VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

ÚSTAV SOUDNÍHO INŽENÝRSTVÍ
INSTITUTE OF FORENSIC ENGINEERING

RIZIKOVÉ CHOVÁNÍ ETL PROCESŮ V PROSTŘEDÍ DATOVÉHO SKLADU

RISK BEHAVIOUR OF ETL PROCESSES IN A DATA WAREHOUSE

DIPLOMOVÁ PRÁCE
DIPLOMA THESIS

AUTOR PRÁCE
AUTHOR

Bc. KATEŘINA KOŠINOVÁ

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. VLADIMÍR BARTÍK, Ph.D.

BRNO 2015

Vysoké učení technické v Brně, Ústav soudního inženýrství

Akademický rok: 2014/15

ZADÁNÍ DIPLOMOVÉ PRÁCE

student(ka): Bc. Kateřina Košinová

který/která studuje v **magisterském studijním programu**

obor: **Řízení rizik v informačních systémech (3901T047)**

Ředitel ústavu Vám v souladu se zákonem č. 111/1998 Sb., o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma diplomové práce:

Rizikové chování ETL procesů v prostředí datového skladu

v anglickém jazyce:

Risk Behaviour of ETL Processes in a Data Warehouse

Stručná charakteristika problematiky úkolu:

Technická zpráva diplomové práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap, které byly vyřešeny v rámci ročníkového a semestrálního projektu (30 až 40% celkového rozsahu technické zprávy).
Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Cíle diplomové práce:

1. Prostudujte problematiku datových skladů a OLAP technik. Podrobně se zaměřte na ETL procesy.
2. Seznamte se s problematikou analýzy rizik.
3. Implementujte v prostředí MS SQL Serveru ETL proces, v jehož rámci bude prováděna analýza rizik.
4. Na základě provedené analýzy proveďte eliminaci potenciálních rizik.
5. Ověřte funkčnost implementovaného řešení.
6. Zhodnoťte dosažené výsledky a další možné pokračování v tomto projektu.

Při obhajobě semestrální části diplomového projektu je požadováno: body 1 a 2, a bod 6.

Seznam odborné literatury:

Ponniah, P.: Data Warehousing Fundamentals. John Wiley and Sons, 2001.
Laberge, R.: Datové sklady - Agilní metody a business intelligence, Computer Press, Brno, 2012.

Vedoucí diplomové práce: Ing. Vladimír Bartík, Ph.D.

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2014/15.

V Brně, dne 24. 10. 2014

doc. Ing. Aleš Vémola, Ph.D.
ředitel vysokoškolského ústavu

Abstrakt

Tato diplomová práce se zabývá rizikovým chováním ETL procesů v prostředí datového skladu. V první části práce je definována problematika ETL procesů a cíl této práce. Druhá část je zaměřena na teoretická východiska, která jsou potřebná pro vytvoření datového skladu, definici ETL procesů a odhalení možných rizik. Třetí část práce je věnována odhalení potencionální rizik ETL procesů pomocí analýzy a hodnocení rizik. Nedílnou součástí této kapitoly je řízení potenciálních rizik. Čtvrtá část je zaměřena na úpravu ETL procesů tak, aby předcházely potenciálním rizikům. Důležitou součástí této kapitoly je tzv. havarijní plán, který obsahuje důležité postupy, které je nutné dodržet v případě, že nastane riziko. Pátá část této diplomové práce obsahuje shrnutí veškerých poznatků zjištěných v průběhu vývoje a analýzy.

Abstract

This thesis is about hazardous of ETL processes in their data warehouse. In the first part of this thesis I have defined the ETL processes and the aim of this thesis. The second part is about theoretical solutions needed to create a data warehouse, the definition of ETL processes and discovering potential risks. The third part is about discovering potential risks of ETL processes using an analysis and risk assessment. This part also includes a control of the potential risks. The fourth part concentrates on modifying the ETL processes to prevent potential risks. An important part of this chapter is an emergency plan containing necessary processes which must be applied in case of a risk. The fifth part of this thesis is a summary of all knowledge found during the analysis and development.

Klíčová slova

ETL procesy, analýza, potenciální riziko, SQL, datový sklad, databáze

Keywords

ETL processes, analysis, potential risk, SQL, data warehouse, database

Bibliografická citace

KOŠINOVÁ, K. *Rizikové chování ETL procesů v prostředí datového skladu.* Brno: Vysoké učení technické v Brně, Ústav soudního inženýrství, 2015. 83 s. Vedoucí diplomové práce Ing. Vladimír Bartík, Ph.D..

Prohlášení

Prohlašuji, že jsem diplomovou práci zpracovala samostatně a že jsem uvedla všechny použité informační zdroje.

V Brně dne 20. 5. 2015

…….

podpis diplomanta

Poděkování

Na tomto místě bych ráda poděkovala mé rodině a blízkým, kteří mi byli psychickou podporou v průběhu celého studia. V neposlední řadě patří mé „děkuji“ panu Ing. Vladimíru Bartíkovi, Ph.D. za jeho cenné rady ohledně mé diplomové práce.

OBSAH

ÚVOD A CÍL PRÁCE

Tématem mé diplomové práce je rizikové chování ETL procesů v prostředí datového skladu. S daty se setkává každý z nás v každodenním běžném životě. Pomocí ETL procesů dochází k získání určitých dat ze zdroje. Následuje nahrání do databáze, kde jsou data dále upravována do požadované formy. V takovéto formě jsou data zaregistrována do datového skladu, kde jsou uložena v dimenzionálních nebo faktových tabulkách, ze kterých jsou tato data přejímána jako informace. Tyto informace jsou dále předávána businessu, kterému slouží pro další rozhodování.

V průběhu registrace dat pomocí ETL procesů se mohou vyskytnout značná rizika, která mohou ovlivnit úspěšné zanesení do datového skladu.

Cílem této práce je odhalení potenciálních rizik a následný návrh opatření na jejich odstranění, případně minimalizaci.

1 TEORETICKÁ VÝCHODISKA PRÁCE

V následujících podkapitolách této práce jsou definována potřebná teoretická východiska mé diplomové práce. Tato kapitola je členěna do tří podkapitol, kde jsou postupně vysvětleny teoretické poznatky o databázi a datovém skladu, ETL procesech, jazyku SQL a v neposlední řadě také o riziku a jeho analýzách.

1.1 TEORETICKÝ ÚVOD DO DATABÁZE A DATOVÉHO SKLADU

V následujících podkapitolách jsou objasněny potřebné teoretické poznatky o databázích a datových skladech, které budou potřebné při samotné praktické části této diplomové práce.

1.1.1 Databáze a jejich typy [3]

Databázi lze popsat jako systém, v němž jsou data ukládána a následně zpracovávána. Obsah databáze je tvořen:

- daty,
- paměťovým médiem,
- vztahy mezi daty.

Tento fakt můžeme nejsnáze ukázat na praktickém příkladu. Představme si databázi jako papírovou kartotéku u lékaře, kdy data nám představují záznamy o pacientech, paměťové médium je v tomto případě papír a vztahy mezi daty si představme jako návaznost mezi jednotlivými návštěvami lékaře u konkrétního pacienta.

Databáze můžeme rozdělit do následujících typů:

- relační databáze
- síťová databáze
- hierarchická databáze
- objektová databáze
- objektově relační databáze
- dokumentově orientovaná databáze.

1.1.2 Relační databáze [3][5][7]

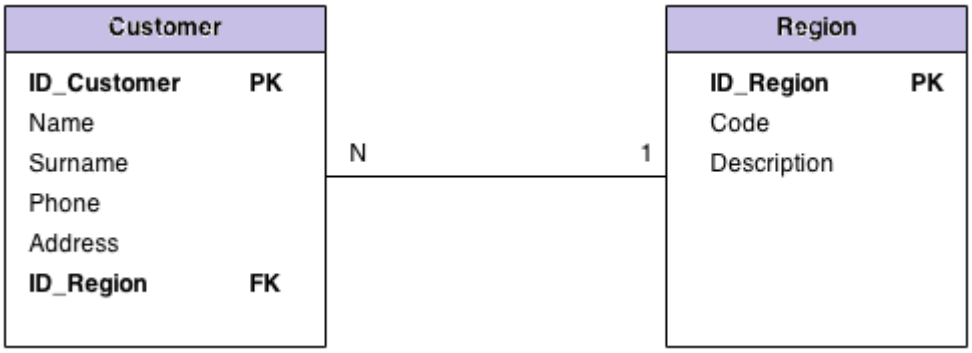
Relační databáze je tvořena několika tabulkami a vztahy mezi nimi. Jednotlivé tabulky jsou tvořeny konkrétními sloupci neboli atributy a řádky. V atributech jsou ukryty vlastnosti objektů (datové typy a délky), které budou vloženy do tabulky.

Vztahy mezi tabulkami

„Relace představují souvislosti mezi tabulkami relační databáze. Zatímco každá relační tabulka může existovat samostatně, databáze jsou především o ukládání souvisejících dat. Pomocí relací lze provázat související tabulky formálním způsobem, který je snadno použitelný, chcete-li ve stejném dotazu spojit data z více tabulek, ale flexibilně přitom zahrnout pouze informace, které vás zajímají.“ [7, s.11]

Vazba 1:N

Vazba 1:N říká, že každé n-tici (primárnímu klíči) jedné relace odpovídá právě jedna nebo více n-tic druhé relace, jak je možné vidět na obrázku č. 1.1.



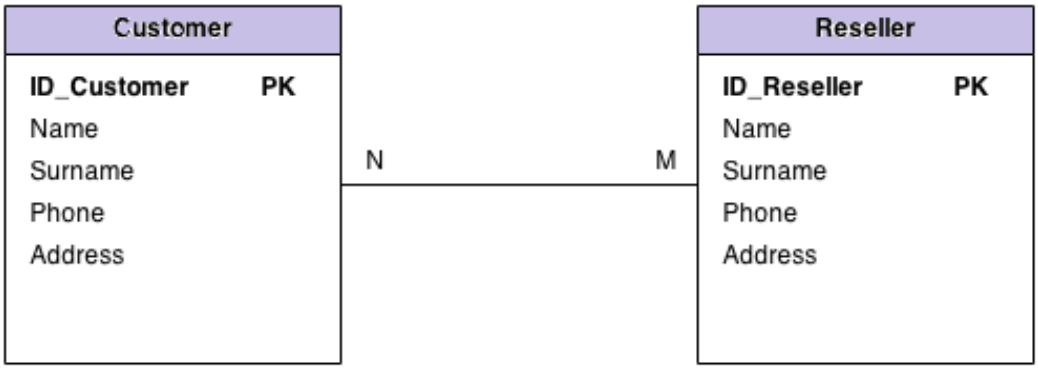
Obrázek 1.1: Vazba 1:N

Zdroj: vlastní tvorba

Vazba N:M

Následující obrázek (obrázek č. 1.2) zobrazuje relační vazbu N:M, která říká, že jedné nebo více n-ticím jedné relace odpovídá právě jedna nebo více n-tic relace druhé.

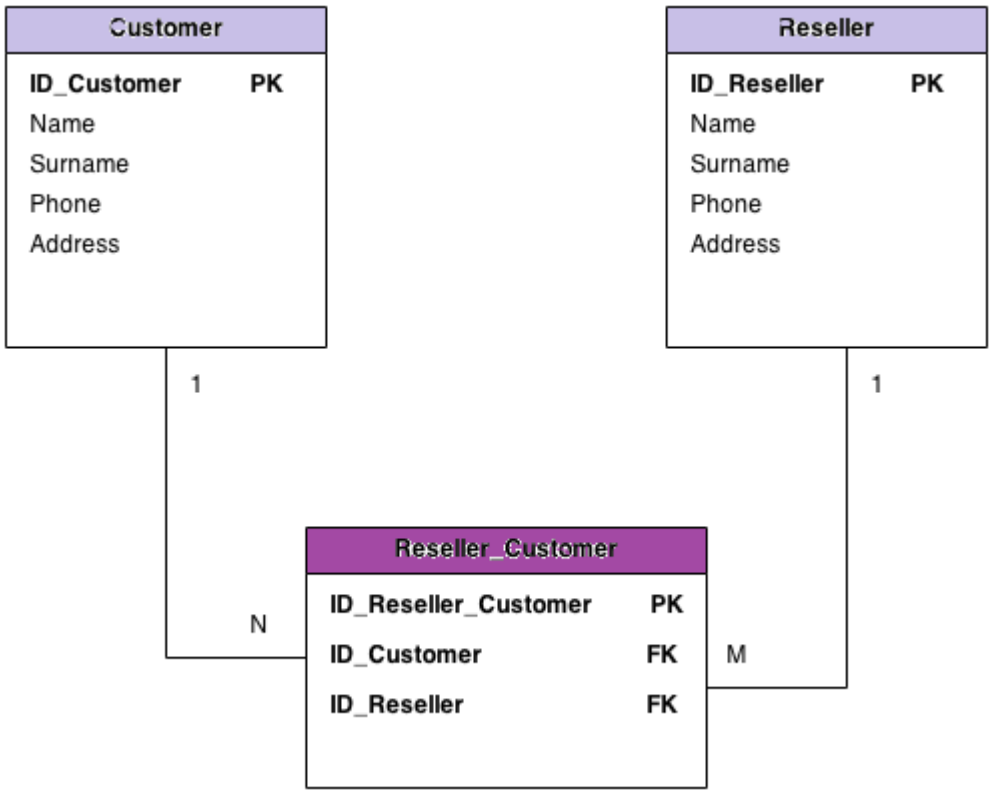
V takovéhoto typu vztahu mezi relacemi je nutné do návrhu databáze začlenit dekompozici těchto tabulek, jak je vyobrazeno na obrázku č. 1.3.



Obrázek 1.2: Vazba N:M

Zdroj: vlastní tvorba

Dekompozice se provádí vložením „propojovací“ tabulky mezi vazbu N:M, která obsahuje primární klíče z obou tabulek (v tomto případě ID_Customer a ID_Reseller). V dekompoziční tabulce tyto klíče vystupují jako cizí klíče.



Obrázek 1.3: Vazba N:M – dekompozice

Zdroj: vlastní tvorba

1.1.3 Primární a cizí klíč [3][5]

Pomocí ***primárního klíče*** dochází k označení jednoho nebo více záznamů v tabulce, což zabezpečí jedinečnost dat na každém řádku tabulky. Primární klíč může být tvořen více sloupci tabulky, v takovém případě se jedná o složený primární klíč. Užití složeného primárního klíče je například na místě v případě, kdy chceme sledovat vývoj jednotlivých záznamů v čase, jak je možné vidět v tabulce č. 1.1.

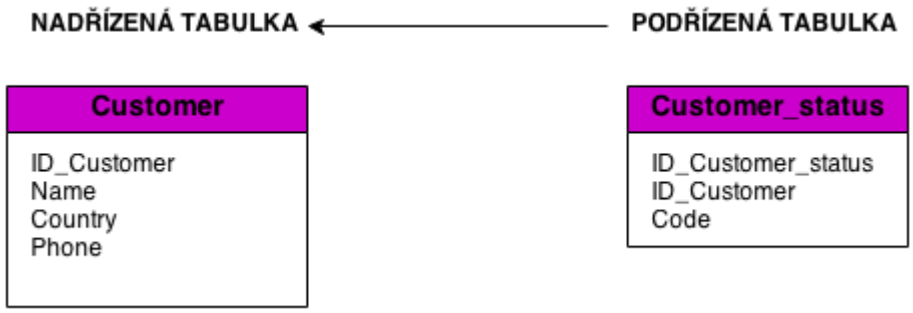
FORDATE	ID_CUSTOMER	NAME	COUNTRY	PHONE
2013-02-12	**123**	Mike Dowling	United States	666 423 001
2013-05-12	**123**	Mike Dowling	United Kingdom	666 423 001
2013-05-20	**112**	Alice West	Australia	452 214 223

Tabulka 1.1: Tabulka zákazníků obsahující složený primární klíč

Zdroj: vlastní tvorba

Z tabulky č. 1, která obsahuje složený primární klíč (FORDATE a ID_CUSTOMER), lze zjistit, že zákazník 123, Mike Dowling, žil do 12. května 2013 ve Spojených státech a poté se přestěhoval do Velké Británie.

Cizí klíč, neboli primární klíč z nadřízené tabulky, zajišťuje referenční integritu mezi dvěma tabulkami. Cizí klíč vytváří vazbu na primární klíč v jiné tabulce. Obrázek č. 1.4 zachycuje vztah nadřízenosti mezi tabulkami.



Obrázek 1.4: Vztah nadřízenosti a podřízenosti mezi tabulkami

Zdroj: Vlastní tvorba

Výše vyobrazený obrázek zachycuje vztah mezi tabulkami Customer a Customer_status, kdy tabulka Customer_status je podřízená nadřízené tabulce Customer.

Jinými slovy tabulka Customer_status přejímá primární klíč ID_Customer z tabulky Customer, pomocí něhož lze zjisti status u konkrétního zákazníka.

1.1.4 Normalizace a její formy [3][5][7][18]

Normalizace databáze představuje proces, kdy dochází k redukci opakujících se dat v databázi. Pokud dojde při návrhu datového modelu k porušení některé z normálních forem, není datový model navržen optimálně. Při procesu normalizace dochází k úpravě relací do takového tvaru, aby splňovaly podmínky bezztrátovosti při zpětném spojení, aby nedošlo k porušení závislostí a současně došlo k odstranění opakujících se záznamů. Pro určení míry databázové normalizace se používají normální formy následujících stupňů:

- první normální forma
- druhá normální forma
- třetí normální forma
- Boyce-Coddova normální forma
- čtvrtá normální forma
- pátá normální forma.

První normální forma, také nazývána jako multizávislost, říká, že relace (tabulka) splňuje podmínky první normální formy, pokud jsou veškeré atributy této tabulky definovány nad stálými (neproměnnými) hodnotami. Jinými slovy lze říci, že veškeré atributy relace musí být jednoduché.

FORDATE	ID_CUSTOMER	NAME	ADDRESS
2013-02-12	**123**	Mike Dowling	Down street 321, New York

Tabulka 1.2: Porušení první normální formy

Zdroj: vlastní tvorba

Tabulka č. 1.2 obsahuje složený atribut „ADDRESS“, čímž porušuje první normální formu. Tento atribut musí být rozdělen na dílčí atributy, jak je zachyceno v tabulce č. 1.3.

