monitorovanie zariadení typu RVO. Zdroj: Vlastná tvorba.

Obrazovka monitoringu RVO pozostáva z nasledujúcich častí:

- časová os - Slúži na zobrazovanie dát v zadanom časovom intervale a tak poskytuje možnosť prezerať aj historické dáta.
- serverový čas - Aktívny webový prvok zobrazujúci presný čas serveru.
- triggers - Zobrazuje stav jednotlivých triggrov (spúšťačov udalostí) nastavených na RVO zariadení.
- teplota + Batéria - Zobrazuje vývoj teploty a napätia batérie zariadenia.
- istič - Zobrazuje stav ističov jednotlivých fáz, či sú zapnuté alebo vypnuté.
- napätie - Graf vývoja napätia jednotlivých fáz RVO a celkovej spotreby energie.
- doba svietenia - Zobrazuje celkový vývoj zapnutia osvetľovacích prvkov.

Výsledkom diskusie s dispečerskými pracovníkmi je návrh spoločnej obrazovky pre zariadenia monitoringu úsekovej rýchlosti (jednotkové stanovište SU-MUR a server MUR-Server) aby bolo jednoduchšie porovnávanie jednotlivých monitorovaných úsekov. Návrh je zobrazený na obrázku číslo 5.6.



Obr. 5.6: Návrh obrazovky pre monitorovanie zariadení typu SU-MUR a MUR-Server. Zdroj: Vlastná tvorba.

Popis častí návrhu MUR obrazovky je nasledovný:

- časová os - Slúži na zobrazovanie dát v danom čase a tak poskytuje možnosť prezerať aj historické dáta.

- SU-MUR Vjazd/Výjazd - Graf zobrazujúci detekované vozidlá na vjazde do respektíve výjazde z úseku. Farebne sú rozlíšené prípady úspešného (zelená) a neúspešného (červená) rozpoznania vozidla.
- prekročený limit - Zobrazuje počet vozidiel prekračujúcich stanovený rýchlostný limit.
- párovanie - Zobrazuje počet spárovaných záznamov z SU-MUR zariadení (to je takých, kde sa zhoduje identifikátor prejazdu vjazdového a výjazdového stanovišťa monitorovaného úseku).

5.6 Implementácia konektorov pre jednotlivé zariadenia

Nasledujúca kapitola bude obsahovať návrh konektorov pre jednotlivé zariadenia dohliadané pomocou ADS. Ďalšou časťou bude popis konkrétnej implementácie konektorov. Záverečnou časťou bude testovanie implementovaných konektorov a jeho vyhodnotenie. Príklad zdrojového kódu implementácie je možné vidieť v prílohe A.

5.6.1 Príprava systému Zabbix

Pred samotnou implementáciou jednotlivých konektorov, je potrebné správne nakonfigurovať časti dohľadového systému, ktoré budú komunikovať s týmito konektormi. Medzi základné časti systému popísané v časti 4.6.2 patria Zabbix server a Zabbix agenti. Po správnom nastavení konfiguračných súborov je overená funkčnosť komunikácie Zabbix server-agent. Ďalším dôležitým prvkom dohľadového systému je webové rozhranie. Po základnej konfigurácií je možné zobraziť informácie o serveroch, na ktorých bežia Zabbix agenti a tak isto aj o serveri, na ktorom beží samotný Zabbix aj s webovým rozhraním. Keď je všetko nastavené a otestovaná základná funkčnosť dohľadového systému, môžeme sa začať implementácia jednotlivých konektorov.

5.6.2 MUR - Monitoring úsekovej rýchlosti

Prvými zariadeniami, pre ktoré je potrebné vytvoriť konektor komunikujúci s dohľadovým systémom sú jednotky monitorovacie úsekovú rýchlosť. Monitoring úsekovej rýchlosti pozostáva z jednotiek SU-MUR a serveru, ku ktorému sú priradené.

SU-MUR

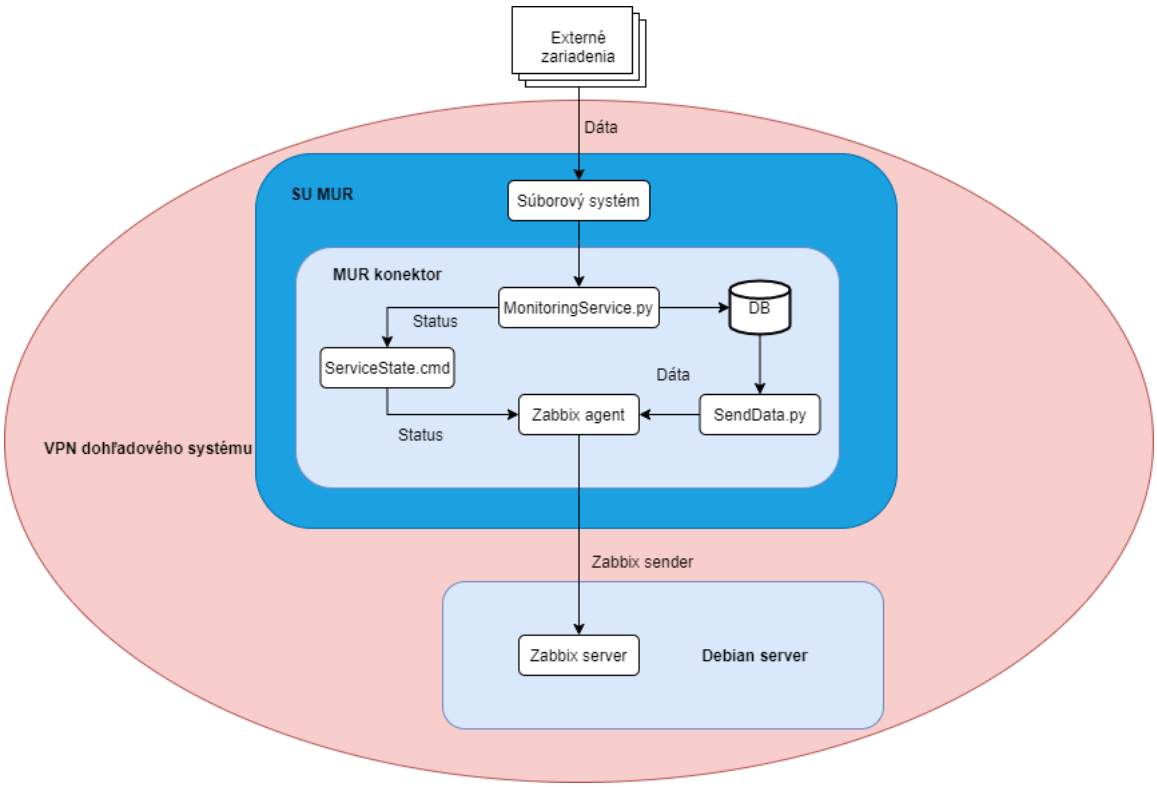
SU-MUR je zariadenie bežiace pod systémom Windows. Na jednom monitorovanom úseku sa nachádzajú dve jednotkové stanovištia SU-MUR - jedno na začiatku, druhé na konci. Zariadenia v pravidelných intervaloch zhromažďujú obrazové dáta získané z kamerových systémov umiestnených na cestnej komunikácií. Dáta sa ukladajú v adresárovej štruktúre systému Windows podľa identifikátora zariadenia, ktoré dáta vytvorilo, dátumu a času ich vytvorenia. Tieto dáta obsahujú hlavnú informáciu o správnom detekovaní cieľového vozidla, ktorého rýchlosť je na danom úseku monitorovaná. To, koľko vozidiel bolo správne rozpoznaných je jednou z hlavných požiadaviek na dohľadový systém.

V rámci konektora bol implementovaný Python skript MonitoringService.py, ktorý slúži na extrahovanie dát uložených v adresárovej štruktúre systému Windows a ich následné uloženie v lokálnej databáze. Dáta obsahujú informácie o počte rozpoznaných respektíve nerozpoznaných vozidiel za jednotku času. Tento skript je nasadený priamo na jednotke SU-MUR formou služby systému Windows, ktorá je automaticky spustená pri naštartovaní systému Windows. Na získanie dát uložených v lokálnej databáze vytvorenej službou MonitoringService.py slúži skript SendInfo.py. Vstupom skriptu je počet posledných záznamov a požadovaný typ pre spracovanie dát a to na základe ich využitia, či sa jedná o spracovanie pre textové alebo grafické zobrazenie v dohľadovom systéme. Skript SendInfo.py je využívaný Zabbix agentom, ktorý je na zariadení SU-MUR nasadený ako služba. V konfigurácii Zabbix agenta je vytvorený prístupový bod pre Zabbix server, pomocou ktorého sú odosielané definované dáta z agenta na server. Tento prístupový bod zabezpečí získanie zadaných dát pomocou volania skriptu SendInfo.py.