FORDATE	ID_CUSTOMER	NAME	STREET	POST CODE	CITY
2013-02-12	**123**	Mike Dowling	Down street	321	New York

Tabulka 1.3: Tabulka splňující první normální formu

Zdroj: vlastní tvorba

Druhá normální forma, někdy také nazývaná jako funkční závislost, pojednává o tom, že relace splňuje podmínky funkční závislosti v případě, kdy splňuje podmínku multizávisloti, tedy je v první normální formě, a současně jsou všechny dílčí atributy této tabulky plně funkčně závislé na celém kandidátním, respektive primárním, klíči.

Třetí normální forma jiným názvem také tranzitivní závislost je dodržena za předpokladu, že relace je v druhé normální formě a současně neklíčové atributy této relace jsou navzájem nezávislé. V druhé normální formě musí být neklíčové atributy plně funkčně závislé na celém klíči, avšak pokud je tato závislost splněna přes jiný neklíčový atribut, pak hovoříme o tranzitivní závislosti atributu na klíči.

Boyce-Coddova normální forma je představována variací třetí normální formy, kdy platí tvrzení, že relace je v Boyce-Coddově normální formě, pokud je také ve třetí normální formě, avšak obráceně je toto tvrzení vyloučeno. Přesná definice Boyce-Coddovy normální formy zní: *„Relace je v Boyce-Coddově normální formě, pokud mezi kandidátními klíči není žádná funkční závislost a to za těchto podmínek:*

- *relace musí mít dva nebo více kandidátních klíčů*
- *nejméně dva z kandidátních klíčů musí být složené*
- *kandidátní klíče se v některých atributech musí překrývat.“* [3, s. 59]

Tato normální forma je určena pouze ve specifických případech a málokdy je jich dosaženo, stejně tak i následujících dvou normálních forem.

Relace splňuje požadavky ***čtvrté normální formy*** v případě, kdy splňuje požadavky Boyce-Coddovy normální formy a současně jsou veškeré vícehodnotové závislosti funkčně závislé na kandidátních klíčích.

*„Relace je v **páté normální formě** jestliže je ve čtvrté normální formě a nemůže-li být dále bezeztrátově rozložena. Jinými slovy relace, která má n klíčových atributů (n >= 3) a*

která se rozloží na relace o n-1 klíčových atributech, nemůže být opětovně spojena operací přirozeného spojení do jedné relace, aniž by došlo ke ztrátě informace.“ [18]

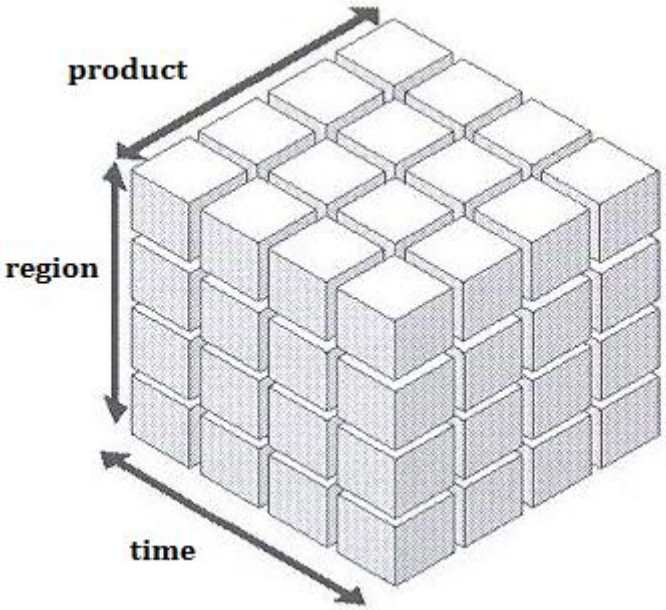
1.1.5 Multidimenzionální databáze a multidimenzionální databázový model [4][5]

Multidimenzionální databáze se využívá pro získávání sumarizovaných a agregovaných informací. Tyto databáze poskytují rychlý a ucelený přístup k datům, která mohou být dále použita pro modelování a komplexní analýzy. V multidimenzionální databázi se na rozdíl od relační databáze používají výhradně nenormalizované tabulky, které jsou rozděleny na dva druhy a to tabulky faktů a tabulky dimenzí.

Tabulky faktů obsahují analyzovaná data, jako například číselné a finanční hodnoty, které jsou dále používány k analytickým výpočtům, tříděním atd. Tyto tabulky obsahují detailní údaje pro veškeré zdroje a jsou pomocí cizích klíčů propojeny s dimenzemi.

Dimenze jsou představovány seznamy hodnot, které se používají ke třídění a za kategorizování dat ve faktových tabulkách.

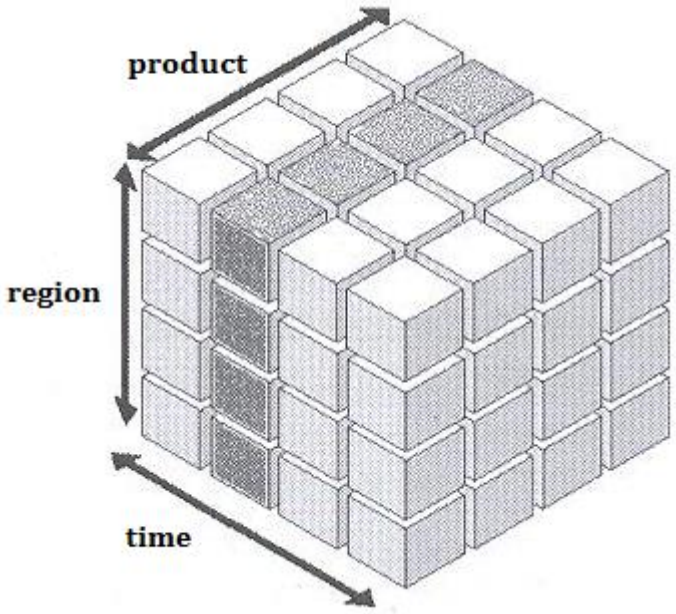
Multidimenzionální datová struktura, někdy nazývána jako multidimenzionální datová kostka, je výsledkem agregace a analýzy dat, která jsou většinou organizována v relační databázi obsahující dvojrozměrné relační tabulky. Multidimenzionální datová kostka představuje rovnocennou náhradu tabulky v relační databázi. Analytické databáze, často nazývané také OLAP (Online Analytical Processing), obsahují datové struktury a služby pro analýzu, pomocí nichž dochází k rozboru velkého počtu dat, jejichž výstupem jsou tzv. reporty používané pro budoucí rozhodování a plánování. Každá multidimenzionální kostka obsahuje několik dimenzí. Na obrázku č. 1.5 je vyobrazen princip trojdimenzionální kostky, přičemž dimenzemi jsou čas, region a produkt.



Obrázek 1.5: Princip trojdimenzionální kostky

Zdroj: [5]

V multidimenzionálních kostkách lze sledovat data dle definovaných kritérií. Dle kostky, zachycené na obrázku 1.6, lze sledovat vývoj dat v konkrétním časovém období. Tato informace je důležitá například pro vyhodnocení výrobních a prodejních plánů. Díky této analýze jsme schopni říci, zda firma vyrábí dostatečné množství pro prodej.



Obrázek 1.6: Analýza dat pro určité časové období

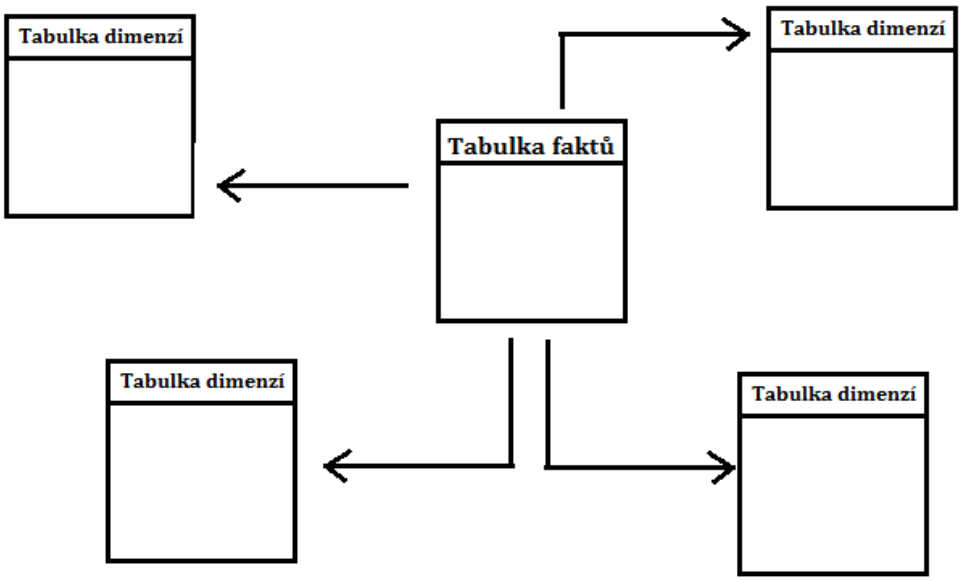
Zdroj: [5]

1.1.6 Ukládání dat do multidimenzionálních databází [5]

Schématu neboli dimenzionálního modelu je využíváno pro tvorbu multidimenzionální OLAP. Lze rozlišit tři druhy dimenzionálních schémat, a to:

- hvězdicové schéma
- schéma sněhové vločky
- souhvězdí.

Nejčastěji používaným schématem je právě schéma hvězdy, někdy také nazývané jako star schema. Hvězdicové schéma z pravidla tvořeno jednou faktovou tabulkou a několika dimenzemi. Faktová tabulka obsahuje cizí klíče, pomocí nichž je propojena s tabulkami dimenzí, jak je vyobrazeno na obrázku č. 1.7.

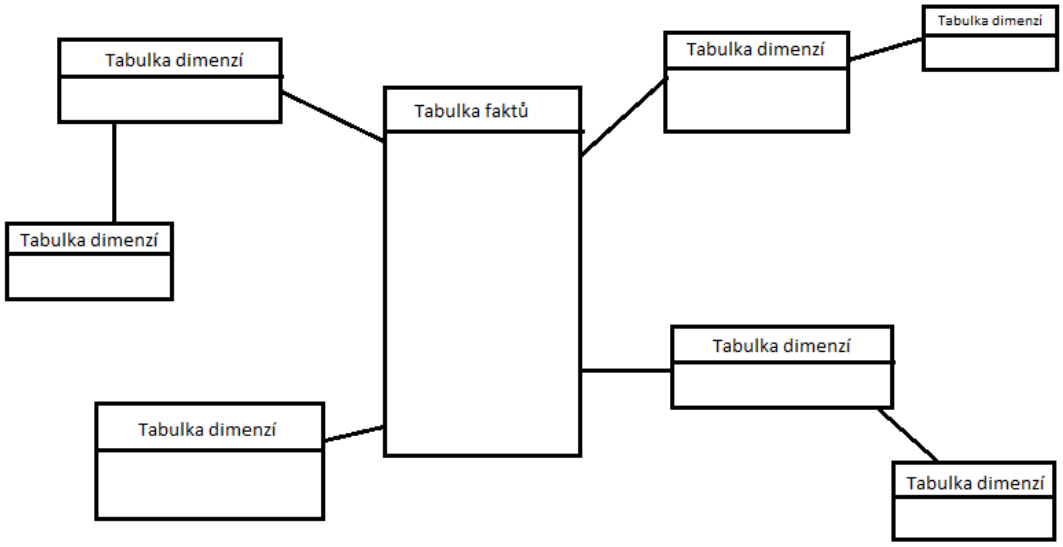


Obrázek 1.7:Hvězdicové schéma

Zdroj: vlastní tvorba

Schéma sněhové vločky, také nazývané jako snow flag schema, obsahuje dimenze, které jsou složené z více tabulek. Toto dimenzionální schéma slouží k rychlejšímu zavádění dat do tabulek splňující podmínky normalizace. Nevýhodou schématu sněhové vločky oproti hvězdicovému schématu je velké množství navzájem propojených tabulek, což snižuje dotazovací výkon.

Pokud dojde k rozdělení tabulky faktů, avšak toto rozdělení je v souladu s hierarchiemi dimenzí, vzniká tzv. schéma souhvězdí. Souhvězdicové schéma, viz obrázek 1.8, je tvořeno několika tabulkami faktů, které se odkazují do stejných dimenzí, ale jsou vždy propojeny pomocí jiných atributů.



Obrázek 1.8: Schéma sněhové vločky

Zdroj: vlastní tvorba

1.1.7 Ukládání multidimenzionálních dat MOLAP, ROLAP a HOLAP [5][7]

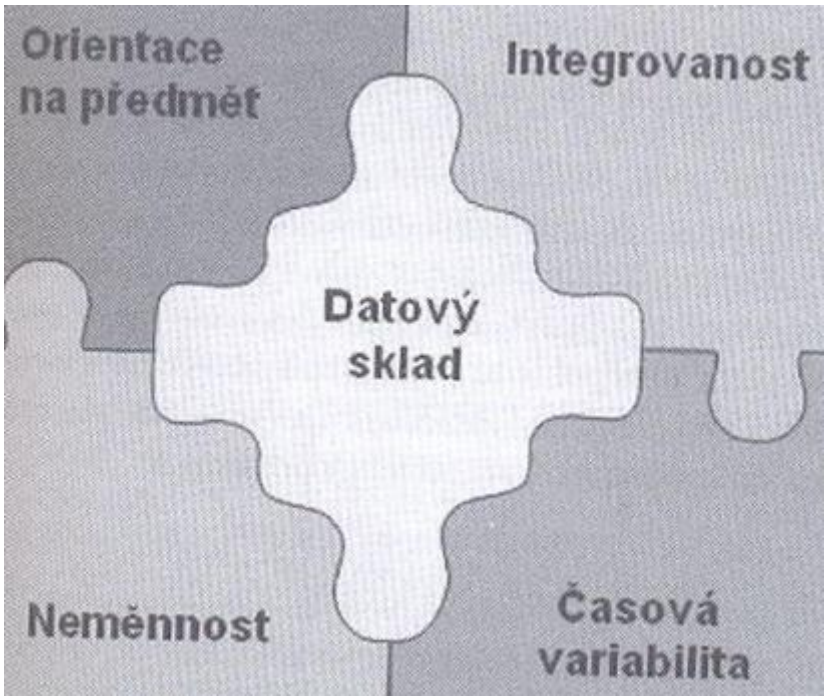
Pomocí dat získaných z datového skladu nebo z operačních zdrojů můžeme provádět multidimenzionální online analytické zpracování, zkráceně nazývané MOLAP. Tento mechanizmus ukládá analytická data ve formátu vlastních datových struktur, případně sumářích, z čehož vyplývá, že data jsou ukládána jako předem vypočtená pole. Multidimenzionální databáze umožňuje rychlé získávání požadovaných dat.

„Relační online analytické zpracování dat (ROLAP) získává data pro analýzy z relačního datového skladu. Tato data z relačních databází se po zpracování předkládají uživateli jako multidimenzionální pohled.“ [7, s. 181] Na rozdíl od multidimenzionální OLAP zůstávají data uložena v relační databázi, proto nedochází k redundanci dat.

Hybridní OLAP vzniká kombinací multidimenzionálního a relačního OLAP úložiště, přičemž data se ukládají v relačních databázích, ale sumarizovaná data se ukládají do multidimenzionálních struktur.

1.1.8 Datový sklad [4][5][15]

Datový sklad, také nazývaný jako data warehouse, lze definovat jako systém umožňující organizaci, uchovávání, sdílení a nepopiratelně shromažďování historických dat. Bill Inmon definoval datový sklad jako: *„Podnikově strukturovaný depozitář subjektově orientovaných, integrovaných, časově proměnných, historických dat použitých pro získávání informací a podporu rozhodování. V datovém skladu jsou uložena atomická a sumární data.“* [5] Tuto známou a výstižnou definici datového skladu lze jednoduše graficky znázornit, jak je možné vidět na obrázku č. 1.9.



Obrázek 1.9: Grafické znázornění definice datového skladu dle Billa Inomona

Zdroj: [5]

Pokud je orientace zaměřena na subjekt, pak dochází ke kategorizaci dat v datovém skladu na základě subjektu. Podmínka integrovanosti je splněna za předpokladu, že jsou data do datového skladu vložena právě jednou, proto je nezbytně nutné zavést výstižnou a jednotnou terminologii. V datovém skladu uložená data vystupují jako série snímků představující určitý časový úsek. Díky tomuto způsobu ukládání splňuje datový sklad charakteristiku časové variability. Pokud jsou data již uložena v datovém skladu, nelze je měnit či mazat, ale naopak jsou data v pravidelných intervalech přihrávána.

1.2 TEORETICKÝ ÚVOD DO ETL PROCESŮ [15][19]

Jedním z druhů ukládání dat do datových skladů je proces ETL, který se skládá z následujících kroků:

- extraktce (extraction)
- transformace (transformation)
- nahrávání (load).

Tento proces lze považovat za jakési jádro datového skladu. V první části tohoto procesu, nazývané extrakce, dochází ke čtení dat z jednoho nebo více zdrojových systémů. V druhém kroku neboli transformaci, dochází k provedení potřebných změn v datech do takové podoby, aby mohla být uložena do databáze. Ve třetí části ETL procesu dochází k nahrání (uložení) požadovaných dat do datového skladu.

Nejčastěji bývají ETL procesy budovány pomocí skriptů v některém z běžně používaných skriptovacích jazyků, jako například SQL, C/C++, Perl a podobně. V následující kapitole bude blíže představen jazyk SQL.

Dalším způsob, jak budovat ETL procesy, je pomocí využití Microsoft Visual Studia. MS Visual Studio je balík obsahující nástroje a služby určené pro vývoj softwarových aplikací, tedy i ETL procesů. Kapitola 1.4 je věnována teoretickému úvodu do používání MS Visual Studia pro vývoj ETL procesů.

Jako vše, i ETL procesy mají své světlé i stinné stránky. Mezi výhody používání ETL procesů můžeme zahrnout například flexibilitu, kdy lze velmi jednoduše dílčí procesy modifikovat, rozšiřovat atd. Další výhodou je výkon, kdy ETL procesy jsou definovány tak, aby optimálně využívaly dostupný hardware a tak dosahovaly maximálního výkonu. Mezi neopomenutelné výhody můžeme také začlenit schopnost podpory metadat, kdy ETL nástroje jsou schopné intenzivně pracovat a využívat metadata, která si můžeme představit jako informace o zdrojových objektech.

Mezi nevýhody ETL procesů můžeme zahrnout vysokou náklonost k riziku chybování při vytváření, udržování a správě skriptů. Psaní kódu je značně vyčerpávající a časově náročné a vyžaduje velké soustředění a zkušenosti vývojáře. Proto lze velmi zjednodušeně říci, že značnou nevýhodou může být v tomto případě lidský faktor.

1.2.1 Extrakce [20]

Proces extrakce je část ETL procesů, ve které dochází k získávání přístupů ke zdrojům dat. V této fázi je velmi důležité definování, aby byl definován formát a struktura dat, ve které budou exportovány na předem domluvené úložiště. Nezbytnou součástí tohoto procesu je správné porozumění datům, abychom byli schopni v dalších fázích s nimi správně pracovat. Data, s nimiž budeme dále pracovat, mohou být v textovém formátu jako například formátu .csv, ve formátu XML nebo mohou pocházet z různých kancelářských aplikací.

Důležitou součástí procesu extrakce je i ověření, zda jsou data v požadované kvalitě. Proto v běžné praxi dochází k nahrávání dat do tzv. source databází, kde pro jednotlivé atributy jsou navrženy předem domluvené datové typy a jejich délky. V této části dochází také k ověření, zda atribut musí být vyplněn či nikoliv.

Pokud jsou data dostupná na zdrojovém serveru a současně jsou kvalitní, dochází k jejich nahrání do databáze, která bývá v praxi nazývána stage. Na tomto místě jsou data upravována do požadované podoby neboli transformace.

1.2.2 Transformace [20]

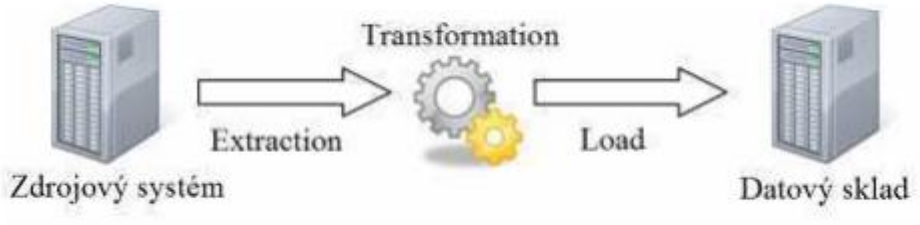
Ve fázi transformace dochází k jednoduchým úpravám, kterými může být úprava datového typu, případně normalizace atributů. Změna datového typu bývá prováděna nejčastěji u atributů časového a datumového charakteru, kdy bývá použit datový typ date, někdy datetime. Pod pojmem normalizace si lze představit sjednocení hodnot typu True/False, Yes/No. Při normalizaci se rozhodujeme pro jeden typ hodnoty, který používáme napříč celou databází.

Často bývá požadavkem začlenit do databází hodnoty, které jsou dopočítávány ze zdrojových dat. V praxi bývají běžně kladeny požadavky na reporting finančních částek v různých měnách. Proto v průběhu procesu transformace dochází k přepočtu zdrojových dat dle předem domluveného kurzu do různách měn.

Ve fázi transformace také dochází ke kontrole datové kvality a dochází k odstranění duplicitních hodnot. Porušení datové kvality může být způsobeno pochybením lidského faktoru, případně poruchou systému.

1.2.3 Loadování [20]

Fáze loadování je fáze, kdy dochází k nahrání již upravených dat do cílové databáze, která je předem připravena a je v předem definované databázové struktuře. Data jsou plněna do takzvaných tabulek faktů a dimenzí, které jsou zdrojem pro vytváření datových kostek.



Obrázek 1.10: ETL proces

Zdroj: [20]

1.3 TEORETICKÝ ÚVOD DO JAZYKU SQL

V následujících podkapitolách je představen programovací jazyk SQL, jeho historie a základní programovací klauzule, které je nutné ovládat při tvorbě ETL procesů.

1.3.1 Jazyk SQL a jeho historie[5][7][10]

Structured Query Language, zkráceně SQL, je programovací jazyk, který se využívá pro komunikaci s databází, konkrétně s relační databází. Obecně lze říci, že SQL se využívá ke správě a údržbě těchto databází. Tento programovací jazyk nelze řadit mezi procedurální jazyky, z čehož vyplývá, že uživatel prostřednictvím tohoto jazyka sdělí počítači jaké chce získat výsledky. Jazyk SQL je tvořen několika částmi, které lze definovat následovně:

- jazyk Data Definition Language
- jazyk Data Query Language
- jazyk Data Manipulation Language
- jazyk Data Control Language
- příkazy řízení transakcí.

Jednotlivé kategorie jsou detailněji popsány v následujících podkapitolách.

V 70. letech minulého století přišla společnost IBM s experimentální relační databází, kterou nazvali System/R. Tato experimentální databáze vychází z poznatků práce Dr. E. F. Codda. Doktor Edgar F. Codd byl anglický počítačový specialista pracující pro společnost IBM. Jeho profesní kariéře je připisováno vynalezení modelu relační databáze.

„Společnost IBM sice přišla s první implementací SQL, ale na trhu ji předstihly dva jiné produkty, které zahrnovaly dotazovací jazyky s odlišnými názvy. Prvenství mezi komerčními relačními databázemi si tedy vydobyly Oracle společnosti Relation Software a INGRES od firmy Relational Technology. Společnost IBM v roce 1982 vydala databázi SQL/DS, jejíž dotazovací jazyk byl pojmenován SQL (Structured Query Language).“[7, s. 37]

1.3.2 Data Definition Language [7]

Jazyk Data Definition Language, zkráceně DDL, obsahuje příkazy pro vytváření nových databázových objektů a úpravu struktur stávajících objektů. Příkazy jazyka DDL, která jsou reprezentována slovy CREATE, ALTER nebo DROP, nemají vliv na data, jenž jsou uložena v databázi, nýbrž na kontejnery, ve kterých jsou data uložena.

1.3.3 Příkazy jazyku DDL [7]

Příkazy výše zmiňovaného jazyka neumožňují upravovat ani vkládat data do databázových objektů, ale vytvářejí nebo upravují samotné databázové objekty.

Příkaz CREATE

Obecně se příkaz CREATE používá k vytváření objektů jako je např. databáze, tabulka, pohled nebo index. Následující syntaxe zachycuje příklad vytvoření databáze.

```
CREATE DATABASE název_databáze [specifické_parametry_dodavatele];
```

Jedním z dalších příkazů je příkaz pro vytvoření tabulky. Obecně lze definovat syntaxi takto:

```
CREATE TABLE název_tabulky
       (název_sloupce datový_typ (délka datového typu) NULL / NOT NULL,
        název_sloupce datový_typ (délka datového typu) NULL / NOT NULL,
        název_sloupce datový_typ (délka datového typu) NULL / NOT NULL,
        název_sloupce datový_typ (délka datového typu) NULL / NOT NULL);
```

V rámci syntaxe pro vytvoření nové tabulky lze definovat i primární a cizí klíč. Primární klíč lze definovat dvěma různými způsoby.

```
CREATE TABLE název_tabulky
       (název_sloupce datový_typ (délka datového typu) NULL / NOT NULL PRIMARY KEY,
        název_sloupce datový_typ (délka datového typu) NULL / NOT NULL,
        název_sloupce datový_typ (délka datového typu) NULL / NOT NULL,
        název_sloupce datový_typ (délka datového typu) NULL / NOT NULL);


CREATE TABLE název_tabulky
       (název_sloupce datový_typ (délka datového typu) NULL / NOT NULL,
        název_sloupce datový_typ (délka datového typu) NULL / NOT NULL,
        název_sloupce datový_typ (délka datového typu) NULL / NOT NULL,
        název_sloupce datový_typ (délka datového typu) NULL / NOT NULL,
        PRIMARY KEY (název_sloupce));
```

„Indexy představují silný nástroj, protože umožňují systému řízení báze dat značně urychlit vyhledávání dat – podobně jako rejstřík na konci knihy pomáhá rychle najít požadované téma. Indexy pro sloupce cizích klíčů mohou navíc výrazně zvýšit rychlost spojování tabulek. Na druhou stranu však indexy zabírají místo v databázi a vyžadují údržbu. Pokaždé, když se změní hodnota sloupce, na kterou se index odkazuje, je nutné upravit i vlastní index.“ [7, s. 62] Definování příkazu pro vytvoření indexu je mnohem jednodušší než vytvoření tabulky.

```
CREATE INDEX název_indexu ON název_tabulky (
       název_sloupce);
```

Při vytváření indexů lze také definovat, v jakém pořadí bude index ve sloupci růst. Klíčové slovo ASC definuje vzestupný růst indexu. Požadujeme-li sestupný růst indexů, lze použít klíčové slovo DESC.

```
CREATE INDEX název_indexu ON název_tabulky (
       název_sloupce ASC/DESC);
```

V průběhu tvorby indexů, lze také použít klíčové slovo UNIQUE, které deklaruje unikátní index. Pomocí užití unikátního indexu může dojít k omezení, aby žádné dva řádky v tabulce neměli shodnou kombinaci hodnot sloupce.

```
CREATE UNIQUE INDEX název_indexu ON název_tabulky (
       název_sloupce ASC/DESC);
```

Pohledy neboli view umožňují uživateli přizpůsobit data zanesená do databáze individuálním požadavkům uživatele. Pohledy nevyžadují ukládání dat. Pod pojmem pohled si lze představit dotaz, na který se můžeme pomocí Data Manipulation Language nebo Data Query Language odkazovat tak, jako na existující tabulku v databázi.

```
CREATE VIEW název_pohledu AS
       SELECT název_sloupce
       FROM název_tabulky
              WHERE podmínka
```

Příkaz ALTER table

Pomocí příkazu ALTER TABLE dochází k modifikaci již vytvořené tabulky. Tento příkaz je používán z praktických důvodů, v současnosti je kladen důraz na nepřetržitý provoz databáze a mimořádně rychlý přírůst dat. Proto není moc praktické odstraňovat a znovu vytvářet již existující tabulky.

Příkaz ALTER TABLE může být také používán pro přehlednost. Mnoho správců databází upřednostňuje jednoduchý příkaz CREATE TABLE a doplňují příkaz ALTER TABLE pomocí něhož vytvářejí např. primární a cizí klíče. Nevýhodou tohoto použití je příliš dlouhý zdrojový kód.

Příkaz DROP

Pomocí příkazu DROP může dojít k odstranění objektů, jakými jsou například index, tabulka nebo pohled. Tento příkaz je jedním z nejjednodušších příkazů v jazyce Data Definition Language. Syntaxe vypadá následovně:

```
DROP TABLE název_tabulky;

DROP INDEX název_indexu;
```

1.3.4 Data Query Language [7]

Pomocí jazyka Data Query Language neboli DQL dochází k načítání dat z databáze. Tento jazyk obsahuje jediné klíčové slovo, a to SELECT. Toto volání dat z databáze může být rozšířeno o podmínky, kterými jsou například WHERE, WHERE EXIST atd.

1.3.5 Data Manipulation Language [7]

Data Manipulation Language jazyk se běžně používá pro modifikaci již zanesených dat v databázi. Tento jazyk obsahuje příkazy, kterými jsou INSERT, UPDATE a DELETE.

Příkaz INSERT

Příkazu INSERT je využíváno pro přidávání nových datových řádků do tabulek. Tento příkaz může být použit ve dvou základních formách. První formou je vkládací příkaz obsahující hodnoty sloupců ve vlastním příkazu. Syntaxe je následující:

Druhým typem příkazu INSERT je příkaz s použitím vnořeného SELECTu. Syntaxe tohoto typu příkazu je následující:

```
INSERT INTO název_tabulky
          (nazev_sloupce, nazev_sloupce)
VALUES (hodnota, hodnota)
```

Druhým typem příkazu INSERT je příkaz s použitím vnořeného SELECTu. Syntaxe tohoto typu příkazu je následující:

```
INSERT INTO název_tabulky
          (nazev_sloupce, nazev_sloupce)
SELECT (nazev_sloupce, nazev_sloupce)
   FROM nazev_tabulky
```

Příkaz UPDATE

Příkazu UPDATE se využívá v případě, kdy je potřeba upravit záznam, který je již registrován v databázi. Obecná syntaxe příkazu je následující:

```
UPDATE název_tabulky
   SET nazev_sloupce = výraz
```

Příkaz DELETE

„Příkaz DELETE odebere jeden nebo více řádků z tabulky. Příkaz může také odkazovat na pohled, ale musí se jednat o pohled založený na jediné tabulce (tzn. Příkaz DELETE nemůže odkazovat na pohledy, které obsahují spojení). Příkaz DELETE nikdy neodkazuje na sloupce, protože odebírá celé řádky včetně všech datových hodnot na každém zpracovávaném řádku.“ [7, s. 139]

1.3.6 Data Control Language [7]

Data Control Language zkráceně DCL je jazyk, který umožňuje řízení přístupů k databázím, tedy i k datům samotným. Tento jazyk také umožňuje správcům databáze použití různých systémových opatření, kterými jsou například spuštění , případně vypnutí databáze.

2 TEORETICKÝ ÚVOD DO ANALÝZY RIZIK

Tato kapitola obsahuje vysvětlení jednotlivých pojmů, které jsou úzce spjaty s riziky a analýzou rizik. Následně tato kapitola obsahuje podrobné vysvětlení vybraných metod analýzy rizik.

2.1 TEORETICKÝ ÚVOD DO PROBLEMATIKY RIZIK

Následující kapitola je věnována vysvětlení základních pojmů, které jsou nezbytně nutné k porozumění jednotlivých metod analýzy rizik a výpočet očekávaných ročních ztrát.

2.1.1 Riziko a základní pojmy s ním spjaté [9][11]

Riziko jako takové lze definovat jako vysokou míru nezdaru či nebezpečí, neboli hrozba představuje riziko. Riziko můžeme také chápat jako zranitelné místo zkombinované s hrozbou, jak je patrné na obrázku 2.1. Rizika lze rozdělit na následující druhy:

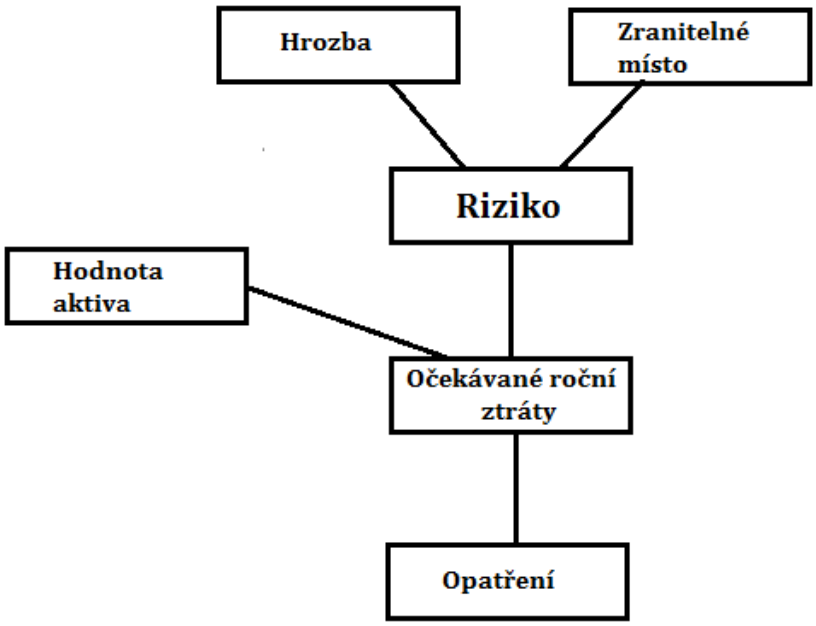
- systematické riziko
- nesystematické riziko
- systémové riziko
- finanční riziko
- přírodní riziko.

Systematické riziko je nerozlišitelné riziko, které je spjaté s celým trhem, oproti tomu nesystematické riziko se soustředí na konkrétní investice. Systémové riziko představuje hrozbu kolapsu celého trhu nebo celého finančního systému. Finanční riziko představuje hrozbu, že pravděpodobnost návratnosti investice bude jiná, z pravidla nižší, než očekávaná návratnost investic. Přírodní riziko zahrnuje přírodní katastrofy nebo živelné pohromy.

Zranitelné místo je představováno slabinou v informačním systému, která může být využita k potenciálnímu útoku. Tyto slabiny vznikají v důsledku chyb, které vznikly při návrhu nebo implementaci informačního systému. Základ zranitelného místa může být:

- fyzický
- přírodní
- v hardwaru (softwaru)
- fyzikální
- lidský faktor.

Slabiny neboli zranitelná místa nejčastěji vznikají v důsledku pochybení při návrhu (specifikaci požadavků) informačního systému. Časté chybování nastává také ve fázi řešení neboli konstrukci projektu informačního systému. Zřídkakdy vzniká slabina v informačním systému přímo za provozu.



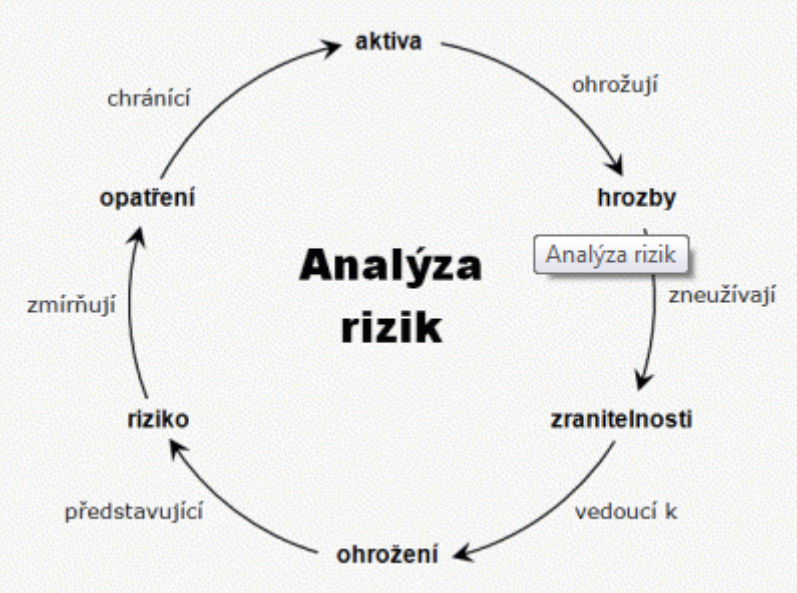
Obrázek 2.11: Proces analýzy rizik

Zdroj: vlastní tvorba

Hrozba představuje možnost využití zranitelného místa informačního systému, čímž vznikne riziko, které snižuje hodnoty aktiv. Hrozby lze rozdělit do dvou skupin, a to na objektivní a subjektivní. Objektivní hrozby obsahují následující podkategorie přírodní, fyzické, fyzikální a technické hrozby. Subjektivní hrozby neboli hrozby, které vyplynuly z pochybení lidského faktoru, lze rozdělit do následujících podkategorií, a to neúmyslné a úmyslné.