Za určitých okolností sa môže stať, že služba MonitoringService.py prestane bežať. Z tohoto dôvodu bol vytvorený skritp ServiceState.cmd (B), ktorý odosiela informácie o stave služby do Zabbix serveru pomocou Zabbix agenta, čo umožňuje dispečerom promptne zareagovať na vzniknutú situáciu. Diagram implementovaného konektora pre zariadenia SU-MUR je zobrazený na obrázku číslo 5.7.

Do dohľadového systému bol pridaný aj konektor na zariadenie monitorujúce okamžitú rýchlosť (MOR), ktorých princíp je podobný ako pri zariadeniach typu MUR (5.7), avšak dáta z externých zariadení sú ukladané do SQLite databáze. Skript na získavanie dát je mierne upravený MonitoringService.py, ktorý na rozdiel od MUR neprehľadáva súborový

systém ale databázu, ku ktorej cesta je definovaná v MOR konfiguračnom súbore formátu XML.



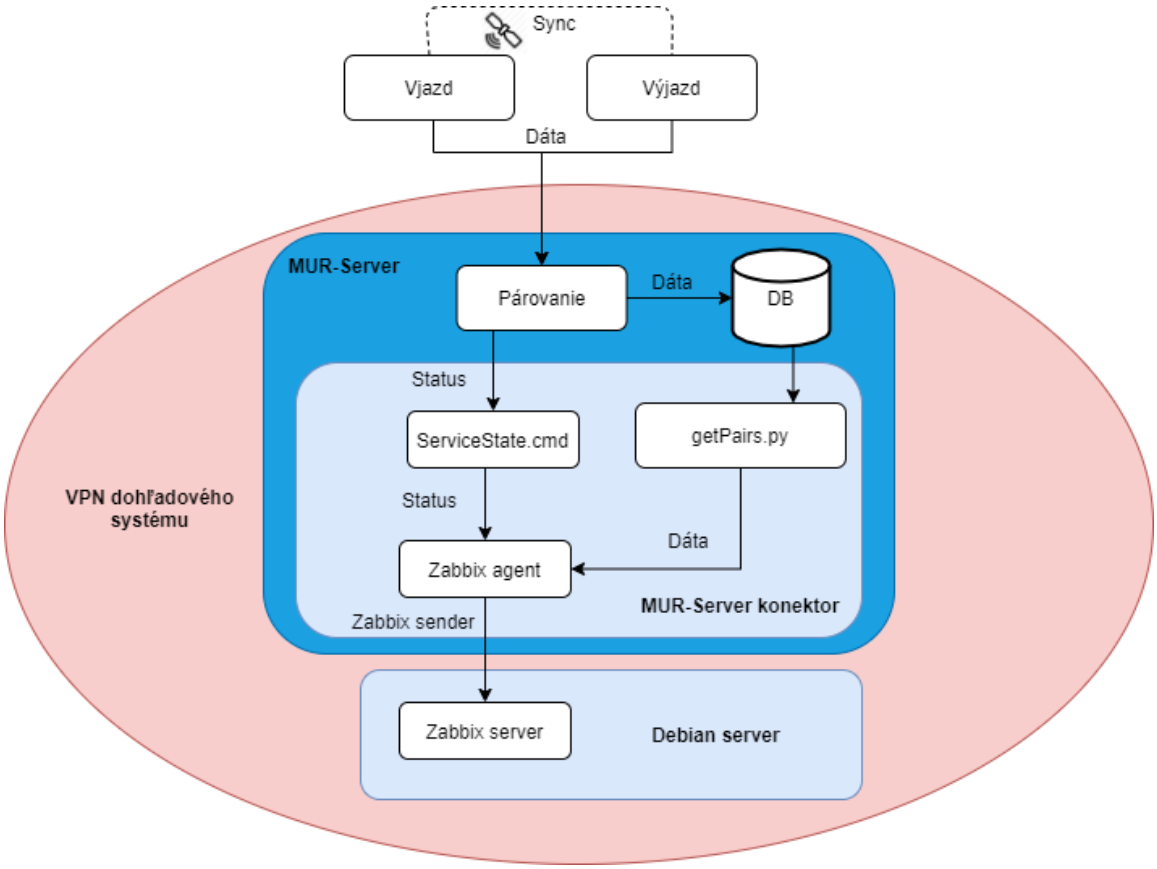
Obr. 5.7: Detail konektora pre SU-MUR zariadenia. Zdroj: Vlastná tvorba.

MUR-Server

MUR-Server slúži na spracovanie dát z jednotlivých SU-MUR zariadení, ktoré sú na tento server pripojené. Monitorovanie úsekovej rýchlosti pozostáva zo zachytenia vozidla na začiatku a konci monitorovaného úseku pomocou SU-MUR zariadení popísaných vyššie. Následne dáta vytvorené týmito jednotkami sú odosielané na spracovanie do serverovej časti. Úlohou serveru je vytvoriť správny pár vjazdových a výjazdových jednotiek a tento záznam uložiť do databázy. V prípade neúspešného spárovania je záznam odstránený.

Úlohou dohľadového systému je v tomto prípade monitorovanie správnej činnosti MUR-Serveru a jednotiek pripojených k nemu. V prípade väčšieho počtu nespárovaných záznamov je vyššia pravdepodobnosť nefunkčnosti jedného zo zariadení SU-MUR. Táto nefunkčnosť môže byť spôsobená buď nepriaznivým počasím, ktoré sa odzrkadlí na nekvalitných videových záberoch, vypadnutím jednotky GPS slúžiacej na synchronizovanie času medzi vjazdovými a výjazdovými jednotkami alebo celkovou nefunkčnosťou SU-MUR zariadenia.

Konektor bol implementovaný pomocou skriptu getPairs.py, ktorý sa využitím knižnice ODBC pripojí na lokálnu *Microsoft databázu SQL* (MSSQL) databázu a vyčíta z nej informácie o pároch a nespárovaných záznamoch. K monitorovaniu párovacích a ďalších nevyhnutných služieb serveru slúži skript ServiceState.cmd (viď príloha B) použitý tiež v konektore pre SU-MUR, ktorý v prípade zlyhania služby okamžite upozorní dispečera. Diagram konektora pre MUR-Server je zobrazený na obrázku číslo 5.8.

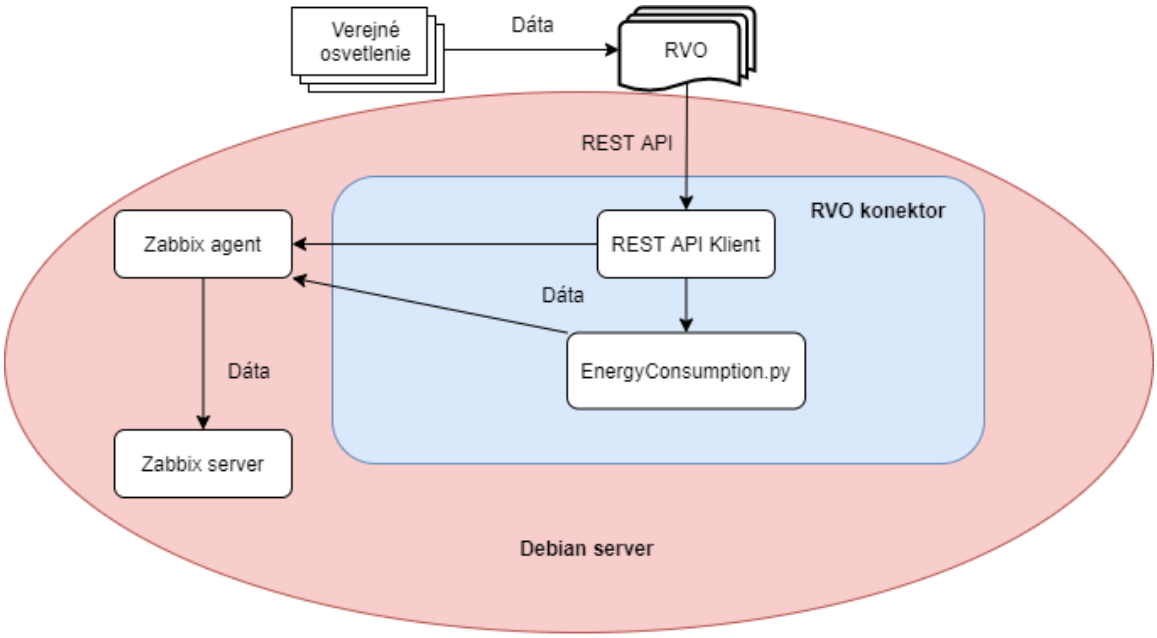


Obr. 5.8: Detail konektora pre MUR-Server zariadenia. Zdroj: Vlastná tvorba.

5.6.3 RVO - Riadenie verejného osvetlenia

Ďalším konektorom, potrebným pre pripojenie k dohľadovému systému je konektor pre zariadenia typu RVO. RVO slúži na riadenie verejného osvetlenia, čiže je k nemu pripojených niekoľko verejných svetiel, ktoré sú ovládané z tohoto zariadenia. Taktiež RVO zbiera dostupné dáta a stavové informácie z jednotlivých svetiel, ktoré si ukladá do lokálnej databázy. Na RVO zariadení beží jednoduchý webový server a prístup k dátam je definovaný a dostupný pomocou REST API.

Hlavnou časťou konektora pre RVO zariadenia je REST API klient implementovaný v jazyku Python. Vstupnými parametrami sú identifikátory požadovaných dát. Výstupom sú spracované dáta z jednotiek RVO. Klient umožňuje spustenie v testovacom režime, ktorý vypíše na štandardný výstup všetky údaje z komunikácie s RVO zariadením. Tento klient je v pravidelných intervaloch volaný Zabbix agentom, ktorý následne dáta predá Zabbix serveru. Ďalšou požiadavkou na monitorovanie bolo sledovanie spotreby energie jednotlivých vetiev verejného osvetlenia ako aj celého systému. K tomuto účelu bol vytvorený skript EnergyConsumption.py, ktorý využitím klienta REST API získa potrebné informácie pre výpočet spotreby za dané obdobie a odošle ich cez Zabbix agenta do Zabbix serveru. Diagram konektoru a toku dát je zobrazený na obrázku číslo 5.9.



Obr. 5.9: Detail konektora pre RVO zariadenia. Zdroj: Vlastná tvorba.

5.6.4 EIP - Elektronický informačný panel

Posledný konektor pre pripojenie k dohľadovému systému sa týka zariadení typu EIP. Tento typ zariadení odosiela v pravidelných intervaloch dáta obsahujúce stav zariadenia a dáta nazbierané z rôznych senzorov. Zariadenia komunikujú na princípe sieťovej architektúry klient-server, kde je využitý TCP protokol.

Z vyššie uvedenej špecifikácii vyplýva, že konektor bude implementovaný formou serveru pre EIP zariadenia takzvaný EIP server. Tento server bude slúžiť na príjem a spracováva-

nie dát prichádzajúcich z EIP zariadení a odosielať ich na zobrazenie do Zabbix serveru pomocou Zabbix sendera (utilita príkazového riadku slúžiaca na odosielanie dát do Zabbix serveru).

Server bol implementovaný formou Python skriptov, ktorých diagram je znázornený na obrázku číslo 5.10. Architektúra výsledného riešenie pozostáva s nasledujúcich tried (kompletná implementácia je v prílohe A):

- Main - Hlavná časť EIP serveru. Je v nej vytvorený logger, ktorý má na starosti zálohu všetkých prijatých, spracovaných a odoslaných dát, ktoré prebehnú cez server. Logger je implementovaný formou kruhového bufferu, kde je možné nastaviť počet súborov, do ktorých sa budú ukladať dáta ako aj veľkosť týchto súborov. Ďalšou úlohou je vytvorenie TCP serveru pomocou triedy TCPServer, ktorej je formou parametru predaný maximálny počet pripojených zariadení.

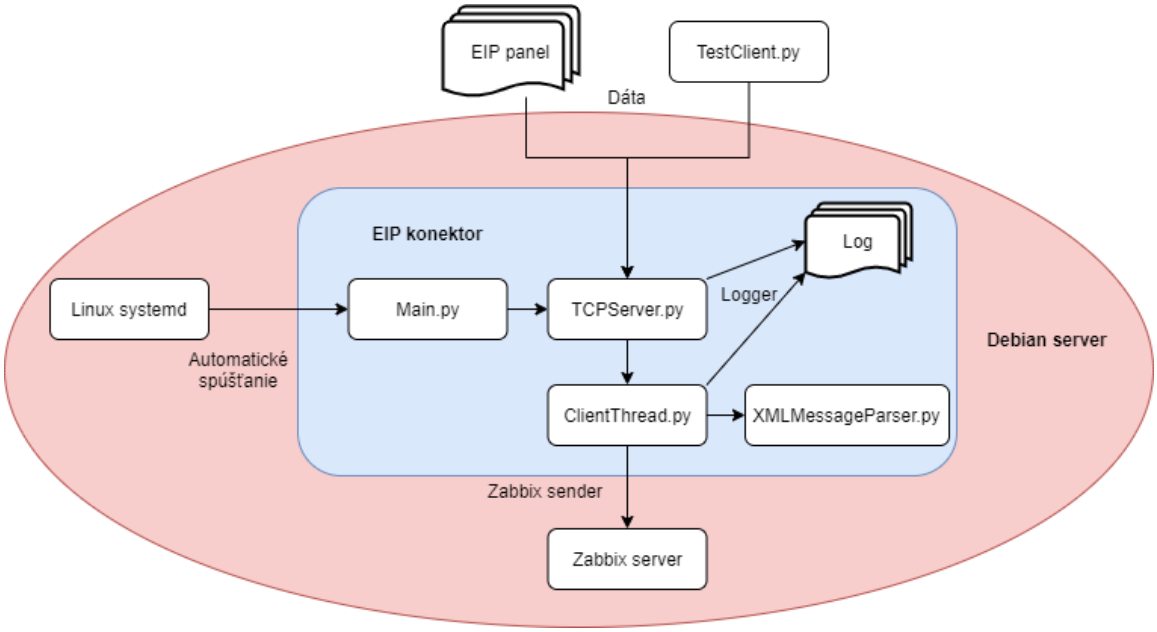
- TCPServer - Hlavná trieda TCP serveru, ktorá čaká na klientov (EIP zariadenia) a inicializuje ich spojenie alebo odpojenie. Využíva triedu ClientThread.

- ClientThread - Pre každého klienta, ktorý sa pripojí je vo vlastnom vlákne spustený kód triedy ClientThread, ktorý sa stará o príjem dát ich spracovanie, odoslanie odpovedi klientovi a odoslanie spracovaných dát do Zabbix serveru pomocou Zabbix sendera. Na spracovanie dát je využívaná trieda XMLMessageParser.

- XMLMessageParser - Trieda, ktorá slúži na parsovanie prijatej správy vo formáte XML. Táto trieda obsahuje metódy na získanie požadovanej informácie zo správy.

- TestClient - Trieda simulujúca EIP zariadenie. Trieda je využívaná pre testovanie bez potreby napojenia na reálne zariadenia v prevádzke. Je implementovaná formou TCP klienta, ktorý odosiela všetky možné stavové dáta a dáta zo senzorov na server.

Po implementácií a otestovaní serveru na príjem a spracovanie dát bol nasadený na Zabbix server. Nasadenie pozostávalo s niekoľkých krokov a to presun zdrojových súborov na server, inštalácia potrebných Python knižníc a nastavenie automatického spúšťania serveru po zapnutí Zabbix serveru.



Obr. 5.10: Detail konektora pre EIP panely. Zdroj: Vlastná tvorba.

5.7 Webové rozhranie dohľadového systému

Zozbierané dáta zo sledovaných zariadení nemajú samé o sebe zmysel, pokiaľ nie sú vhodne interpretované používateľom, v našom prípade dispečerom. Spracovanie dát systémom Zabbix prebieha v dvoch fázach. V prvej fáze sa definujú komunikačné body, cez ktoré získa Zabbix prístup k požadovaným dátam. V druhej fáze sú dáta interpretované konečnému používateľovi formou rôznych textových informácií a grafov. Jednotlivé fázy sa uskutočňujú pomocou webového rozhrania systému Zabbix popísanému v nasledujúcej časti práce.

5.7.1 Nastavenie komunikačných bodov systému Zabbix

K zobrazeniu dát vo webovom rozhraní je najprv potrebné definovať jeho jednotlivé časti, ktoré budú komunikovať s externými preprocesormi a ostatnými servermi. Toto nastavenie prebieha pomocou vyššie spomínaného webového rozhrania systému Zabbix.

Prvým krokom je vytvorenie skupín (Groups) podľa typov monitorovaných zariadení (SU-MUR, MUR-Server, RVO, EIP). Následne je pre každý typ zariadenia vytvorená šablóna obsahujúca všetky definované dáta (z implementovaných konektorov), ktoré daný typ zariadenia podporuje. Okrem dostupných dát je možné v šablóne definovať aj grafy vytvorené z týchto dát a triggre, ktoré informujú o ich aktuálnom stave. Pre každé monitorované

zariadenie je následne v systéme Zabbix vytvorený HOST, kde je definovaný názov, skupina, šablóny, IP adresa Zabbix Agenta a popis zariadenia.

V tejto chvíli sú komunikačné body zariadení v Zabbixe nachystané na ďalšie spracovanie. V nasledujúcej časti je popísané vytvorenie upozornení na vzniknuté udalosti 5.7.2 a vizualizácia nazbieraných dát 5.7.3.