Pod pojmem aktivum si lze představit vše, co má nějakou hodnotu a je potřebné chránit patřičnými opatřeními před potenciálním rizikem.

Často bývá zaměňován pojem riziko s pojmem hrozba. Důležité si je vždy uvědomit, že hrozba jako taková nikdy riziko nepředstavuje. Hrozby využívají zranitelná místa k vytvoření ohrožení, z čehož vzniká riziko. Vzniklá rizika jsou snižována opatřeními, která chrání aktiva před znehodnocením jejich hodnoty. Tento „koloběh“ je vyobrazen na obrázku číslo 2.2.



Obrázek 2.12: Koloběh analýzy rizik

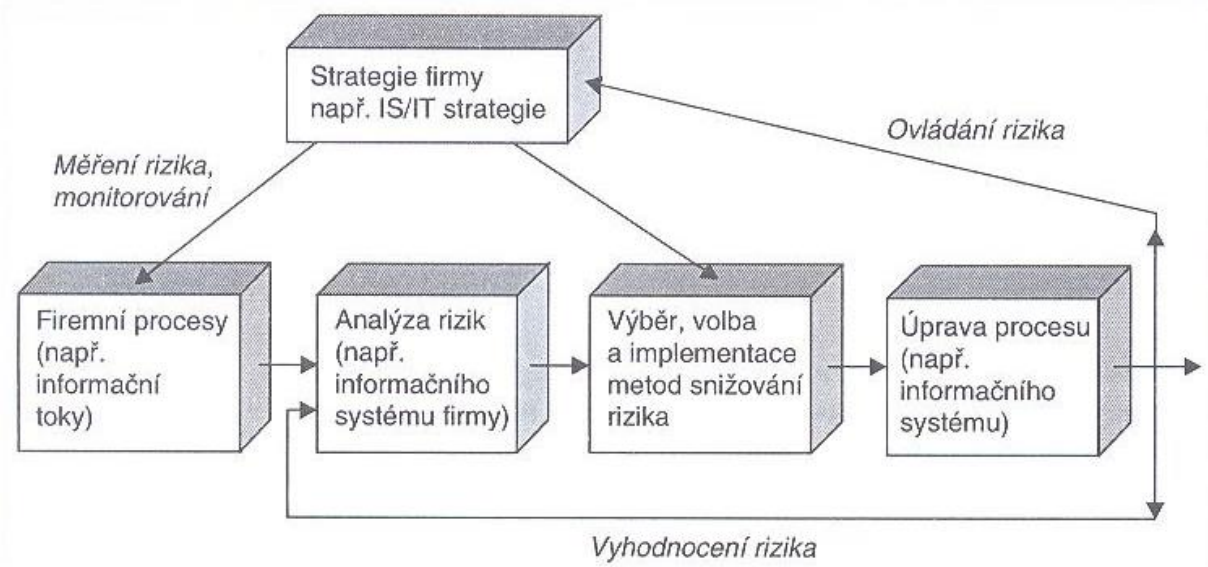
Zdroj: [11]

2.2 TEORETICKÝ ÚVOD DO ANALÝZY A ŘÍZENÍ RIZIK

V následujících podkapitolách jsou uvedeny teoretické podklady pro řízení rizik a vybrané metody analýzy rizik. Tento podklad slouží pro nastudování potřebné problematiky, pro provedení následné analýzy rizik procesu extrakce dat.

2.2.1 Řízení rizik [9][12]

Při řízení rizik dochází ke snaze o zamezení působení rizik, která již existují nebo by se mohla v budoucnu potenciálně vyskytnout. Součástí takovéhoto řízení je návrh preventivních opatření, která jsou nápomocna při eliminaci účinku nežádoucích vlivů.



Obrázek 2.13: Proces řízení rizik IS/IT ve firmě

Zdroj: [12]

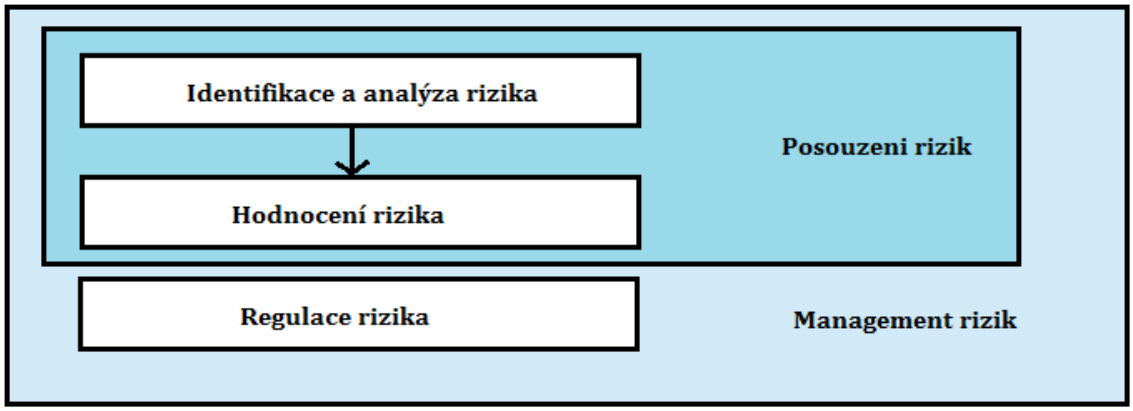
Hlavním cílem řízení rizik je snaha o snížení nákladů, které jsou vynaloženy při výskytu rizik. Tento proces lze shrnout do tří kroků, kdy v prvním kroku dochází k analýze a hodnocení rizika. V druhém kroku dochází k rozhodnutí, zda odhalená rizika jsou přijatelná nebo nepřijatelná. Toto rozhodnutí je velice problematické a závislé na konkrétní situaci. Tento problém lze eliminovat zavedením dvou úrovní a to úrovní, kdy je riziko zanedbatelné a naopak, kdy riziko už zanedbatelné není. Mezi hranicí, kdy je riziko přijatelné a nepřijatelné, lze definovat oblast, kdy je riziko přijatelné s určitým opatřením. Opatření, která jsou navržena na snížení rizik, jsou dále prozkoumána z ekonomického, politického a

sociálního pohledu. Ve třetím, tedy konečném kroku, se provádí rozhodnutí o provedení opatření ke snížení rizik a následné sledování konkrétního problému.

Konečným výsledkem každé etapy rizikového řízení je rozhodnutí o řešení, přičemž ve většině případů je navrženo několik variant takovéhoto řešení. Pokud riziko bylo zařazeno do nepřijatelné úrovně, následuje pozastavení probíhajícího procesu a přijetí preventivních opatření. Pokud se jedná o přijatelné riziko, avšak nelze říci bezvýznamné, dochází k vypracování plánu a definici preventivních opatření, která by měla být navržena tak, aby vedla k redukci tohoto rizika. Pro rizika, u kterých nelze efektivně eliminovat nebo alespoň snížit riziko, jsou vypracovávány krizové plány.

2.2.2 Analýza rizik [12]

Analýze rizik můžeme připsat prvenství v žebříčku procesu snižování rizik. Tato analýza je sestavena z procesu definování hrozeb a vyčíslení pravděpodobnosti jejich výskytu a následného vyhodnocení dopadů na aktiva, jak je vyobrazeno na obrázku č. 2.3.



Obrázek 2.14: Proces odhalení a regulace rizik

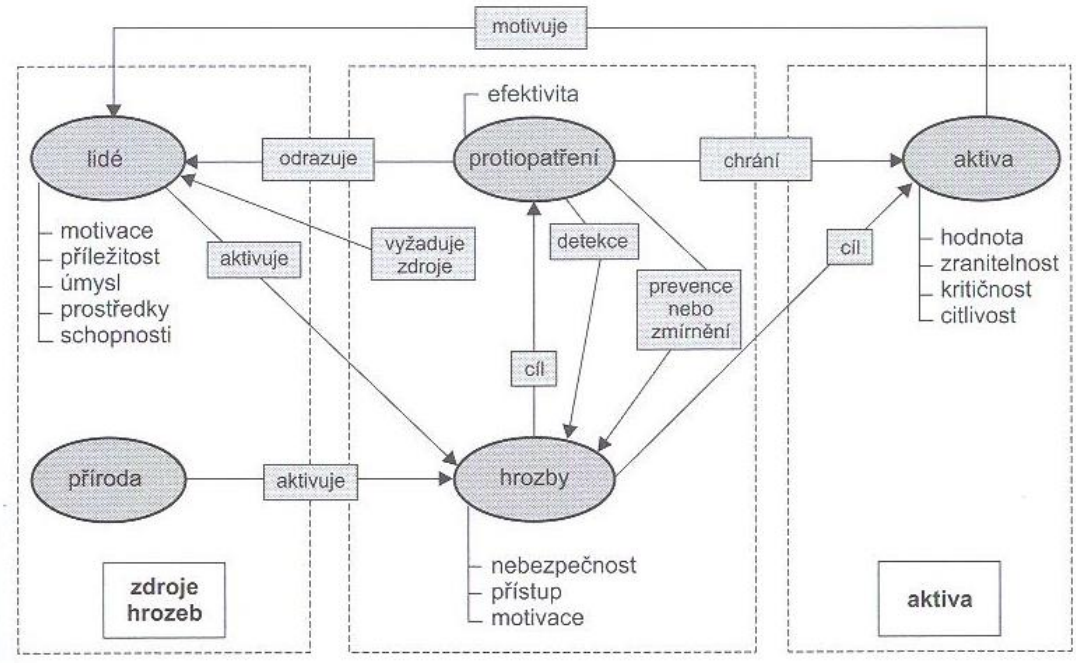
Zdroj: vlastní tvorba

„Analýza rizik zpravidla zahrnuje:

1. *Identifikace aktiv – vymezení posuzovaného subjektu a popis aktiv, které vlastní,*
2. *Stanovení hodnoty aktiv – určení hodnoty aktiv a jejich význam pro subjekt, ohodnocení možného dopadu jejich ztráty, změny či poškození na existenci či chování subjektu.*

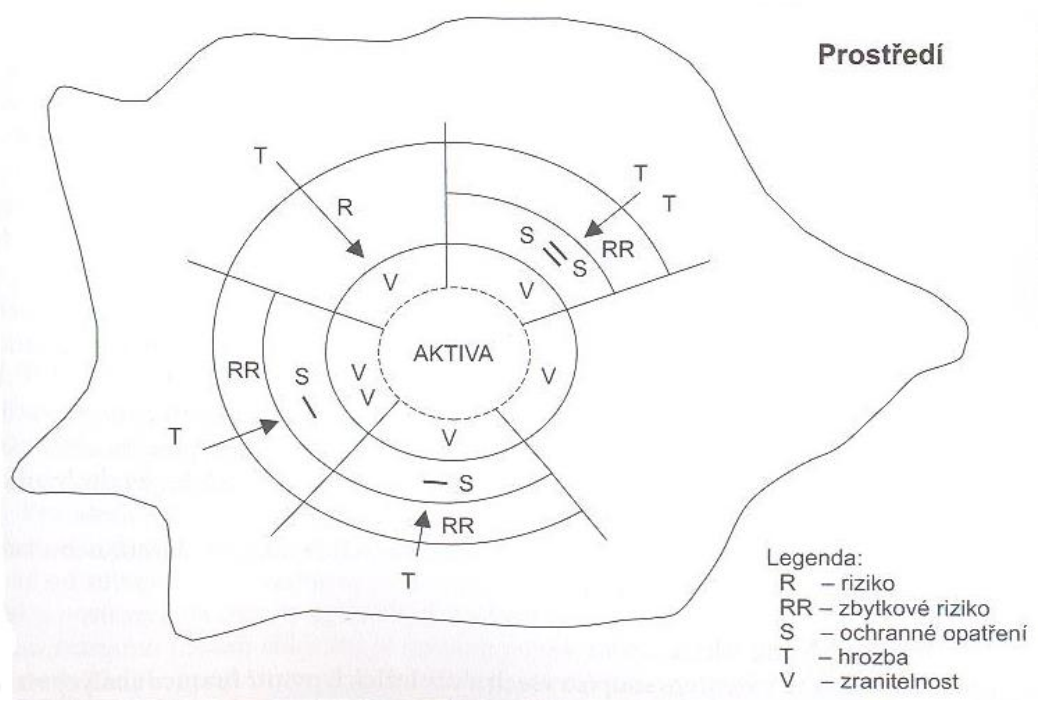
3. *Identifikaci hrozeb a slabin (zranitelnosti) – určení druhů událostí a akcí, které mohou ovlivnit negativně hodnotu aktiv, určení slabých míst subjektu, která mohou umožnit působení hrozeb.*
4. *Stanovení závažnosti hrozeb a míry zranitelnosti – určení pravděpodobnosti výskytu hrozby a míry zranitelnosti subjektu vůči dané hrozbě.“* [12, str. 95]

Důležitým milníkem v bezchybném a úspěšném provedení analýzy rizik je správné pochopení vztahů mezi prvky. Obrázek číslo 2.4 zachycuje základní vztahy mezi prvky. Obrázky číslo 2.5 a 2.6 zachycují různé typy modelů, pomocí nichž lze vyobrazit vztahy mezi konkrétními prvky analýzy a řízení rizik.



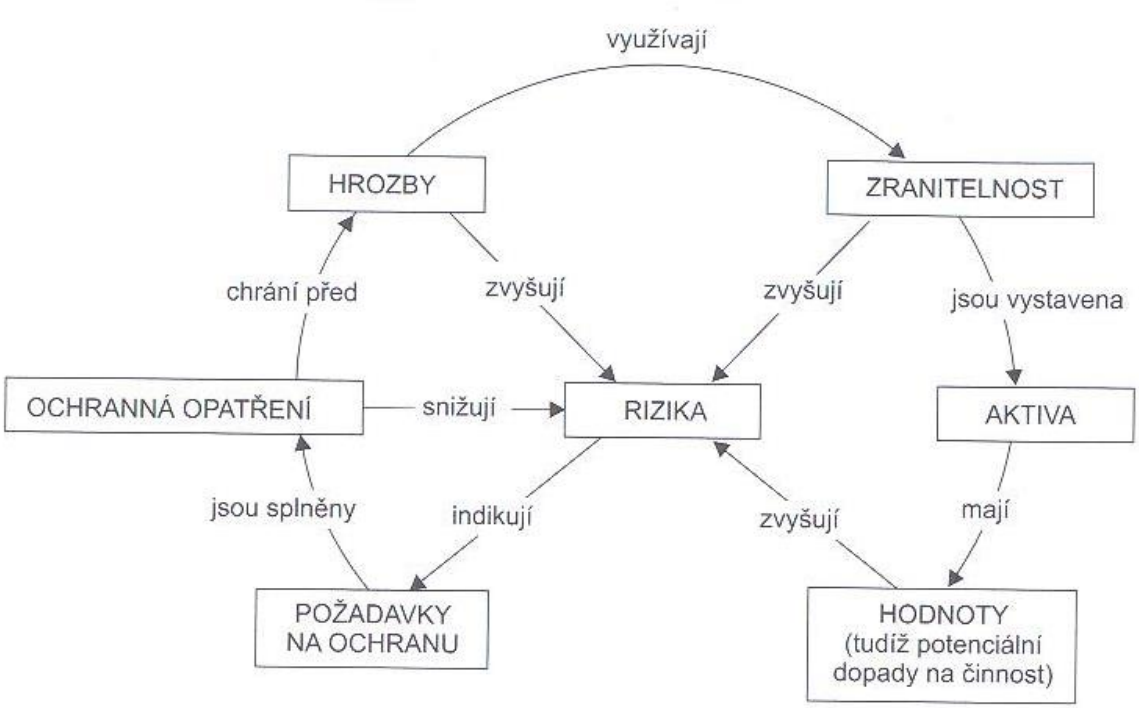
Obrázek 2.15: Vztahy mezi prvky v analýze rizik

Zdroj: [12]



Obrázek 2.16: Vztahy mezi prvky při analýze rizik

Zdroj: [12]



Obrázek 2.17: Vztahy při řízení rizik

Zdroj:[12]

Analýza rizik je sestavena z několika po sobě jdoucích obecných činností, mezi které v první řadě platí vytvoření pomyslné hranice analýzy rizik. Jedná se o fiktivní čáru, která odděluje od sebe jednotlivá aktiva. *„Aktiva, která mají vzhledem k probíhajícímu procesu snižování rizik vztah k cílům managementu, budou zahrnuta do analýzy a budou ležet uvnitř hranice analýzy. Ostatní aktiva budou ležet mimo hranici analýzy rizik.“* [12, str. 102]

Po vytvoření pomyslné hranice následuje identifikace aktiv, při níž dochází k soupisu všech aktiv, která v průběhu stanovování hranice byla zařazena dovnitř hranice pro analýzu rizik.

Následujícím krokem je stanovení hodnoty a seskupování aktiv. Posouzení hodnoty aktiv vyplývá z velikosti škody, která byla způsobena poškozením nebo ztrátou konkrétního aktiva. Vyčíslení hodnoty aktiv obvykle vyplývá z nákladových, někdy i z výnosových, charakteristik. Pod pojmem nákladové charakteristiky si lze představit pořizovací nebo reprodukční pořizovací ceny. Výnosové charakteristiky lze definovat jako zisky, případně významné přínosy pro subjekt, jednoho konkrétního aktiva.

Jedním z nejdůležitějších kroků analýzy rizik je identifikace hrozeb, která spočívá ve výběru těch, které by mohly potenciálně ohrozit alespoň jedno aktivum subjektu. Tato identifikace vyplývá ze seznamu hrozeb, který se sestavuje na základě zkušeností, poznatků zjištěných z literatury a provedených průběžných analýz. U aktiv, kterými identifikujeme možnost výskytu hrozby, následuje určení úrovně hrozby a zranitelnosti. Úroveň hrozby je založena na faktorech, kterými jsou například motivace, nebezpečnost a přístup. Naopak citlivost a kritičnost se zohledňuje při stanovení úrovně zranitelnosti.