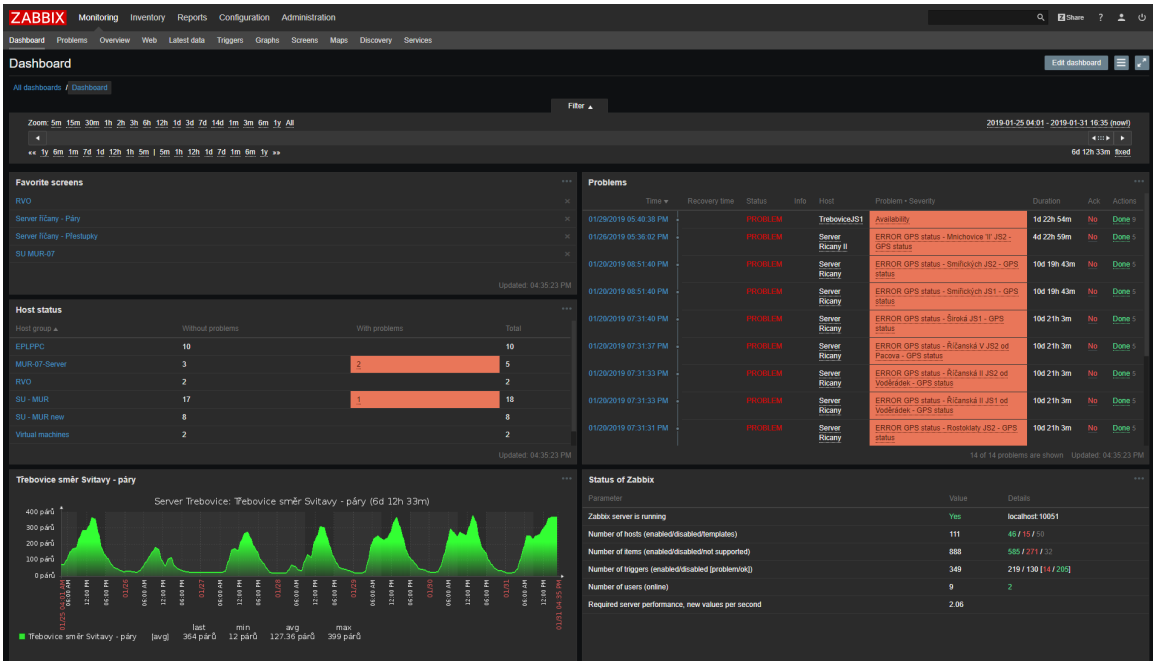
5.7.2 Nastavenie upozornení na vzniknuté udalosti

Ak nastane nejaký problém na dohliadaných zariadeniach je potrebné čo najskôr o danej situácií informovať kompetentného používateľa. Na toto informovanie slúži systém notifikácií, ktorý je súčasťou systému Zabbix. Je možné nastaviť odosielanie informatívnych e-mailov alebo SMS správ definovaným používateľom alebo skupinám používateľov.

Pre funkciu notifikácií boli v Zabbixe vytvorené skupiny používateľov (dispečerov), ktorí sú zodpovední za jednotlivé zariadenia alebo skupiny zariadení monitorovaných v dohľadovom systéme. Hneď po vzniknutí neočakávaného stavu je vytvorený automatický e-mail obsahujúci informácie o probléme a zariadení, na ktorom vznikol. V prípade, že porucha nie je v konfigurovateľnom čase vyriešená, je opätovne rozposlaný e-mail o stave zariadenia. Je možnosť nastaviť rozposielanie notifikácií aj v prípade opravenia poruchy, kedy je zariadenie v normálnom stave ale v našom prípade sme toto nastavenie vynechali.

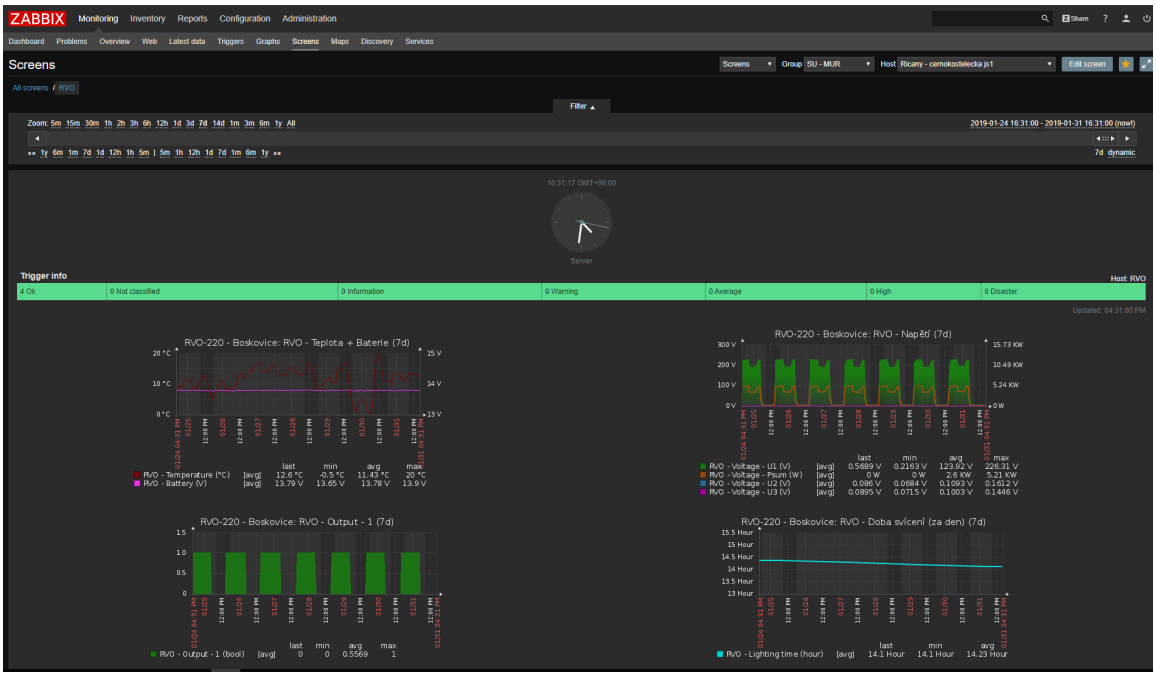
5.7.3 Vizualizácia dát pomocou systému Zabbix

Pri vizualizácií dát sa vychádzalo z predom definovaného rozmiestnenia jednotlivých komponentov rozhrania navrhnutého v sekcii 5.5. Prvým krokom bola konfigurácia dashboard stránky, ktorej výsledok je možné vidieť na obrázku číslo 5.11.

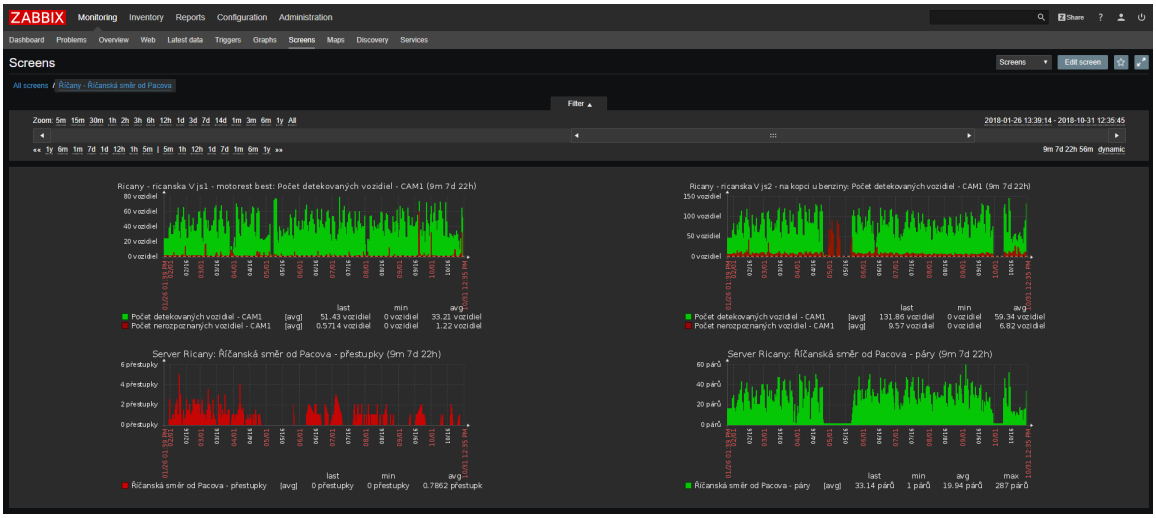


Obr. 5.11: Dashboard monitorovacieho systému Zabbix. Zdroj: Vlastná tvorba.

Ďalšími vytvorenými obrazovkami podľa návrhov sú RVO a MUR zobrazené na obrázkoch 5.12 a 5.13.



Obr. 5.12: Obrazovka monitorovania RVO zariadení. Zdroj: Vlastná tvorba.



Obr. 5.13: Obrazovka monitorovania SU-MUR a MUR-Server zariadení. Zdroj: Vlastná tvorba.

Tieto obrazovky sú pre názornosť doplnené o testovacie dáta. Rozmiestnenie komponentov obrazoviek bolo vytvorené podľa spoločného návrhu avšak webové rozhranie Zabbixu umožňuje jednoducho každú obrazovku editovať a tak prispôsobiť individuálnym potrebám každého jedného dispečera.

5.8 Využité princípy správy kódu a metodika vedenia práce

Pri vytváraní, zmene alebo údržbe rôznych informačných systémov je dôležité uchovávanie vytváraných verzií zdrojových kódov, čo pomáha pri rýchlej obnove nejakej staršej funkčnej verzii alebo umožňuje prácu viacerých členov tímu na rovnakom projekte. Z tohoto dôvodu bol aj pri tvorbe ADS používaný softvér GIT umožňujúci správu verzií zdrojových kódov.

Rovnako dôležitá pri vývoji kvalitného informačného systému je aj metodika jeho vedenia. V rámci implementácie ADS bola využívaná agilná metodika nazývaná SCRUM. Vývoj prebiehal v dvojtýždňových cykloch. Každý druhý deň boli organizované krátke 15 minútové stretnutia tímu, na ktorých sa prezentovala aktuálne vykonávaná práca, plán na ďalšie dni a prípadné prekážky v práci, aby mohli byť čo najskôr odstránené. Na konci každého cyklu sa konalo zhodnotenie odvedenej práci a naplánovanie úloh na nasledujúci cyklus. Na plánovanie práce a evidenciu jednotlivých úloh bol využívaný program REDMINE.

Postupne popri implementácii bolo neoddeliteľnou súčasťou vytváranie dokumentácie potrebnej k správnemu nasadeniu dohľadového systému spolu s pridávaním monitorovaných

zariadení. Distribuovaná je formou zdieľaného súboru dostupného oprávneným používateľom na firemnom úložisku.

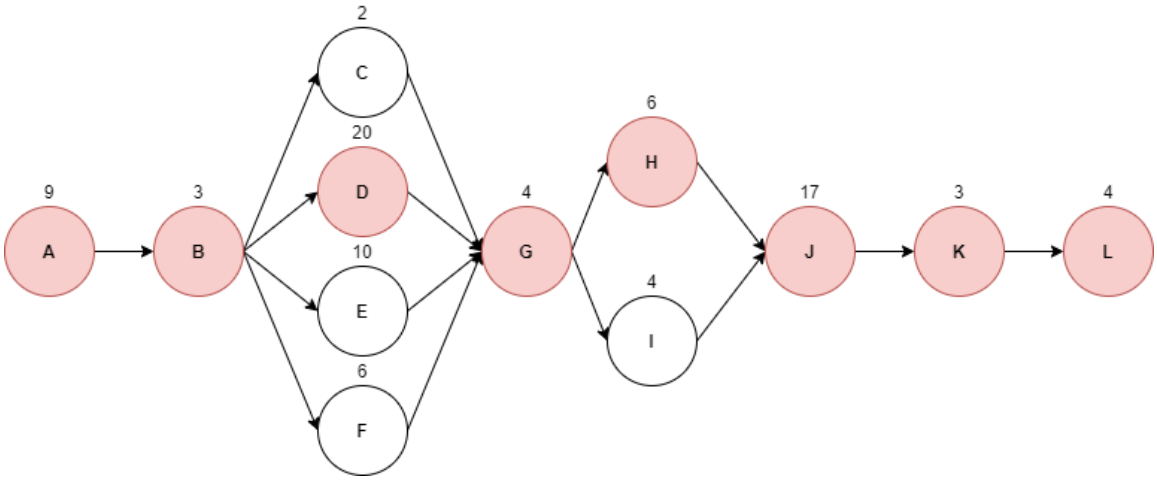
5.9 Časový a obsahový harmonogram nasadenia IS

Nasledujúca časť sa bude venovať časovému a obsahovému harmonogramu nasadenia ADS do prostredia firmy. Nasadeniu IS predchádza niekoľko dôležitých krokov, ktorých doba trvania bola určená kvalifikovaným odhadom. Tieto činnosti sú zobrazené v tabuľke číslo 5.1.

Činnosť	Názov činnosti	Následník	Doba trvania (dni)
A	Analýza a návrh systému		9
B	Schválenie projektu	A	3
C	Príprava infraštruktúry	B	2
D	Implementácia	B	20
E	Testovanie	B	10
F	Tvorba dokumentácie	B	6
G	Úpravy na základe testovania	C, D, E, F	4
H	Finalizácia riešenia	G	6
I	Nasadenie do prostredia firmy	G	4
J	Testovacia prevádzka	H, I	17
K	Schválenie nasadenia	J	3
L	Konečné nasadenie do reálnej prevádzky	K	4

Tabuľka 5.1: Tabuľka s činnosťami potrebnými pre nasadenie ADS. Zdroj: Vlastná tvorba.

Na základe identifikovaných činností potrebných k nasadeniu IS z predchádzajúcej tabuľky bol vytvorený zjednodušený sieťový graf metódy CPM zobrazený na obrázku číslo 5.14. Uzly grafu reprezentujú jednotlivé činnosti. Nad každou činnosťou je zobrazená odhadovaná doba trvania a červenou farbou je vyznačená kritická cesta, ktorá pozostáva z nasledujúcich činností: A, B, D, G, H, J, K a L, u ktorých je časová rezerva rovná nule. Z kritickej cesty vyplýva, že najkratší možný termín ukončenia projektu je 66 dní.



Obr. 5.14: Sieťový graf zobrazujúci činnosti spolu s ich dobou trvania. Kritická cesta je zobrazená červenou farbou. Zdroj: Vlastná tvorba.

5.10 Vyhodnotenie prevádzky

Nasledujúca sekcia je zameraná na zhodnotenie testovania a prevádzky dohľadového systému, jeho náklady a prínos.

5.10.1 Testovanie

Prvá fáza testovania systému prebiehala asi 3 mesiace (do vytvorenia tejto práce, čiže momentálne sme na konci činnosti J definovanej v harmonograme sekcie 5.9) a to súbežne s predchádzajúcim spôsobom monitoringu. Dispečeri hodnotili pozitívne možnosť monitoringu všetkých zariadení na jednom mieste. Priemerný čas na manuálne skontrolovanie všetkých zariadení, ktoré má na starosti 1 dispečer trvá asi pol hodinu. Táto kontrola spočíva všeobecne z nasledujúcich krokov:

- Pripojenie sa na zariadenie pomocou vzdialenej plochy alebo príkazového riadku.
- Skontrolovanie správnosti údajov buď v databázy, súborovom systéme alebo inom programe bežiacom na zariadení.
- Odhlásenie zo zariadenia .
- Vytvorenie záznamu o vykonanej kontrole.

V prípade implementovaného ADS dispečer skontroluje iba stav všetkých zariadení v dashboarde, prípadne v detaily zariadenia. Pri nastaní problému mu príde e-mail s konkrétnymi údajmi o zariadení a probléme. Systém Zabbix pravidelne ukladá požadované dáta, čím sú nahradené záznamy vykonávané dispečermi.

Testovaním bol odhalený problém zahltenia e-mailových schránok s falošnými problémami, ktoré vznikali pre nevhodne zvolené medze v triggroch a tým pádom systém nebol schopný odhaliť výkyvy v hodnotách získaných zariadeniami, ktoré sú bežné. Po prenastavení medzných hodnôt jednotlivých triggrov sa systém správal podľa očakávania.

5.10.2 Náklady

Súčasne beží systém Zabbix na servery s OS Linux (Debian - 4 jadrá CPU a 4GB RAM), ktorý na aktuálny počet (testovací) zariadení postačuje. Nasledujúca časť venujúca sa nákladom počíta aj s reálnymi potrebami väčšieho počtu zariadení, preto je rozdelená na základnú a optimálnu konfiguráciu dohľadového systému.

Systém Zabbix patrí medzi slobodný softvér a cena jeho používania je nulová. Avšak do nákladov je potrebné zahrnúť programátora, ktorý ho nasadí a implementuje konektory pre jednotlivé zariadenia. Ďalším dôležitý prvkom je dohľadová miestnosť, ktorá bude vybavená dostatočne veľkými monitormi na zobrazenie všetkých potrebných dát. Keďže dispečeri nemajú žiadne skúsenosti s používaním ADS a programovania, bude potrebné ich zaškoliť, aby vedeli rýchlo reagovať na vzniknuté situácie a prípadne rozširovať konektory na ďalšie nové zariadenia pridané do dohľadového systému. Pre beh systému Zabbix možno použiť hardvér vlastnený firmou avšak rozumnejšie do budúcnosti bude zaobstarať výkonnejší stroj, keďže sa očakáva pribúdanie nových zariadení. Náklady sú spočítané v nasledujúcej tabuľke 5.2.