„Při analýze rizik musíme posoudit pravděpodobnost naplnění každého scénáře, tj. s jakou pravděpodobností se naplní určitá hrozba a jak využije zranitelnosti. Např. podle ČSN ISO/IEC 27005 při určování pravděpodobnosti je nutné brát v úvahu:

- *zkušenosti a platné statistiky o pravděpodobnosti hrozeb*
- *u zdrojů úmyslných hrozeb: motivaci a schopnosti, které se časem mění, a zdroje přístupné případným útočníkům, jakož i vnímání atraktivity a zranitelnosti aktiv pro případného útočníka*
- *u zdrojů náhodných hrozeb: geografické faktory, například těsná blízkost chemických nebo naftových zdrojů, možnost extrémních atmosférických podmínek a faktory, které by mohly mít vliv na lidská selhání a funkční poruchy zařízení*

- *zranitelnosti, jak jednotlivě, tak v souvislostech*
- *existující opatření a jejich účinnost na snížení zranitelnosti.“* [12, str. 105]

2.2.3 Metody analýzy rizik [9][11][12][14]

Metody analýzy rizik lze rozdělit na základě způsobu vyjádření veličin, s kterými se při analýze rizik pracuje. Metody analýzy lze rozdělit do dvou skupin na základě přístupu k jejímu řešení:

- kvantitativní
- kvalitativní.

Při analýze rizik se používá buď jeden s těchto přístupů nebo kombinace obou.

Kvantitativní metody analýzy rizik jsou založeny na matematických výpočtech rizika. Tento výpočet se provádí na základě znalosti frekvence výskytu hrozby a možném dopadu. Riziko se v tomto případě nejčastěji vyjadřuje formou roční předpokládané ztráty. Roční předpokládaná ztráta neboli „Annualized loss epectancy“ je vyjádřena finanční částkou. Kvantitativní metody bývají časově náročnější než metody kvalitativní, avšak poskytují finanční vyjádření rizika, což je nápomocné při jejich zvládání. Tento typ metod má i své nevýhody, mezi které lze zařadit:

- náročnost na provedení
- vysoce formální postup na provedení těchto metod
- potenciálně vysoká zranitelnost.

Kvalitativní metody vyplývají z popisu závažnosti dopadu a z pravděpodobnosti, která vyjadřuje stav, že nežádoucí stav nastane. V případě této metody jsou rizika vyznačována v konkrétním rozsahu, případně intervalu. Někdy bývají vyjádřena i slovně, např. malá, střední a velká. Mezi hlavní výhody kvalitativních metod patří rychlost vypracování. Naopak za hlavní a velmi významnou nevýhodu této metody můžeme považovat náchylnost vůči subjektivnímu postoji k riziku. Tato náchylnost se vyskytuje převážně ve fázi, kdy dochází k posouzení přijatelnosti výše finančních nákladů, které mají být vynaloženy na preventivní opatření na snížení hrozby. Kvalitativní metoda je využívána při nedostačující kvalitě případně kvantitě získaných údajů, které jsou číselně vyjádřeny, pro využití v kvantitativních metodách analýzy rizik.

Hlavním pilířem kombinovaných metod jsou číselné údaje, ze kterých tyto metody vyplývají. Výsledné rozhodnutí se blíží realitě díky kvalitativnímu hodnocení na rozdíl od výsledků získaných kvantitativní metodou.

Při volbě nejvýhodnější metodiky analýzy rizik lze vybírat ze čtyř možností přístupu, mezi které patří základní nebo neformální přístup, podrobná analýza rizik a kombinovaný přístup.

Analýza rizik se sestává z orientační analýzy a detailní analýzy rizika. Orientační analýza umožňuje následnou volbu vhodné metodiky pro analýzu samotnou. V první fázi této analýzy dochází k posouzení a identifikaci klíčového objektu a jeho náchylnost vůči rizikům. Klíčovým objektem může být problém, aktivum, systém a jiné. Pokud je identifikován objekt s vysokou náchylností k rizikům, je provedená následná detailní analýza rizik, při které se využívá kvalitativních nebo kvantitativní metod, případně jejich kombinace. Kombinace obou metod je nejvýhodnější volbou mezi metodami analýzy rizik, avšak jedná se o velmi nákladnou a zdlouhavou metodu.

Při výběru konkrétní metody je důležité zohlednit reálný stav analyzovaného prostředí a výhody, případně nevýhody, uvažované metody. Rozhodnutí, který přístup vybrat, závisí na tom, jakého cíle chceme dosáhnout při užití analýzy rizik, pro jaké účely daný objekt používáme, zda objekt vykazuje kritické funkce a zhodnocení výnosnosti objektu a návratnosti investic.

Mezi základní metody pro stanovení rizik například patří:

1. Fault Tree Analysis
2. Event Tree Analysis
3. Failure Mode and Effect Analysis
4. Quality Function Deployment
5. Check List
6. Safety Audit
7. What – If Analysis
8. Preliminary Hazard Analysis.

2.2.4 Fault Tree Analysis [14][17]

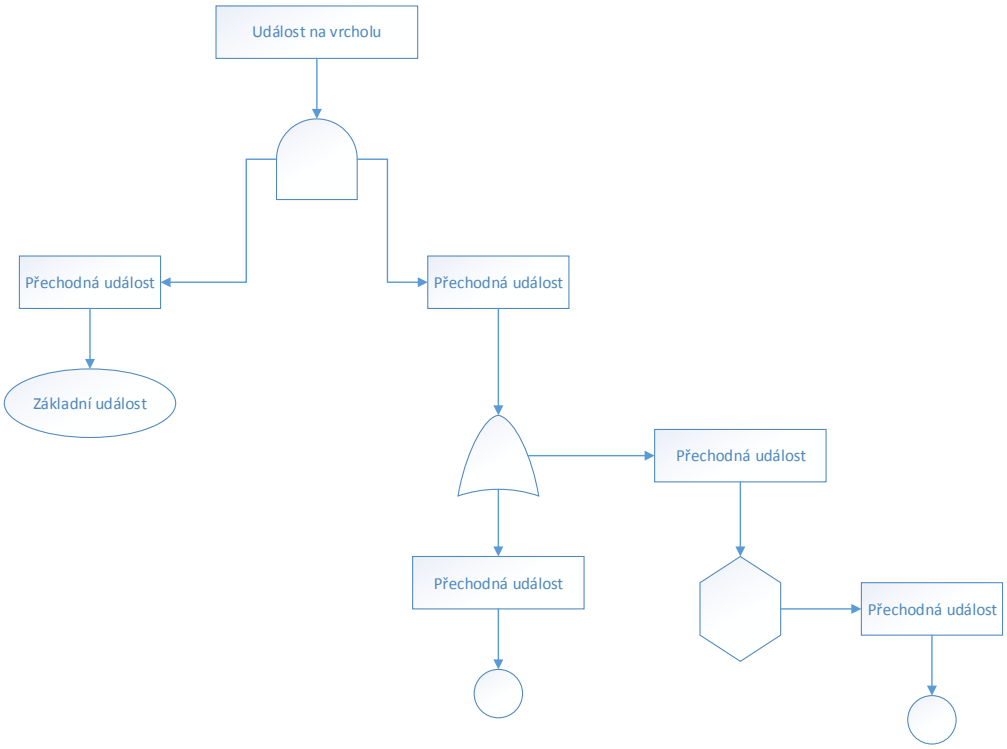
Fault Tree Analysis neboli analýza stromu poruch je založena na rozboru jednotlivých událostí, které mohly potenciálně vést k vybrané nežádoucí události. FTA vychází z faktů týkajících se jednotlivých procesů, případně jevů odhalujících navzájem provázané řetězce událostí, na jejichž konci dochází k odhalení příčiny, proč nežádoucí jev nastal.

Metoda FTA bývá vyobrazována pomocí rozvětveného grafu doplněného o hodnotovou symboliku a podrobnější popis. Na základě tohoto vyobrazení lze mluvit o takzvané graficko – statistické metodě. Analýza poruch stromu má za hlavní cíl posouzení pravděpodobnosti vrcholové události za pomoci statistických, případně analytických metod.

Aby byla tato metoda plně využita, měly by být splněny tři důležité milníky, kdy prvním a nejspíše nejdůležitějším je porozumění fungování systému, případně podniku. Dalším, tedy druhý bodem, je důkladné porozumění a schopnost orientace v nákresu a postupu. Třetím, nikoliv zanedbatelným bodem, je znalost možného způsobu selhání jednotlivých komponent a jejich následných dopadů.

Největší předností analýzy stromu je schopnost identifikovat a klasifikovat kombinace, které jsou tvořeny kombinací základních poruch. Tyto poruchy spočívají ve slabinách zařízení a lidských chyb, které mohou potenciálně vést k nežádoucímu jevu, nehodě, případně havárii. Kvalitně provedená FTA umožní analytikovi zaměřit se na eliminaci základních příčin, které mohou způsobit nehodu.

„Strom poruch je tedy konstruován tak, aby popsal sled událostí, které samostatně nebo v kombinaci s jinými událostmi mohou vést k vrcholové události. Takovým příkladem může být například posouzení pravděpodobnosti automobilové nehody na křižovatce, kdy strom poruch je tvořen událostmi a tzv. hradly (angl. gates) A a NEBO (angl. AND a OR). Strom je tedy tvořen dedukcí podmínek vzniku vrcholové události a posloupností jednotlivých úrovní až do definování nejnižší úrovně příčin.“ [17] Přiřazením pravděpodobnosti výskytu ke každé definované příčině dochází k výpočtu pravděpodobnosti výskytu nejkritičtějšího problému. Tento krok vyžaduje schopnost dedukce, s jakou pravděpodobností se daný jev vyskytuje. Aby bylo možné vypočítat konečnou pravděpodobnost, je nutné vyčíslit pravděpodobnost u všech definovaných jevů, přičemž u hradla A vyplývá z výpočtu součinu všech pravděpodobností u hradel NEBO.



Obrázek 2.18: Struktura analýzy stromu chyb

Zdroj: vlastní tvorba

2.2.5 Event Tree Analysis [14]

Event Tree Analysis, někdy také označnován jako analýza stromu událostí, popisuje způsob, jak sledovat průběh konkrétního procesu od iniciační události na základě příznivé nebo nepříznivé možnosti. Stejně jako je tomu u analýzy stromu poruch, ETA představuje graficko-statistickou metodu. Systémový strom je vyobrazen pomocí rozvětveného grafu, kterému náleží konkrétní popisy a předem definovaná symbolika.

Aby byla metoda stromu událostí plně využita, je nezbytně nutné znát veškeré možné inicializační události, funkci bezpečnostních systémů, případně nouzových procedur, které zmírňují dopady nežádoucích vlivů.

Pomocí metod analýzy stromů dochází k identifikaci slabých míst v procesech. Ne vždy se procesy vyvíjejí směrem, který je předpokládaný, případně žádaný, proto je zde na místě obezřetnost a schopnost podnikat bezpečnostní opatření.

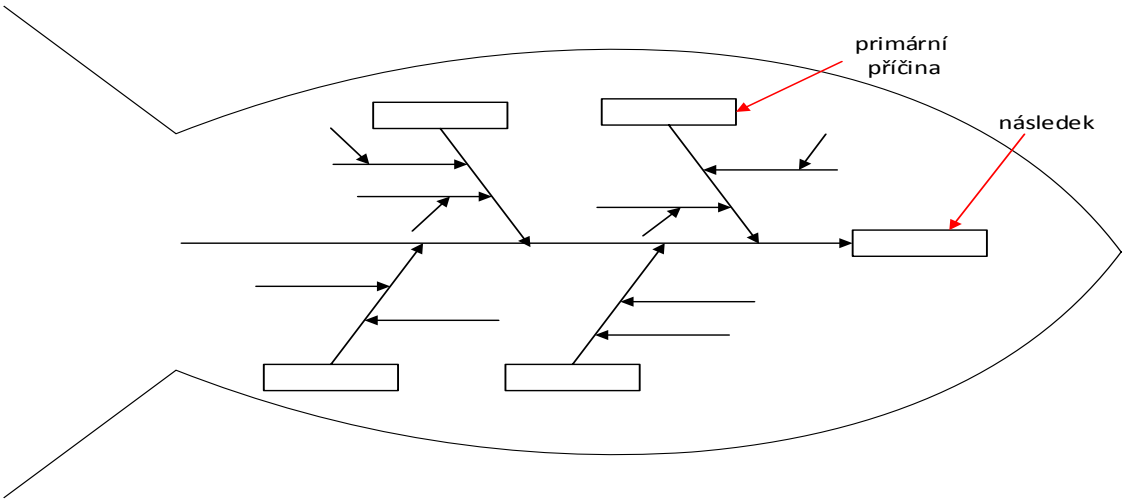
2.2.6 Failure Mode and Effect Analysis [12][14]

Failure Mode and Effect Analysis, zkráceně FMEA, je analýza možností vzniků vad, případně následků, které tyto vady vyvolají. Tato preventivní metoda zajišťující kvalitu je určena k vyhledávání možných chyb a vad a sestavení následných preventivních opatření. V příloze č. 2 je vyobrazen vzorový formulář pro provedení této analýzy.

Jako hlavní výhodu této metody můžeme definovat schopnost optimalizace návrhu, odhalení možných chyb, vad. Díky této schopnosti lze říci, že vede k zlepšení plánu jakostí produktů a zvýšení celkové spokojenosti zákazníků. FMEA nepatří mezi finančně nákladné metody, proto lze i mezi výhody zahrnout úsporu nákladů v případě, že by došlo k výskytu chyby.

2.2.7 Ishikawův diagram [11][12][14]

Ishikawův diagram je díky své jednoduchosti velmi často používanou metodou pro zpracování analýzy rizik, kdy dochází k definici problému, který může nastat tzv. následkem. V další fázi jsou hledány příčiny, které způsobily již definovaný problém. Ishikawův diagram získal jméno po svém stvořiteli Kaoru Ishikawovi, který jej poprvé použil v roce 1943. Pro jeho charakteristický tvar ryby, jak je patrné z obrázku č. 2.19, bývá velmi často nazýván diagramem rybí kosti. Hlava této ryby je tvořena definicí nežádoucího výsledku, tedy následkem. Jednotlivé „kosti“, které tvoří rybí kostru zachycující příčiny, které způsobily nežádoucí výsledek. Tyto příčiny se mohou dále větvit do sekundárních, někdy i terciálních částí diagramu.



Obrázek 2.19: Grafické znázornění Ishikawova diagramu

Zdroj: vlastní tvorba

3 ANALÝZA VYBRANÉHO ETL PROCESU

V následujících podkapitolách je definována mnou vybraná část ETL procesů, konkrétně extrakce. Tuto část procesu vyobrazím pomocí vývojového diagramu, který následně slovně okomentuji. Druhým krokem této definice procesu je extrakce pomocí jazyka SQL na základě vývojového diagramu. Ve třetí části této kapitoly podrobím extrakci metodám analýzy rizik, od kterých očekávám vyobrazení potenciálních rizik. Jednotlivé analýzy rizik provedu nezávisle a v časovém odstupu, abych odstranila vzájemnou ovlivnitelnost mezi jednotlivými metodami.

Veškeré poznatky zachycené v následujících podkapitolách vycházejí z poznatků získaných z praxe v oboru.

3.1 DEFINICE EXTRAKCE POMOCÍ VÝVOJOVÉHO DIAGRAMU

Jak je již patrné z teoretické části mé práce, proces extrakce je tvořen několika fázemi, které jsou představovány získáním souboru dat, která byla vyexportována poskytovatelem na předem definované úložiště a kontrolou struktury těchto dat. Jelikož se jedná o celkem rozsáhlý proces, rozdělila jsem extrakci do dvou částí.

Diagram číslo 3.1 znázorňuje jednoduchý průběh extrakce souboru dat ze zdojového systému. Jak je z vývojového diagramu patrné, v první fázi extrakce dochází k ověření dostupnosti serveru. Jinými slovy je prováděna kontrola, zda je server aktivní či nikoliv. Pokud není server aktivní, celý proces extrakce končí chybou. V případě, že je server dostupný, následuje kontrola přístupových oprávnění.

Zjednodušeně lze říci, že probíhá kontrola toho, zda má uživatel právo přistupovat na server a číst z něj data. Pokud toto oprávnění není uživateli přiděleno, proces skončí chybou. V opačném případě pokračuje kontrolou dostupnosti souboru.

Tato kontrola spočívá v ověření, že je požadovaný soubor nahrán do předem definovaného úložiště. Jestliže soubor není na předdefinovaném serveru, proces končí chybou. V opačném případě proces pokračuje ověřením formátu.

Každý soubor musí být vyexportován na definovaný server ve formátu, který byl předem domluven. Pokud tato dohoda není splněna, nastává konec extrakce a celý proces je ukončen chybovou hláškou. V opačném případě následuje ověření, zda je dostupné cílové úložiště, tedy cílový server.

Ve chvíli, kdy je cílové úložiště dostupné, následuje proces ověřování přístupových práv, avšak v opačném případě dochází k ukončení procesu extrakce. Tento proces opět končí chybou.

Přístupová práva jsou v běžné praxi přidělována administrátorem konkrétního serveru, který je pověřen přidělovat tato práva vybraným lidem a současně se starat o chod konkrétního serveru. Pokud v průběhu procesu ověřování přístupových oprávnění dojde k zjištění, že uživatel tato práva nemá, celý proces extrakce opět končí neúspěchem. V opačném případě je extrahovaný soubor uložen na cílové úložiště a proces je úspěšně ukončen.