Vstupné náklady				
Hardvér				
	konfigurácia			
	základná		**optimálna**	
	typ	**cena**	**typ**	**cena**
Server	Aktuálne využívaný server	0 €	Server DELL PowerEdge R440	1 772,35 €
Monitor	DELL S3220DGF	348,02 €	4x DELL S3220DGF	1 392,08 €
Drobný majetok (káble, adaptéry, disk na zálohu atď.)	-	200 €	-	200 €
Softvér				
Systém	Zabbix	0 €	Zabbix	0 €
Ostané				
Programátor (40 človekodní)	Implemetácia	3 125 €	Implementácia	3 125 €
Školenia	Školenie dispečerov	700 €	Školenie dispečerov	700 €
Spolu	-	4 373,02 €	-	7 189,43 €
Mesačné náklady				
Softvér				
Operačný systém	Linux Debian	0 €	RedHat Enterprise Linux	315,38 €

Tabuľka 5.2: Tabuľka nákladov spojených s nasadením a prevádzkou dohľadového systému Zabbix. Zdroj: Vlastná tvorba.

Pri počítaní nákladov programátora sa uvažovalo o 40 dňoch práce za priemerný plat v obore. Školenie by prebehlo v rámci dvoch dní, kedy programátor nasadzujúci systém preškolí cca 5 dispečerov na používanie Zabbix dohľadového systému a základy skriptov v jazyku Python. Nákup Serveru a plateného OS Linux RedHat Enterprise umožní využiť Zabbix systém aj pri jeho maximálnom možnom zaťažení s vyše 10 000 monitorovanými zariadeniami, čo bude pre účely dohľadu servisného oddelenia bohato stačiť aj do budúcnosti. Ceny serveru, monitorov a operačného systému boli získané z oficiálnych stránok

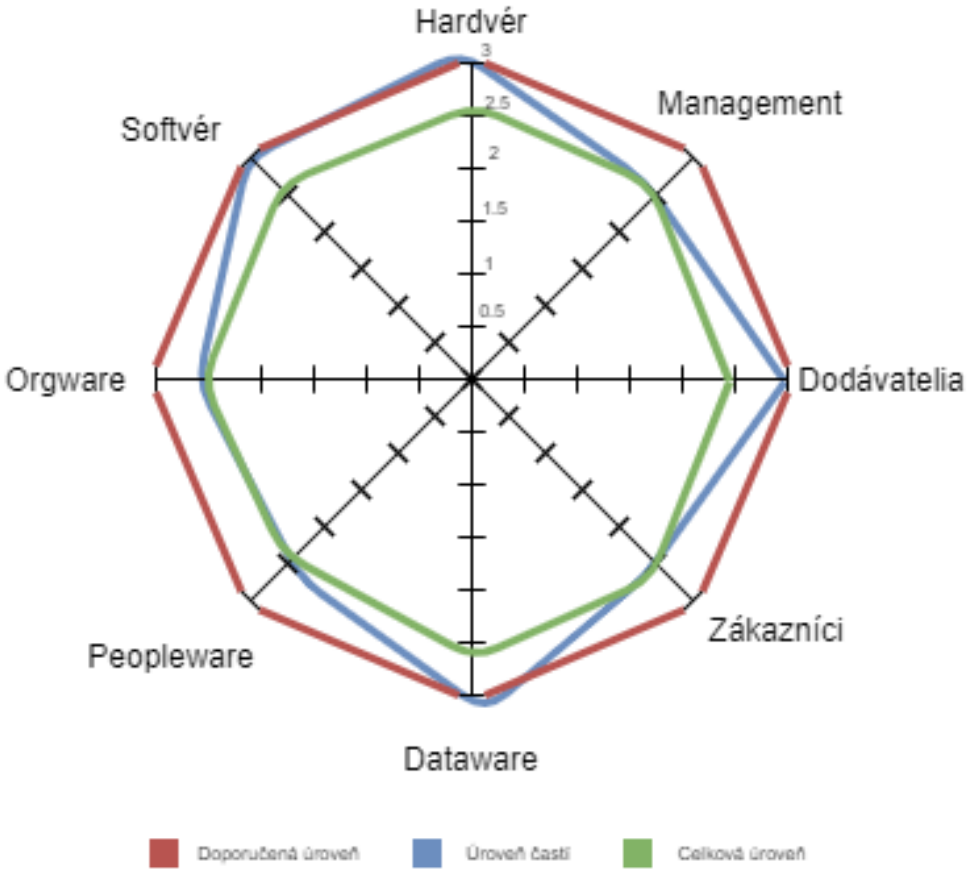
spoločností Dell a Red Hat. Všetky ceny (aj kurzy Európskej centrálnej banky, s ktorými bolo počítané) sú aktuálne k dátumu 3.2.2020.

Po spočítaní všetkých vstupných nákladov sa výsledná suma pohybuje na úrovni 4 373€ pre základnú a 7 189€ pre optimálnu konfiguráciu systému. S optimálnou konfiguráciou sú spojené navyše mesačné náklady za operačný systém vo výške 315€. S nasadením ADS sú spojené očakávania na zníženie celkových nákladov na servis popísaných vyššie. Táto úspora by mala zaručiť návratnosť vstupných aj mesačných nákladov systému.

5.10.3 Efektívnosť servisných procesov podľa HOS po nasadení dohľadového systému

Vyplývajúc z analýzy efektívnosti v sekcii 4.6 sú v systéme isté slabšie miesta, ktoré spôsobujú neefektivitu celého systému. Po nasadení ADS očakávame zlepšenie poskytovaných služieb zákazníkom a efektívnejšie využitie ľudských zdrojov, konkrétne dispečerov a aktualizovanie noriem týkajúcich sa servisu.

Ako bolo spomenuté v predchádzajúcej časti 5.10.1, tak ADS je nasadený zhruba 3 mesiace. Za toto krátke obdobie bola zaznamenaná rýchlejšia reakcia dispečerov na vzniknuté problémy, avšak stále to nedosiahlo požadovanú úroveň viď graf číslo 5.15. Toto je ale pravdepodobne spôsobené tým, že dispečeri počas testovacej prevádzky musia prípadné udalosti verifikovať starým spôsobom až pokým nebude ADS používaný ako primárny, čo súvisí aj s rovnakou efektivitou pri využívaní zamestnancov. Čo sa týka nových noriem, tak ich vytvorenie prebehne až po ukončení testovacej prevádzky. Či budú vyslovené predpoklady naplnené sa ukáže až po dlhšom období používania systému. Aktuálny stav efektívnosti divízie DAST je zobrazený na nasledujúcom grafe číslo 5.15.



Obr. 5.15: Graf posúdenia efektívnosti divízie DAST podľa metódy HOS po nasadení ADS. Zdroj: Vlastná tvorba.

Kapitola 6

Záver

Hlavným cieľom práce bolo navrhnúť a implementovať automatizovaný dohľad nad požadovanými zariadeniami firmy AŽD Praha s.r.o. V teoretickej časti sú popísané základné princípy a metodiky potrebné k pochopeniu danej problematiky. Nasleduje bližšie predstavenie zamerania zvolenej firmy a jej organizačnej štruktúry.

Analýzou súčasného prístupu firmy k monitorovaniu zariadení a s pomocou analýzy HOS sa potvrdila nutnosť zavedenia automatizovaného dohľadového systému k zlepšeniu efektívnosti servisu firmy. V ďalšej časti analýzy boli predstavené tri systémy slúžiace k monitorovaniu zariadení. Komparatívna analýza umožnila zvoliť ako najlepšie riešenie systém Zabbix. Medzi hlavné dôvody, ktoré prispeli k použitiu tohto systému, patrili cena a predchádzajúca skúsenosť s týmto systémom.

Následne prebehol návrh celého systému, z ktorého vznikla potreba vytvorenia konektorov (preprocesorov) pre jednotlivé typy monitorovaných zariadení. Tieto konektory slúžia na prvotné spracovanie dát zo zariadení, ktoré sú následne odoslané do systému Zabbix. Pre dosiahnutie čo najvyššej bezpečnosti bol celý navrhovaný systém umiestnený do virtuálnej privátnej siete. Nasledoval výber technológií potrebných na implementáciu konektorov a integráciu systému Zabbix. Spolupráca s dohľadovými pracovníkmi (dispečermi) umožnila vytvorenie základného návrhu vzhľadu webového rozhrania systému Zabbix, ktorý pozostáva z hlavnej stránky (dashboard) a stránok detailov jednotlivých typov monitorovaných zariadení. Sieťovou analýzou bola odhadnutá doba realizácie celého projektu nasadenia automatizovaného dohľadového systému na 66 dní.