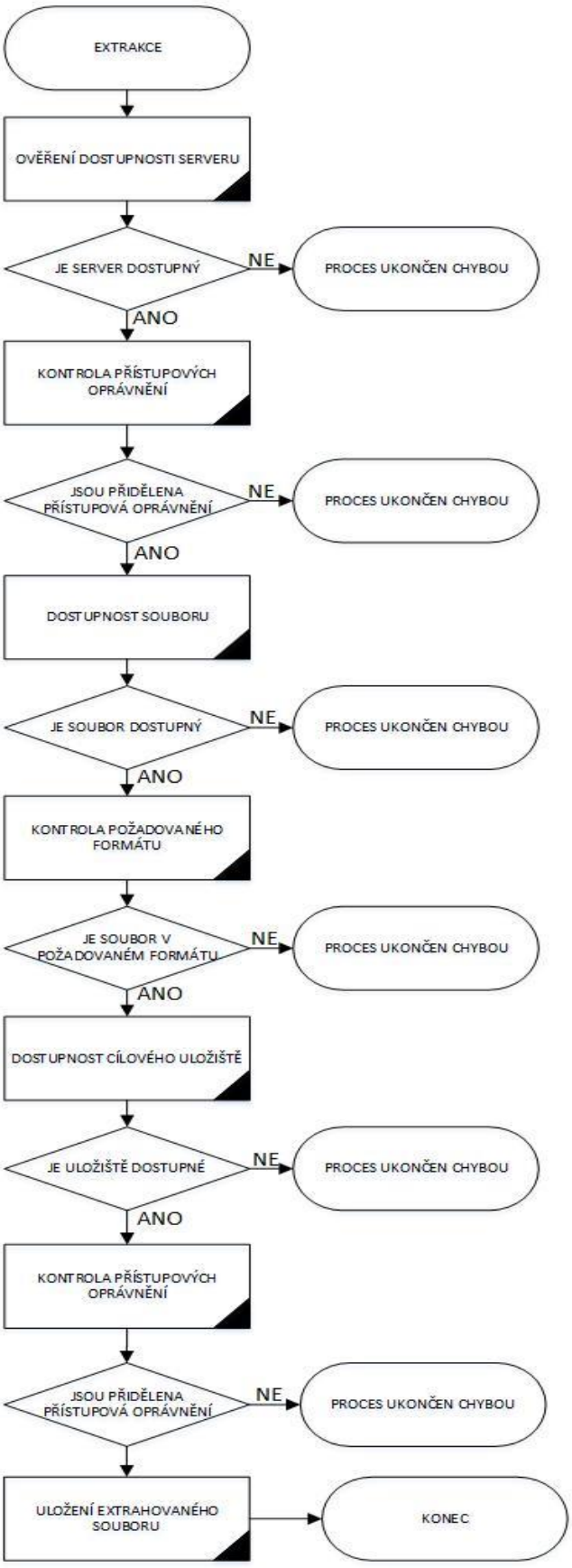


Diagram 3. 1: Vývojový diagram extrakce (získání zdrojových dat) před analýzou rizik

Zdroj: vlastní tvorba

Diagram číslo 3.2 zachycuje proces extrakce, kdy dochází k nahrání dat do zdrojové databáze. V rámci této fáze extrakce dochází k ověření struktury dat. Níže vyobrazený diagram obsahu jednoduchý průchod procesu.

Prvním krokem této fáze je ověření, zda je dostupné úložiště, na které jsme soubor s daty uložili v předcházející fázi extrakce. Ve chvíli, kdy ověření dostupnosti je úspěšné, proces pokračuje v ověřování dostupnosti databáze. V opačném případě tento proces končí chybou. Jelikož je proces extrakce rozdělen do dvou částí, některá ověření se ve vývojových diagramech opakují, jako tomu je právě v tomto případě.

Následující částí této fáze extrakce je ověření dostupnosti databáze, která bývá často nazývána jako „source“ databáze. Pokud je databáze dostupná a současně máme přístupová oprávnění pro zápis dat do jednotlivých tabulek, proces pokračuje dál. Jinak proces opět kočí chybou.

Při exportu dat je velmi důležité dodržet předem domluvenou strukturu dat. Přesněji řečeno je nezbytné klást pozornost na formát souboru, oddělovače jednotlivých hodnot a další dílčí vlastnosti. Tyto vlastnosti musí být nastaveny poskytovatelem před přípravou exportu dat a současně tato definice musí být součástí procesu extrakce, abychom byli schopni provést ověření, zda data mají správnou strukturu. Pokud je definice těchto vlastností chybná, proces se nezdaří a končí chybou. V opačném případě dochází k nahrání dat do zdrojové, tedy „source“ databáze.

V rámci nahrávání dat je důležité definovat cílové úložiště, tedy název databáze a tabulky, kam mají být jednotlivá data uložena. Pokud je tato definice dodržena, mělo by dojít k úspěšnému nahrání dat. V opačném případě celý proces končí chybou. Pokud jsou data úspěšně nahrána, proces je úspěšně ukočen.

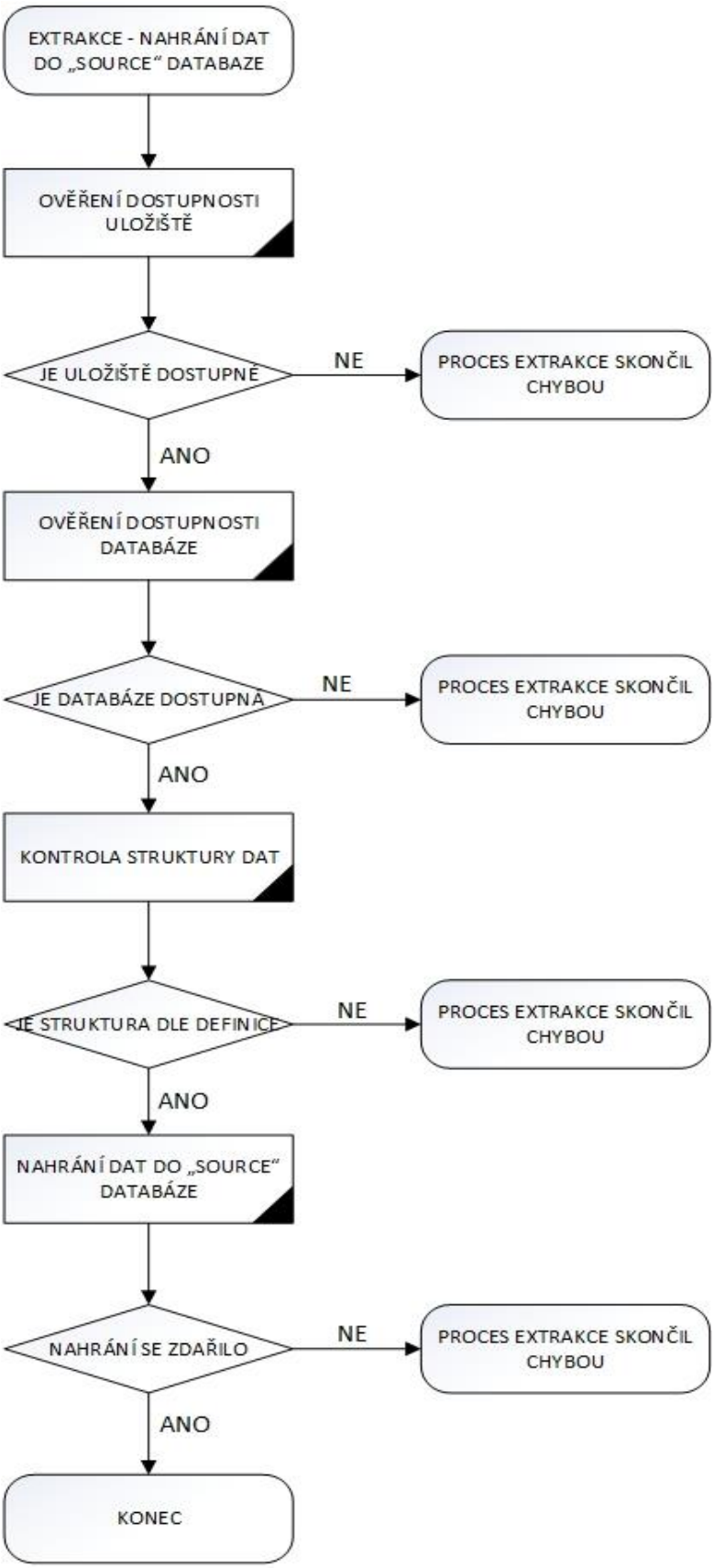


Diagram 3. 2: Vývojový diagram extrakce (nahrání dat do „source“ databáze) před analýzou rizik

Zdroj: vlastní tvorba

3.2 PROCES EXTRAKCE VYJÁDŘENÝ POMOCÍ SQL

V následujících podkapitolách jsou rozebrány jednotlivé části procesu extrakce, který jsem vyjádřila pomocí SQL kódu. Pro spoštění procedur je důležité dodržet pořadí jejich procesingu. Jako první je spuštěna procedura pro získání dat ze zdrojového úložiště. Následně je provedeno nahrání zdrojových dat do „source“ databáze pomocí mnou definované procedury. Souvislý kód je součástí příloh číslo 6 a 7.

3.2.1 Extrakce – získání souboru dat ze zdrojového úložiště

Přenesení souboru ze zdrojového na cílové úložiště lze provést několika způsoby, avšak ne všachna možná řešení jsou bezpečná. Jako hlavního kandidáta, který může vyvolat bezpečnostní incident si lze představit právě přenos souboru pomocí jazyka SQL. Pro kopírování souborů existuje v tomto jazyce předdefinovaná procedura xp_cmdshell. Použitím tohoto příkazu dojde k otevření systémového přístupu, kterého může využít potenciální útočník a způsob tím bezpečnostní incident nebo-li škodu. Proto lze říci, že použitím právě tohoto příkazu vytvoříme značnou hrozbu.

Za bezpečný a mnohem jednodušší způsob získání souboru dat ze zdrojového úložiště lze označit přenos pomocí command utility. Příkaz pro přenos souboru stačí napsat do příkazového řádku a spustit pomocí klávesy ENTER. Následující obrázek zachycuje prakticou ukázku použití cmd utility. V příloze číslo 6 je definován obecný kód pro vytvoření cmd souboru, při jehož spuštění je vyplněno zdrojové a cílové úložiště.

3.2.2 Extrakce – nahrání zdrojových dat do source databáze

Následující část zdrojového kódu obsahuje definici databáze, která má být použita pro vytvoření procedury tedy, kde má být zdrojový kód procedury uložen.

Mnou definovaná procedura obsahuje vstupní proměnné, které nejsou naplněny přímo. Jedná se o proměnné @fordate, která představuje den, za který jsou zpracovávána data. Další proměnná je @filename představující název souboru a proměnná @fileformat zastupující formát souboru obsahujícího zdrojová data. Velmi důležitou proměnnou je @DataFolder pomocí níž je definováno úložiště souboru dat. Proměnná @ProjectName představuje název projektu v rámci něhož jsou nahrávána zdrojová data.

```
USE [SRC]
GO

CREATE PROC [dbo].[LOAD_SRC_DATA] @fordate datetime = null, @filename sysname,
                                  @fileformat nvarchar(10), @DataFolder varchar(1024),
                                  @debug bit=1, @ProjectName varchar(255)
AS
```

V následující části kódu dochází k deklaraci proměnných, které mají za úkol usnadnit práci s kódem a slouží i pro jeho lepší čitelnost a přehlednost.

```
DECLARE @bulk_insert varchar (max),
             @extract_fileX  varchar(max),
             @err_message nvarchar(512) = NULL,
             @sqlstm VARCHAR(MAX) = '',
             @extract_date VARCHAR(10)
```

Pokud není název projektu naplněn pomocí vstupní proměnné dochází k vyplnění názvu uvirezální hodnotou.

```
       -- ProjectName
IF @ProjectName IS NULL
SET @ProjectName = 'DefaultProject'
```

Jestliže není vstupní proměnná @DataFolder naplněně na vstupu, dochází k přiřazení hodnoty z tabulky PROJECT, ve které jsou shromažďovány veškeré informace o existujících projektech.

```
-- Get data folder
IF @DataFolder IS NULL
       SET @DataFolder = (SELECT [DATAFOLDER] FROM dbo.PROJECT WHERE PROJECT_NAME =
       @ProjectName)
```

V následující části kódu dochází k přiřazení konkrétního datumu a času, za který jsou daná data zpracovávána. Pomocí funkce CONVERT dochází k úpravě datumu a času do požadovaného tvaru.

```
-- Fordate
IF @fordate IS NULL
       SET @fordate = (SELECT CONVERT(datetime, GETUTCDATE()))

--
```

Proměnná @extract_date je naplněna vstupní proměnnou @fordate, která je převedena do požadovaného formátu. Následně je pomocí proměnné @extract_fileX definován název a cesta k souboru. Tato definice se skládá ze tří proměnných @DataFolder, @filename a @fileformat.

```
--
SET @extract_date =
CONVERT(varchar,@fordate,112)+SUBSTRING(CONVERT(varchar,@fordate,114),1,2)
SET @extract_fileX = @DataFolder + @filename + '.' + @fileformat
--
```

Proměnná @bulk_insert je naplněna definicí BULK INSERTu jehož součástí je mimo jiné definice oddělovače řádků a jednotlivých sloupců. Následně dochází k naplnění proměnné @sqlstm příkazem TRUNCATE TABLE, který má za úkol vyprázdnit cílovou tabulku v SRC databázi.

```
--Bulk insert
SET @bulk_insert = 'BULK INSERT [dbo].[SRC_DATA_'+@filename+']
                              FROM '''+@extract_fileX+'''
                              WITH (DATAFILETYPE  = ''char'',
                              KEEPNULLS,
                        TABLOCK,
                        FIRSTROW = 1,
                              ROWS_PER_BATCH =100000,
                              FIELDTERMINATOR = '','',
                              ROWTERMINATOR =''0x0A'',
                              MAXERRORS = 1
                              )'

--
--
SET @sqlstm = 'TRUNCATE TABLE [dbo].[SRC_DATA_'+@filename+']'
--
```

V poslední sekci procesu extrakce dochází k vyprázdnění cílové tabulky a spuštění BULK INSERTu, který má za úkol naplnit cílovou tabulku čerstvými daty.

```
-- Run/Print statement
IF @debug = 0
       BEGIN
              -- truncate table
              EXEC(@sqlstm)
              -- execute bulk insert command
              BEGIN TRY
                     EXEC(@bulk_insert)
                     IF @@ROWCOUNT = 0
                     BEGIN
                     RAISERROR('Trying to load 0 rows. Check extract file.',16,1)
                     END
              END TRY
                     BEGIN CATCH
                            SET @err_message = (SELECT ERROR_MESSAGE())
                            RAISERROR(@err_message, 16, 1)
                     END CATCH
       END
```

3.3 ANALÝZA RIZIK EXTRAKCE POMOCÍ METODY FTA

Proces extrakce, blíže rozebraný v kapitole 3.2, podrobím analýze rizik pomocí metody FTA. Při tvorbě této analýzy vycházím ze situace, kdy se proces extrakce nezdařil, skončil chybou.

Tento stav může být způsoben čtyřmi faktory. Konkrétně selháním lidského faktoru, případně se vyskytne infrastrukturní chyba, proces selže na administrátorské chybě nebo nastane neopředvídatelná událost. Z běžné praxe vím, že velmi častou příčinou bývá kombinace selhání několika faktorů.

Selhání lidského faktoru může být způsobeno pochybením na straně poskytovatele dat nebo na straně příjemce dat, tedy developera. Při definici procesu extrakce je důležité správné definování zdrojového a cílového úložiště. Cesty k těmto zdrojům dat musí být specifikovány bezchybně. Další důležitou částí definice procesu extrakce je přesná specifikace souboru dat a jeho formát. Pokud dojde k pochybení některé z těchto definic, chyba se projeví v průběhu snahy o extrakci dat a proces skončí chybou. Pravděpodobnost této poruchy jsem vyčíslila na 55%, což se v závěru ukázalo jako nejrizikovější faktor, který ovlivňuje tento proces.

Infrastrukturní chybou můžeme označit výpadek, tedy nefunkčnost, zdrojového nebo cílového serveru, případně chybu „source“ databáze. Nefunkčnost úložiště, ať už cílového nebo zdrojového, může být způsobena softwarovou nebo hardwarovou chybou serveru. V každém případě lze tuto chybu kategorizovat jako závadu na infrastrukturním zařízení. Pravděpodobnost, že proces extrakce skončí chybou, právě v důsledku této závady, je necelých 20%.

Pro přístup na zdrojové, případně cílové, úložiště je nezbytné mít přidělena přístupová oprávnění na předem definovaná úložiště. V případě, že tato oprávnění nejsou přidělena, můžeme hovořit o administrátorském pochybení. Tato oprávnění jsou nezbytná i pro přístup do „source“ databáze. Pravděpodobnost ukončení procesu, které vzniklo právě touto chybou, je velmi malé, a to okolo 1%.

Každá činnost může být překažena nenadálou událostí, proto je důležité vzít tento faktor v úvahu i při analýze rizik extrakce dat. V tomto případě lze označit jako nenadálou mohou vzniknou díky období dlouhých a vytrvalých dešťů, případně nečekané výpadky elektřiny v návaznosti na silné bouřky. Jako neočekávanou závadu lze označit poruchy událost udeření vyšší moci, případně neočekávané závady. Pod úderem vyšší moci si představme například požár vzniklý v důsledku období dlouhých veder a sucha, záplavy, které

mohou vzniknou díky období dlouhých a vytrvalých dešťů, případně nečekané výpadky elektřiny v návaznosti na silné bouřky. Jako neočekávanou závadu lze označit poruchy vodoinstalace nebo elektroinstalace, které vznikají v souladu se stářím, opotřebením. Závady vzniklé právě tímto rizikem vznikají s pravděpodobností 5%, ale i tak je velmi důležité se proti tomuto riziku patřičným způsobem bránit nebo pojistit.

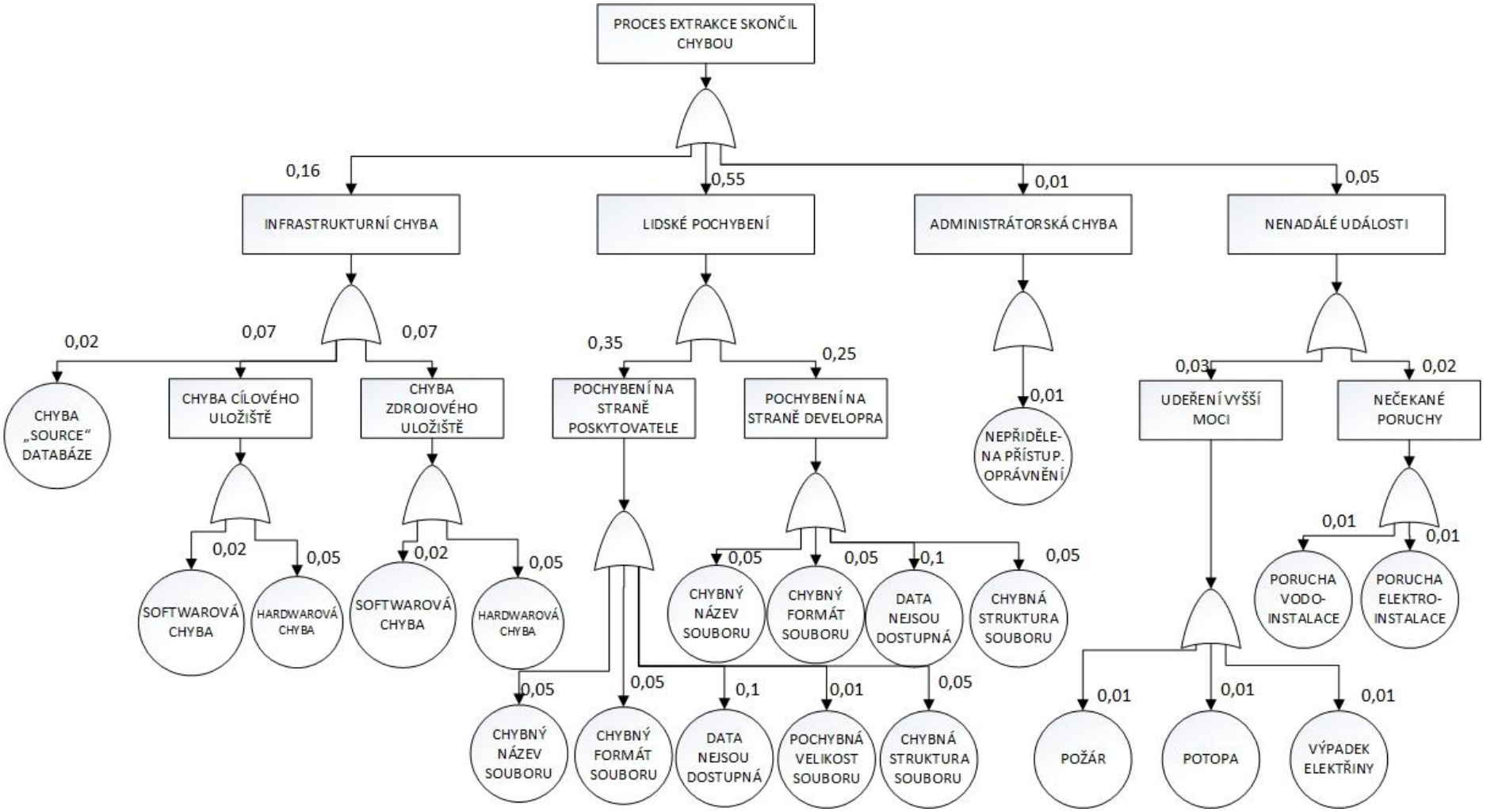


Diagram 3. 3: FTA analýza - porucha extrakce souboru dat

Zdroj:vlastní tvorba

3.4 ANALÝZA PŘÍČIN A NÁSLEDKŮ SELHÁNÍ PROCESU EXTRAKCE VYJÁDŘENÁ POMOCÍ DIAGRAMU RYBÍ KOSTI

Jak je patrné u diagramu č. 3.3, existují čtyři hlavní příčiny selhání procesu extrakce a to:

a) selhání lidského faktoru

b) vznik infrastrukturní chyby

c) administrátoský nedostatek a

d) vznik nenadálé poruchy.

Selhání lidského faktoru můžeme označit za následek pro pochybení na straně poskytovatele dat nebo příjemce. Tito lidé se mohou dopustit pochybení při definici zdrojového, případně cílového, úložiště. Důležité je také klást důraz a pozornost na definici názvu a formátu extaktovaného souboru.

Nefunkčnost serverů v důsledku softwarové, případně hardwarové závady, lze označit jako příčinu pro pochybení na straně infrastruktury, stejně tak i nedostupnost „source“ databáze. S pojmem hardwarová chyba bývá nejčastěji spojován nedostatek místa na zdrojovém nebo cílovém úložišti. Bohužel z mé praxe vím, že se jedná o mnohdy nepředvídatelný problém, proti kterému je často z finančních důvodů obtížné se bránit, a proto dochází k promazávání archivovaných dat.

Při každém procesu extrakce je nezbytné, aby účet, pod kterým je spuštěna procedura pro proces extrakce, měl přidělen přístupová oprávnění na zdrojové, cílové úložiště a samozřejmě do „source“ databáze. V případě, že se jedná o úložiště, kam je soubor dat vyexportován od poskytovatele, stačí, aby účet měl přidělena práva na čtení z tohoto úložiště. Avšak pokud se jedná o úložiště, kam má být soubor uložen a archivován, nebo přístup do „source“ databáze, je zapotřebí přidělit přístupová oprávnění pro zápis.

Vznik nenadálé poruchy je následkem udeření vyšší moci, případně nepředvídatelné závady. Požár, potopu, případně výpadek elektřiny, lze určit jako faktory, které mají za následek vznik nečekané závady v souladu s udeřením vyšší moci. Naopak závadu vodoinstalace nebo elektroinstalace lze přiřadit mezi příčiny vyvolávající vznik neočekávané závady v návaznosti na vznik neočekávaných poruch.

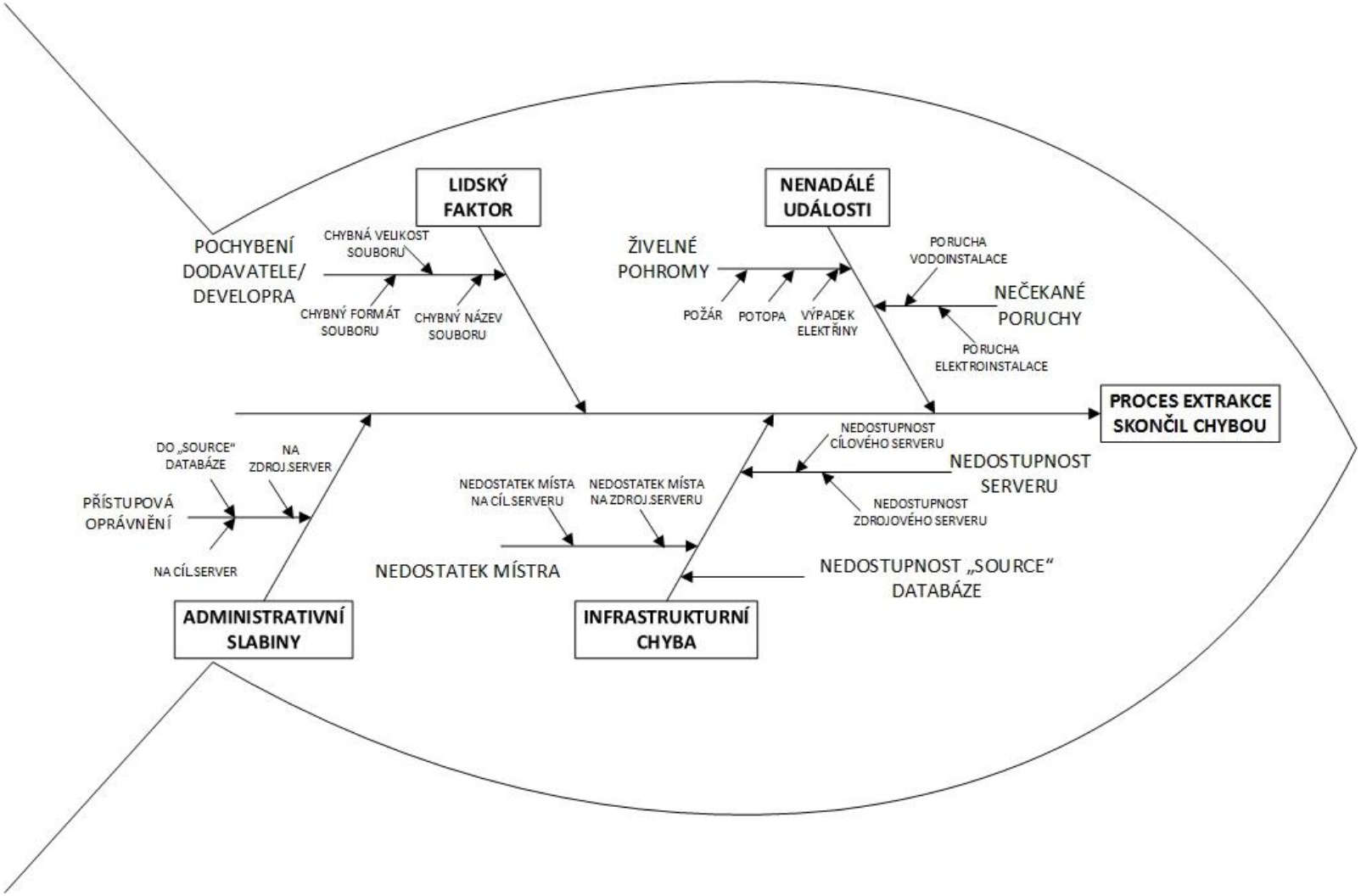


Diagram 3. 4: RYBÍ KOST - proces extrakce

Zdroj: vlastní tvory

4 NÁVRH PREVENTIVNÍCH OPATŘENÍ V PRŮBĚHU PROCESU EXTRAKCE A JEJICH OVĚŘENÍ

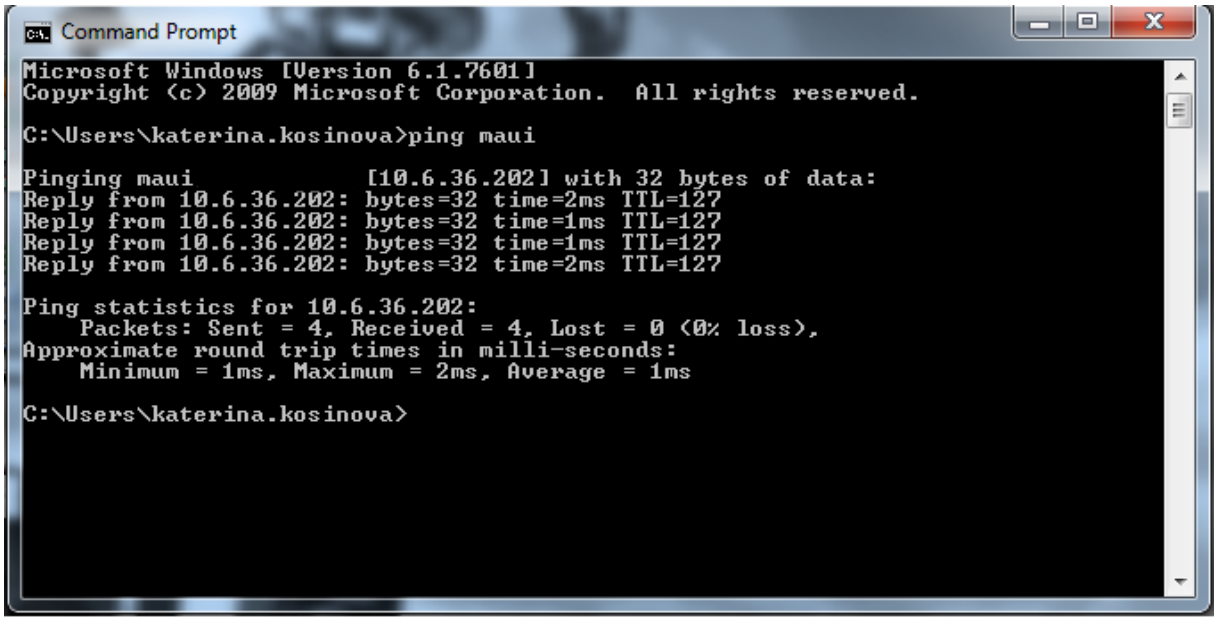
Následující část diplomové práce věnuji definování preventivních opatření, která doporučuji provést, aby došlo ke snížení potenciálních rizik spojených s procesem extrakce.

Mnou navrhovaná opatření vyplývají z poznatků získaných mou vlastní praxí v oboru a souběžné konzultace problému s odborníky, kteří se pohybují taktéž v oboru Business Intelligence a infrastruktury.

Díky zkušenostem, které jsem měla možnost nabýt během mé praxe v oboru Business Intelligence mohu říci, že mnou navrhovaná preventivní opatření lze označit za funkční a plně dostačující.

4.1 DEFINICE PREVENTIVNÍCH OPATŘENÍ POMOCÍ VÝVOJOVÉHO DIAGRAMU

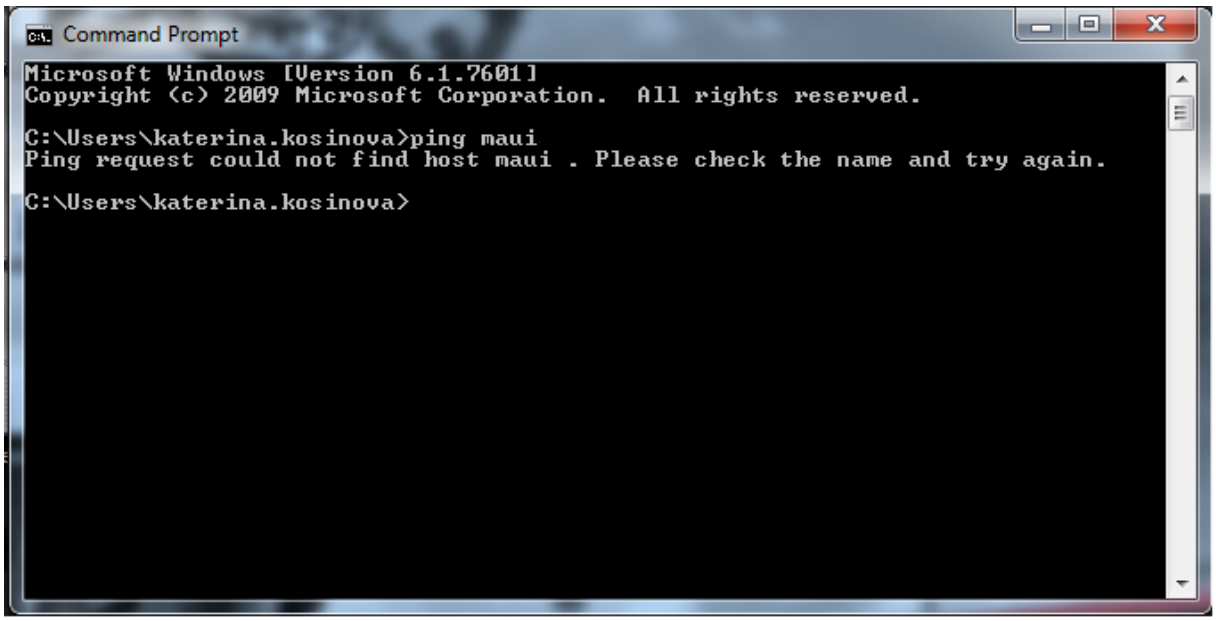
Vývojový diagram, vyobrazený v příloze č. 3, zachycuje proces extrakce – získání zdrojových dat. První fází tohoto procesu je ověření, zda je dostupný server, na který byl soubor dat vyexportován poskytovatelem. Pokud server není dostupný, je nutné provést ověření funkčnosti úložiště. Toto ověření lze provést pomocí dotzu ping v příkazovém řádku. Pokud je úložiště funkční, začne odpovídat, jak je vyobrazeno na obrázku č. 4.20.



Obrázek 4.20: Úspěšné ověření funkčnosti uložiště

Zdroj: vlastní tvorba

V opačném případě je vrácena chybová hláška o tom, že server nebyl nalezen. Příklad chybové hlášky je zachycen na obrázku 4.21.



Obrázek 4.21: Neúspěšné ověření funkčnosti úložiště

Zdroj: vlastní tvorba

Jestliže je úložiště nefunkční je nutné uvědomit infrastrukturní oddělení, které má pro tyto případy připraven záložní server, který okažitě zprovozní. Po zapojení následuje ověření dostupnosti, které probíhá stejným způsobem jako v první fázi tohoto procesu. Pokud server vrátí negativní odezvu, je nutné problém dále s infrastrukturou komunikovat a problém odstranit.

Je-li záložní úložiště dostupné, následuje úprava funkce pro extrakci dat, kde je nezbytné změnit adresu zdroje. Tuto změnu provádí už člen týmu Business Intelligence, který je odpovědný za procesing dat.

Jakmile je příkaz upraven, následuje ověření přístupových oprávnění na daný server. Toto ověření probíhá automaticky při spuštění procedury pro proces extrakce, která má za úkol v jedné fázi se připojit na zdrojové úložiště. Pokud toto úložiště není dostupné, je vypsána chybová hláška a následuje kontaktování administrátora úložiště, který je povinnen přidělit přístupová oprávnění pro daný účet.

Ve chvíli, kdy jsou přístupová práva přidělena následuje ověření, zda je konkrétní soubor dostupný na předdefinovaném zdrojovém uložišti. Zodpovědnost za dostupnost zdrojového souboru nese poskytovatel dat, proto dochází k jeho kontaktování ve chvíli, kdy data na tomto uložišti dostupná nejsou. Celý problém je s poskytovatelem komunikován, nejčastěji prostřednictvím emailové komunikace. Bohužel proti riziku, kdy dojde k pochybení na straně poskytovatele, je pro členy týmu Business Intelligence velmi obtížné se bránit a tato rizika eliminovat.

Je-li soubor dostupný na definovaném uložišti, následuje kontrola formátu, ve kterém byl soubor vyexportován. Za bezchybný export do předem určeného formátu a struktury odpovídá poskytovatel dat. Pokud export není v pořádku, následuje kontaktování poskytovatele, který je povinen chybu odstranit a provést nový export dat.

V následující fázi tohoto procesu probíhá ověření dostupnosti cílového úložiště. Pokud není cílové úložiště dostupné, následuje stejný postup jako v případě, kdy nebylo dostupné zdrojové uložistě. V opačném případě následuje pokus o uložení dat na cílové úložiště.