Posledná časť práce sa venovala implementácii navrhnutého riešenia. Prvý krok predstavoval prípravu operačného systému Linux (distribúcia Debian) pre inštaláciu a konfiguráciu

Zabbix serveru. Nasledovala implementácia samotných konektorov pre jednotlivé typy zariadení. Po implementácií vznikla nutnosť sprístupnenia komunikačných bodov v systéme Zabbix, ktoré využívajú konektory na prenos dát zo zariadení do dohľadového systému. Ako posledný krok prebehlo vytvorenie jednotlivých stránok podľa pripraveného návrhu prostredníctvom webového rozhrania systému.

Testovacia prevádzka trvala asi tri mesiace, počas ktorých fungoval pôvodný dohľad doplnený o automatizovaný dohľad využívajúci systém Zabbix. Táto prevádzka odhalila drobné chyby, ktoré sa podarilo odstrániť. Nasledovalo zhrnutie ako časových tak aj finančných nákladov spojených s implementáciou a nasadením systému Zabbix a potrebným preškolením používateľov.

Ak vezmeme do úvahy testovaciu prevádzku, môžeme povedať, že prišlo k výraznému uľahčeniu práce dispečerov a zvýšeniu schopnosti rýchlej reakcie na vzniknuté problémy. Do budúcnosti prinesie používanie systému efektívnejšie využívanie zamestnancov, rýchlejšie reakcie na nepredvídateľné udalosti, čo prinesie zefektívnenie servisu a nákladov s ním spojených. Overenie pravdivosti týchto predpokladov umožní až dlhšia doba používania systému, avšak už teraz môžem konštatovať, že ciele práce špecifikované v zadaní boli splnené.

Vytvorenie tejto práce znamenalo prospech aj pre mňa osobne. Prinieslo mi to veľké množstvo nových skúseností, či už z tvorby rôznych firemných analýz, z identifikácií rizík a ich hodnotení alebo z nasadenia automatizovaného dohľadového systému. Dohľadové systémy sú v dnešnej dobe nevyhnutnou súčasťou servisnej agendy väčšiny firiem. Navrhnutím a implementáciou systému sa preukázalo, že je možná jeho integrácia aj do zabehnutej firmy, ktorej môže priniesť či už zníženie nákladov alebo zjednodušenie monitoringu. Do budúcnosti je možné rozšíriť tento systém aj pre dohľad ďalších typov zariadení, ktorými sa táto práca nezaoberala.

V dnešnej dobe je otázka efektívnosti aktuálna napriek širokým spektrom oborov. Nasadenie systému v rámci spoločnosti AŽD Praha s.r.o. bolo hodnotené kompetentnými osobami ako úspešné a preto môže byť inšpiráciou aj pre ostatných, ktorí potrebujú zefektívniť či už dohľad nad rôznymi zariadeniami alebo ľubovoľné firemné procesy (napríklad procesy distribúcie, vývoja, testovania a podobne).

Literatúra

[1] 50Minutes. *McKinsey 7S Framework: Boost business performance, prepare for change and implement effective strategies (Management & Marketing)*. 1. vyd. 50Minutes.com, 2015. ISBN 978-2806269959.

[2] Advisor, O. *Why We're Moving to Agile Software Development* [online]. Orion Advisor Technology, 2019 [cit. 2020-01-01]. Dostupné z: https://orionadvisortech.com/blog/why-were-moving-to-agile-software-development/.

[3] BOEHM, B. W. *A spiral model of software development and enhancement*. 1. vyd. Computer, 1988. ISBN 1558-0814.

[4] BOEHM, B. W. a PAPACCIO, P. N. *Understanding and Controlling Software Costs*. 1. vyd. IEEE Transactions on Software Engineering, 1988. ISBN 1939-3520.

[5] BRUCKNER, T., VOŘÍŠEK, J., BUCHALCEVOVÁ, A. et al. *Tvorba informačních systémů : Principy, metodiky, architektury*. 1. vyd. Praha: Grada Publishing, 2012. ISBN 978-80-247-4153-6.

[6] CARDOSO, J., LOPES, R. a POELS, G. *Service Systems: Concepts, Modeling, and Programming*. 1. vyd. Springer International Publishing, 2014. ISBN 978-3-319-10813-1.

[7] CHACON, S. a STRAUB, B. *Pro Git* [online]. 2. vyd. Apress, 2014 [cit. 2020-01-01]. Dostupné z: https://git-scm.com/book/en/v2.

[8] COLE, J. *What is a VPN?* [online]. vpnpro.com, júl 2018 [cit. 2020-01-01]. Dostupné z: https://vpnpro.com/vpn-basics/what-is-a-vpn.

[9] Datadog. *Datadog* [online]. 2019 [cit. 2020-01-01]. Dostupné z: https://www.datadoghq.com/.

[10] DELANEY, K., BEAUCHEMIN, B., CUNNINGHAM, C., KEHAYIAS, J., RANDAL, P. S. et al. *Microsoft SQL Server 2012 Internals (Developer Reference)*. 1. vyd. Sebastopol: Microsoft Press, 2013. ISBN 978-0-7356-5856-1.

[11] DOWNEY, A. B. *Think Python*. 1. vyd. O'Reilly Media, Inc., 2012. ISBN 9781449332020.

[12] DRAKE, F. L. *Python documentation* [online]. Python Software Foundation, 2019 [cit. 2020-01-01]. Dostupné z: https://docs.python.org.

[13] ČERMÁK, M. *ITIL tajemství zbavený* [online]. December 2009 [cit. 2020-01-01]. Dostupné z: https://www.cleverandsmart.cz/itil-tajemstvi-zbaveny/.

[14] GRASSEOVÁ, M., DUBEC, R. a ŘEHÁK, D. *33 nejpoužívanějších metod strategického řízení*. 1. vyd. Brno: Computer Press, 2010. ISBN 987-80-251-2621-9.

[15] KASHYAP, V. *12 Best Business Management Software You Should be Using Today* [online]. proofhub.com, august 2018 [cit. 2020-01-01]. Dostupné z: https://www.proofhub.com/articles/best-business-management-software.

[16] KLEPPMANN, M. *Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems*. 1. vyd. O'Reilly Media, Inc., 2017. ISBN 9781491903117.

[17] KOČÍ, R. a KŘENA, B. *Úvod do softwarového inženýrství. Studijní opora*. Vysoké učení technické v Brně, 2010.

[18] KOCH, M. *Posouzení efektivnosti informačního systému metodou HOS* [online]. Brno University of Technology, Faculty of Business and Management, Kolejní 2906/4, 612 00 Brno, Czech Republic, júl 2013 [cit. 2020-01-01]. Dostupné z: https://trends.fbm.vutbr.cz/index.php/trends/article/download/211/207.

[19] KOCH, R. *The 80/20 Manager*. 1. vyd. Eastone Books, 2011. ISBN 9788081091773.

[20] LLC, Z. *Zabbix* [online]. Zabbix LLC, 2020 [cit. 2020-01-01]. Dostupné z: https://www.zabbix.com/.

[21] MOLNÁR, Z. *Efektivnost informačních systémů*. 1. vyd. Praha: Grada Publishing, 2000. ISBN 80-7169-410-X.

[22] Movisio. *AŽD Praha s.r.o.* [online]. azd.cz, 2020 [cit. 2020-01-01]. Dostupné z: https://www.azd.cz/cs.

[23] Nagios Enterprises, L. *Nagios* [online]. 2019 [cit. 2020-01-01]. Dostupné z: https://www.nagios.org/.

[24] OLUPS, R. *Zabbix Network Monitoring.* 2. vyd. Packt Publishing, 2016. ISBN 978-1782161288.

[25] RAIS, K. a DOSKOČIL, R. *Risk management : studijní text pro kombinovanou formu studia.* 1. vyd. Brno : Akademické nakladatelství CERM, 2007. ISBN 9788021435100.

[26] RAIS, K. a DOSKOČIL, R. *Operační a systémová analýza I: studijní text pro prezenční a kombinovanou formu studia.* 1. vyd. CERM, 2011. ISBN 978-80-214-4364-8.

[27] RICHARDSON, L., AMUNDSEN, M. a RUBY, S. *Restful Web Apis.* 1. vyd. O'Reilly Media, 2013. ISBN 978-1449358068.

[28] ROYCE, W. *MANAGING THE DEVELOPMENT OF LARGE SOFTWARE SYSTEMS* [online]. Proceedings of IEEE WESCON, 1970 [cit. 2020-01-01]. Dostupné z: http://www-scf.usc.edu/~csci201/lectures/Lecture11/royce1970.pdf.

[29] SCHWALBE, K. *Řízení projektů v IT : Kompletní průvodce.* 1. vyd. Praha: Computer Press, 2011. ISBN 978-80-251-2882-4.

[30] SULLIVAN, D. *NoSQL for Mere Mortals.* 1. vyd. Addison-Wesley Professional, 2015. ISBN 978-0134023212.

[31] Yang, G. *Life Cycle Reliability Engineering.* 1. vyd. Wiley, 2007. ISBN 0471715298.

[32] Öztürk, M. *Nagios* [online]. medium.com, 2019 [cit. 2020-01-01]. Dostupné z: https://medium.com/software-and-technology/nagios-101-437e32d121d8.

Príloha A

Príklad implementácie konektora EIP

Main serveru pre EIP:

```
1  #!/usr/bin/env python
2  # -*- coding: utf-8 -*-
3
4  # subor:    eip_ppc_server.py
5  # autor:    Marek Celka
6  # email:    Celka.Marek@azd.cz
7  #           marekcelka2@gmail.com
8  # popis:    TCP server, ktory obsluhuje poziadavky z eip ppc panelu a zabbixu
9  #           (flask framework)
10 # pouzitie: python eip_ppc_server.py
11 #           (alebo sluzba: service server-eip-ppc start)
12
13 import os
14 from os.path import basename
15 import sys
16 import time
17 import datetime
18 import logging
19 from logging.handlers import RotatingFileHandler
20 import traceback
21 import zipfile
22 import subprocess
23 import re
```

```
24 import csv
25 import collections
26
27 from TCP_Server import TCP_Server
28
29 if __name__ == '__main__':
30     NUMBER_OF_DEVICES = 100
31     TCP_IP = 'IP'
32     TCP_PORT = 'PORT'
33     BUFFER_SIZE = 1024
34     # standartny logger (loguje prichozi data)
35     logHandler_out_data = RotatingFileHandler('PATH/app_info_data.log',
36                                               maxBytes=1024 * 1024 * 10, backupCount=7)
37     # standartny logger (loguje chyby)
38     logHandler_out = RotatingFileHandler('PATH/app_info.log',
39                                             maxBytes=1024 * 1024 * 10, backupCount=7)
40
41     logger_data = logging.getLogger("data_logger")
42     logger_data.setLevel(logging.DEBUG)
43     logger_data.addHandler(logHandler_out_data)
44     ppc_eip_server = logging.getLogger("logger")
45     ppc_eip_server.setLevel(logging.DEBUG)
46     ppc_eip_server.addHandler(logHandler_out)
47
48     tcp_server = TCP_Server(TCP_IP, TCP_PORT, BUFFER_SIZE, logger_data)
49     tcp_server.listening(NUMBER_OF_DEVICES)
50     tcp_server.close_conn()
```

Trieda TCPServer:

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

import socket
import datetime
from XML_Message import XML_Message
from Client_Thread import Client_Thread

class TCP_Server:

    def __init__(self, ip, port, buffer_size, log_file):
        self.log_file = log_file
        self.buffer_size = buffer_size
        self.ip = ip
        self.port = ports
        self.create_socket(ip, port, buffer_size)

    def create_socket(self, ip, port, buffer_size):
        self.s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        self.s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        self.s.bind((ip, port))
        self.threads = []

    def listening(self, x):
        while 1:
            self.s.listen(x)
            (self.conn, (self.ip, self.port)) = self.s.accept()
            newthread = Client_Thread(self.conn, self.ip, self.port, self.log_file)
            newthread.start()
            self.threads.append(newthread)
            self.log_file.info('Connection address: ' + str(self.ip)
                               + " \nTime: " + str(datetime.datetime.now()))

        for t in self.threads:
            t.join()

    def close_conn(self):
        self.conn.close()
```

Trieda ClientThread:

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

import sys
import traceback
import socket
from subprocess import call
from threading import Thread
from SocketServer import ThreadingMixIn
from XML_Message import XML_Message

class Client_Thread(Thread):

    def __init__(self, conn, ip, port, log_file):

        Thread.__init__(self)
        self.conn = conn
        self.ip = ip
        self.port = port
        self.log_file = log_file

        self.state_dict = {'1':'startApp', '2':'stateLED', '3':'sprintServer',
                           '4':'onlineData', '5':'restartApp', '6':'infoWifi',
                           '7':'setSprinter', '8':'statePower', '9':'irFilter',
                           '10':'shakeSensor'}
    def run(self):

        self.log_file.info("[+] New server socket thread started for " + str(self.ip)
                           + ":" + str(self.port))

        try:
            while True :

                data = self.conn.recv(2048)

                if not data == "":

                    self.log_file.info("Server received data:" + str(data))
```

```
39
40                  call(["zabbix_sender -z IP -s 'TEST_PPL' -k 'stateTrap' -o '"
41                          + str(data) + "'"], shell=True)
42
43                  msg = XML_Message(str(data), self.log_file)
44
45                  idElp = msg.get_id_elp()
46
47                  state = msg.get_state()
48
49                  if not state == None:
50                      self.log_file.info("\nStates: " + str(idElp)
51                                         + " - " + str(state))
52                      for item in state:
53                          if not (str(item['@id']) == '7'):
54                              call(["zabbix_sender -z IP -s 'EPL_PPC_"
55                                   + str(idElp) + "' -k '"
56                                   + self.state_dict[str(item['@id'])]
57                                   + "' -o '" + str(item['text'].encode('utf-8'))
58                                   + "'"], shell=True)
59
60                  response = msg.get_response()
61
62                  self.conn.send(response)
63
64              else:
65                  self.log_file.info("[-] Close thread for ip: "
66                                     + self.ip + ":" + str(self.port)
67                                     + " Traceback: " + str(traceback.print_exc()))
68                  sys.exit(0)
69
70          except:
71              self.log_file.info("[-] Close thread for ip: " + self.ip + ":"
72                                 + str(self.port)
73                                 + " Traceback: " + str(traceback.print_exc()))
74              sys.exit(0)
```

Trieda XMLMessageParser:

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

import xmltodict
import json

class XML_Message:

    def __init__(self, message, log_file):

        self.log_file = log_file
        self.parse_message(message)

    def parse_message(self, message):

        xmldict = xmltodict.parse(message, encoding='utf-8')

        json_data_dump = json.dumps(xmldict, ensure_ascii=False).encode('utf8')
        json_data = json.loads(json_data_dump)

        self.msg_id = json_data['msg']['@id']
        self.msg_time = json_data['msg']['@dt']
        self.msg_id_elp = json_data['msg']['@idElp']
        if 'stav' in json_data['msg']:
            if isinstance(json_data['msg']['stav'], dict):
                self.state = [json_data['msg']['stav']]
            else:
                self.state = json_data['msg']['stav']
        else:
            self.state = None

        self.log_file.info("--" + str(json.dumps(xmldict, indent=4)) + "\n")

    def make_response(self):

        xml_response = "<?xml version=\"1.0\" encoding=\"windows-1250\"?><msg id=\""
                       + str(self.msg_id) + "\"></msg>"

```

```
39          return xml_response
40
41      def get_response(self):
42
43          return self.make_response()
44
45      def get_msg_id(self):
46
47          return self.msg_id
48
49      def get_msg_time(self):
50
51          return self.msg_time
52
53      def get_id_elp(self):
54
55          return self.msg_id_elp
56
57      def get_state(self):
58
59          return self.state
```

Testovací klient:

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

import socket
import time
import sys

num_msg = sys.argv[1]
TCP_IP = 'IP'
TCP_PORT = 'PORT'
BUFFER_SIZE = 1024
MESSAGE_COMMON = "<?xml version='1.0' encoding='windows-1250'?>"
                 +"<msg dt='2018-10-29T09:26:12+02:00' idElp='229' id='111111'>"
                 +"<stav id='0' stat='OK' desc='Alive'></msg>"
MESSAGE_ERROR = "<?xml version='1.0' encoding='windows-1250'?>"
                 +"<msg dt='2018-10-29T09:26:12+02:00' idElp='229' id='22222'>"
                 +"<stav id='1' stat='VAR' desc='Power lost'>"
                 +"<text>Power lost</text>"
                 +"<time>2018-05-06T09:04:04</time></stav></msg>"

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((TCP_IP, TCP_PORT))

for x in xrange(10):

    s.send(test)
    data = s.recv(BUFFER_SIZE)
    print "received data:", data
    time.sleep(5)

s.close()
```

Príloha B

Implementácia skriptu (batch) na zistenie stavu Windows služby

Nasledujúca časť kódu reprezentuje batch skript v príkazovej riadke operačného systému Windows, ktorý slúži na monitorovanie stavu služieb OS Windows. V prípade, že zadaná služba beží v normálnom režime skript vráti číslo 1 inak 0.

```
@echo off
setlocal ENABLEDELAYEDEXPANSION
set params=
for %%x in (%*) do
(
    set params=!params! %%x
)
set servicename=!params:~1!
sc query "%servicename%" | find "RUNNING" > NULL
IF NOT ERRORLEVEL 1
(
    echo 1
)
else
(
    echo 0
)
```