Při ukládání dat na cílové úložiště dochází k ověření, zda má daný účet práva pro zápis na server. Pokud tomu tak není, musí být kontaktovám administrátor úložiště a nepordleně přístupová práva přidělena. Dalším z důvodů, kdy nemusí uložení souboru dat skončit úspěšně, je situace, ve které je na cílovém uložišti málo místa. V tomto případě musí

odpovědný pracovní Business Intellingence odstranit některé již nepotřebné soubory a pokus o uložení souboru opakovat.

V případě, kdy zdrojový server je dostupný probíhá velmi podobný scénář, který jsem okomentovala výše.

Diagram číslo 3.5 je věnován implementaci preventivních opatření v rámci procesu extrakce, konkrétně fáze nahrání dat do „source“ databáze. Některá z mnou navrhovaných opatření nelze ošetřit pomocí jazyka SQL, ale je nutný zásah lidského faktoru.

Jelikož je proces extrakce rozdělen do dvou částí, doporučuji provést opětovné ověření dostupnosti úložiště. Toto ověření probíhá jako první krok procesu. Pokud úložiště není dostupné, získáme o tomto faktu chybovou hlášku. V takovém případě navrhuji spustit ověřovací dotaz, pomocí něhož zjistím, zda je server dostupný. Pokud je server dostupný, dojde o opakování ověření dostupnosti serveru. V opačném případě je nezbytné o tomto faktu informovat infrastrukturní oddělení, které zprovozní záložní úložiště, které je v záloze právě pro tyto případy. Následně je nezbytné, aby došlo k úpravě definice cílového úložiště, které máme definované v předchozí fázi procesu extrakce, tedy procesu extrakce – získání zdrojových dat. Po této úpravě musí být tato fáze procesu zopakována. Pokud je soubor uložen na záložní úložiště v pořádku, následuje opětovné spuštění fáze extrakce – nahrání dat do „source“ databáze.

Pokud je úložiště dostupné, následuje ověření přístupových oprávnění na úložiště. Jsou-li práva přidělena, dochází k ověření přístupových oprávnění do databáze. Pro tuto fázi procesu extrakce postačí, když pro účet, pod kterým je proces extrakce – nahrání dat do „source“ databáze - jsou přidělena práva pro čtení z úložiště a práva pro zápis do „source“ databáze. Nejsou-li práva přidělena, je nezbytné kontaktovat administrátora úložiště, databáze a požádat jej o přidělení práv. Po jejich přidělení opakujeme fázi ověření oprávnění k úložišti.

Jak je již uvedeno výše, pro proces extrakce dat je nezbytné předem vydefinovat strukturu souboru, ve kterém budou data poskytnuta ze zdrojového systému. Při každém nahrávání dat do databáze probíhá kontrola, zda je tato struktura dodržena. Jestliže při fázi kontroly zjistíme, že struktura není dodržena, následuje ověření, zda je struktura definována dobře v procesu. Pokud zjistíme, že tomu tak není, je nezbytné kontaktovat vývojáře, který je zodpovědný za daný projekt. Vývojář strukturu upraví dle předem domluvených struktur a proces kontroly se opakuje. Je-li struktura definována vývojářem správně, je nezbytné kontaktovat poskytovatele dat a znovu definovat strukturu, ve které budou data poskytována.

Po domluvě následuje úprava struktury v procesu extrakce a ověření struktury souboru se opakuje.

Poslední fází procesu extrakce je nahrání dat do „source“ databáze. Jestliže pokus skončí chybou, je nezbytné provést kontrolu, zda odpovídá definice příslušné tabulky, do které nahráváme data. Pokud struktura jednotlivých atributů odpovídá, je možné pokus o nahrání dat opakovat. V opačném případě je nezbytné upravit jednotlivé atributy dle definice a následně nahrání dat do databáze zopakovat. Jsou-li data nahrána do databáze, proces extrakce je ukončen.

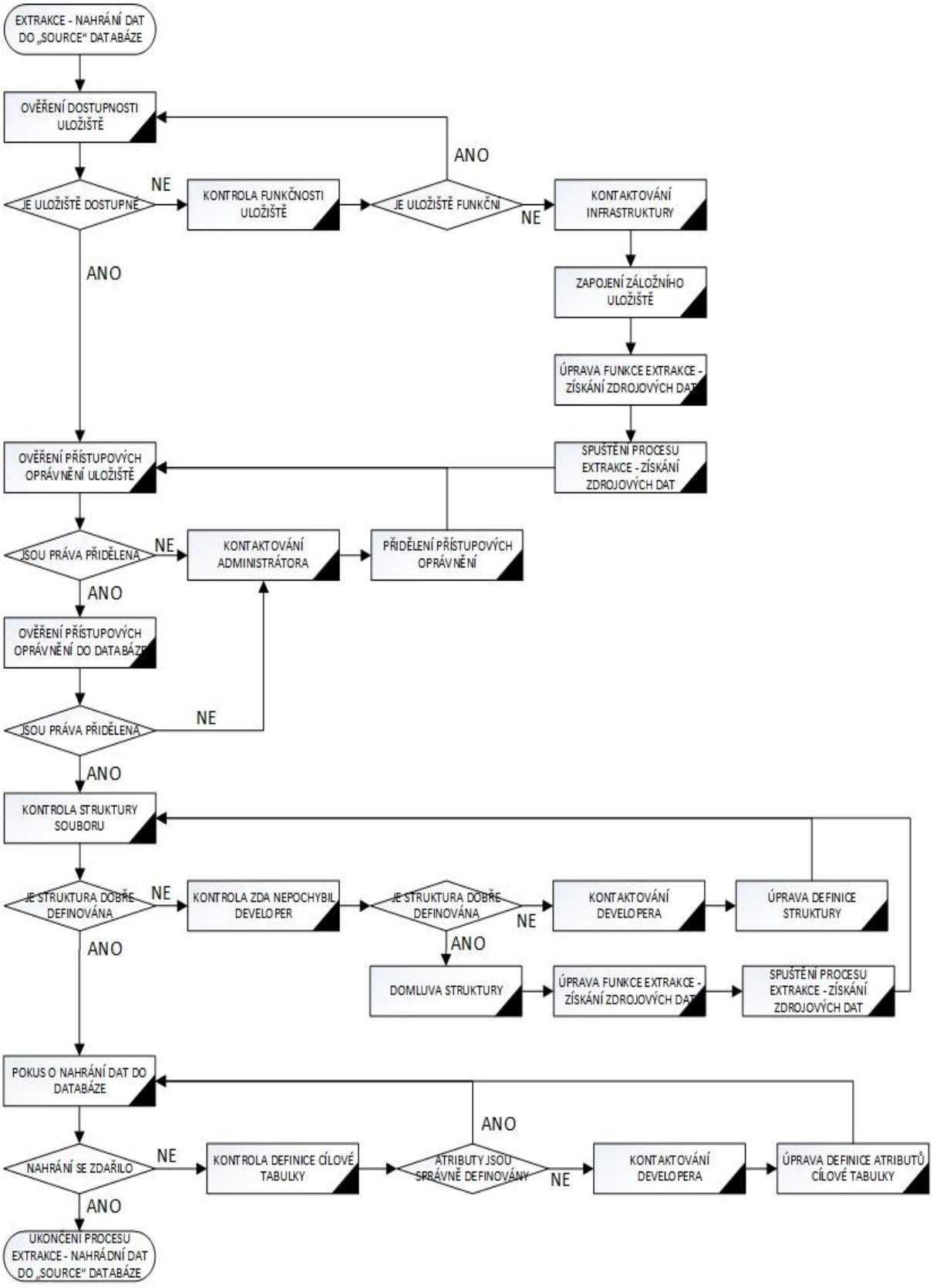


Diagram 3. 5: Vývojový diagram - proces extrakce nahrání dat do „soruce" databáze po provedení změn

Zdroj: vlastní tvorba

4.2 ELIMINACE POTENCIÁLNÍCH INFRASTRUKTURNÍCH RIZIK

Jak jsem již uvedla v předešlé kapitole, eliminace infrastrukturních rizik je primárně řešena přepojením na záložní server. Samozřejmě toto řešení je finančně nákladné, ale v běžné praxi existují odvětví, kde výpadek ETL procesů způsobí mnohem větší ztráty než jsou náklady na pořízení a zprovoznění záložního úložiště.

V běžné praxi je nezbytně nutné pro úspěšnou eliminaci těchto rizik spojit síly několika pracovníků infrastrukturního oddělení a oddělení Business Intelligence. Tento tým jedná v úzké spolupráci a v případě potíží je připraven úzce spolupracovat. Lze říci, že se jedná o jakýsi úzký tým lidí, který je vytvořen napříč dvěmi odděleními.

4.3 ELIMINACE POTENCIÁLNÍCH RIZIK VZNIKLÝCH POCHYBENÍM LIDSKÉHO FAKTORU A POCHYBENÍ ADMINISTRÁTORA

Snížit výskyt rizik vzniklých pochybením člověka je velmi obtížné. V souvislosti s extrakcí dat ze zdrojového systému dochází k pochybení lidského faktoru ve většině případů z nepozornosti, případně z nedostatku znalostí a praxe v oboru.

V souladu s návrhem preventivních opatření mne napadá zavedení pravidelných školení zaměstnanců na juniorních pozicích, kdy osoby služebně starší a s většími zkušenostmi by předávaly své znalosti osobám na nižších pozicích. Tato školení by mohla probíhat v rámci oddělení.

Další možností jak předcházet často se opakujících rizikům, která vzniknou díky nedbalosti zaměstnance je zavedení penalizací v rámci snižování platu, případně pololetních odměn.

4.4 ELIMINACE POTENCIÁLNÍCH RIZIK VZNIKLÝCH V SOULADU S NENADÁLOU UDÁLOSTÍ

Pomocí vybraných metod analýzy rizik jsem identifikovala dva faktory, které mohou nepříznivě ovlivnit proces extrakce v souladu se vznikem nenadálé události. Těmito faktory jsou:

- požár,
- potopa.

Výše zmiňované živly mohou být způsobeny vlivem přírodních živlů nebo nečekanou poruchou vodoinstalace, elektroinstalace. Z tohoto důvodu je nesmírně důležité zabezpečit serverovnu právě proti těmto vlivům.

V příloze číslo 4 je zachycena fotodokumentace zabezpečení serverovny proti případnému požáru. Na fotografii číslo 1 je zachycen detektor požáru, někdy také označovaný jak čidlo identifikující požár. Toto čidlo snímá a vyhodnocuje vzorky vzduchu v serverovně. Ve chvíli, kdy alespoň dva detektory identifikují požár zašlou informaci do ústředy požární signalizace, rozezní se bzučák na řídícím panelu, poplašný signál se podle plánu zašle dál do místnosti se stálou obsluhou a vypnou se ventilační a klimatizační jednotky. Odezva druhého automatického hlásiče požáru spustí hlavní sirény, označující požár, a zahájí se proces hašení. Rozezní se elektrická siréna a spustí se běh nastavené doby zpoždění pro evakuaci v délce 10 sekund. Po uplynutí této doby zpoždění se otevře ventil ocelové nádoby s hasivem, zachycený na fotografii číslo 3. V průběhu likvidace požáru dochází k úbytku energie plamenů v závislosti na snižování koncentrace kyslíku v místnosti. Po úspěšném uhašení požáru následuje uvolnění odvětrávací šachy, viz. fotografie číslo 4, a následuje kompletní dovětrání serverovny.

Příloha číslo 5 obsahuje fotodokumentaci preventivních opatření proti výskytu vody v serverovně. Tato opatření jsou tvořena čidlem, které snímá hladinu vody na podlaze a zasílá výsledky do hlásiče stavu vody. Pokud je v serverovně identifikována voda, hlásič upozorní odpovědné pracovníky na výskyt vody.

ZÁVĚR A NÁVRHY NA POKRAČOVÁNÍ

Cílem mé diplomové práce byla identifikace a případná eliminace potenciálních rizik, které mohou být naplněna v rámci realizace ETL procesů.

V souladu se zadáním diplomové práce jsem si vybrala proces extrakce, který jsem rozdělila na dvě navzájem závislé části, a provedla analýzu rizik pomocí metody FTA a analýzy příčin a následků. Tyto dvě metody jsem prováděla v časovém rozestupu, abych co nejvíce eliminovala riziko ovlivnitelnosti.

Pomocí mnou vybraných metod jsem identifikovala čtyři faktory, které nepříznivně ovlivňují proces extrakce dat ze zdrojového systému. U každého faktoru jsem následně identifikovala dílčí rizika, která působí na výše zmiňovaný proces.

Na základě vlastních zkušeností, které jsem měla možnost získat v průběhu praxe v oboru Business Intelligence, jsem navrhla preventivní opatření. Od těchto návrhů očekávám minimalizaci, případně úplné odstranění identifikovaných dílčích rizik. Z osobní zkušenosti vím, že se jedná o finančně nákladná opatření, ale za to velmi efektivní.

Tuto diplomovou práci je možné rozšířit o zbylé dva ETL procesy, konkrétně transofrmaci a loadování, kdy ke každému z nich je možné provést identifikaci rizik pomocí výše zmiňovaných metod a v neposeldní řadě jejich popis pomocí vývojových diagramů a definic v SQL kódu.

SEZNAM POUŽITÉ LITERATURY – KNIŽNÍ ZDROJE

[1] ČERMÁK, Miroslav. *Řízení informačních rizik v praxi*. 1. vyd. Brno: Tribun EU, 2009. ISBN 978-80-7399-731-1.

[2] DOSTÁL, Petr, Karel RAIS a Zdeněk SOJKA. *Pokročilé metody manažerského rozhodování : konkrétní příklady využití metod v praxi*. 1. vyd. Praha: Grada, 2005. ISBN 80-247-1338-1.

[3] KOCH, Miloš a Bernard NEUWIRTH. *Datové a funkční modelování*. čtvrté rozšířené vydání. Brno: AKADEMICKÉ NAKLADATELSTVÍ CERM, s. r. o., 2010. ISBN 978-80-214-4125-5.

[4] LABERGE, Robert. *Datové sklady: Agilní metody a business intelligence*. Brno: Computer Press, a. s., 2012. ISBN 978-80-251-3729-1.

[5] LACKO, Luboslav. *Business Intelligence v SQL Serveru 2008: Rozhodovací, analytické a další datové služby*. první vydání. Brno: Computer Press, a.s., 2009. ISBN 978-80-251-2887-9.

[6] NOVOTNÝ, Ota, Jan POUR a David SLÁNSKÝ. *Business Intelligence: Jak využít bohatství v našich datech*. první vydání. Praha: Grada Publishing, a. s., 2005. ISBN 80-247-1094-3.

[7] OPPEL, Andy. *SQL bez předchozích znalostí: průvodce pro samouky*. první vydání. Brno: Computer Press, a. s., 2008. ISBN 978-80-251-1707-1.

[8] PONNIAH, Paulraj. *DATA WAREHOUSING FUNDAMENTALS: A Comprehensive Guide for IT Professionals*. 1. vyd. New York: JOHN WILEY & SONS, INC, 2001. ISBN 0-471-22162-7.

[9] PROCHÁZKOVÁ, Dana. *Analýza a řízení rizik*. V Praze: České vysoké učení technické, 2011. ISBN 9788001048412.

[10] STEPHENS, Ryan K. a Ronald R. PLEW. *Naučte se SQL za 21 dní*. první vydání. Brno: Computer Press, 2004. ISBN 80-722-6870-8.

[11] SMEJKAL, Vladimír a Karel RAIS. *Řízení rizik*. 1. vyd. Praha: Grada, 2003. ISBN 80-247-0198-7.

[12] SMEJKAL, Vladimír a Zdeněk SOJKA. *Řízení rizik ve firmách a jiných organizacích*. 4. vyd. Praha: Grada, 2013, 134 s. ISBN 978-80-247-4644-9.

[13] ŠEFČÍK, Vladimír. *Analýza rizik*. Vyd. 1. Zlín: Univerzita Tomáše Bati ve Zlíně, 2009. ISBN 9788073186968.

[14] TICHÝ, Milík. *Ovládání rizika : analýza a management*. 1. vyd. Praha: C.H. Beck, 2006. ISBN 80-7179-415-5.

[15] ŽIŽKA, Jan. *Business Intelligence*. první vydání. Praha: Vysoká škola ekonomie a managementu, 2011. ISBN 978-80-86730-79-0.

SEZNAM POUŽITÉ LITERATURY – ONLINE ZDROJE

[16] Analýza rizik. *Cleverandsmart.cz* [online]. 2013 [cit. 2014-01-15]. Dostupné z: http://www.cleverandsmart.cz/analyza-rizik-jemny-uvod-do-analyzy-rizik/

[17] Ikvalita.cz. *Portál pro kvalitáře* [online]. 2013 [cit. 2014-02-10]. Dostupné z: http://www.ikvalita.cz/tools.php?ID=52

[18] Teorie relačních databází: Normalizace. *Manualy.net* [online]. 2007 [cit. 2014-01-15]. Dostupné z:http://www.manualy.net/article.php?articleID=13

[19] SCHILLER, Martin. Co se skrývá pod zkratkou ETL?. In: *Systemonline.cz* [online]. 2003 [cit. 2014-01-05]. Dostupné z: http://www.systemonline.cz/clanky/co-se-skryva-pod-zkratkou-etl.htm

[20] SPÁLENKA, Ondřej. *Případová studie aplikace ETL procesů v bankovním prostředí*. Praha, 2009. Dostupné z: http://is.bivs.cz/th/6357/bivs_b/SpalenkaOndrej_BIVS_ Bakalarska Prace.pdf.

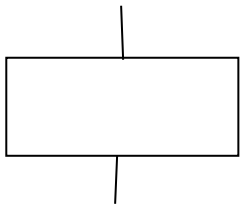
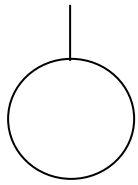
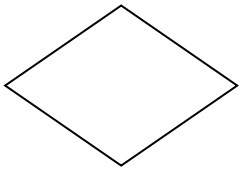
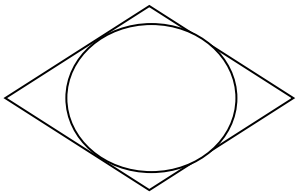
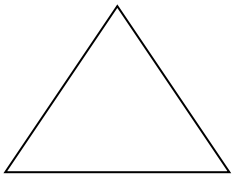
SEZNAM TABULEK

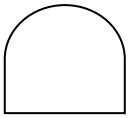
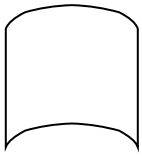
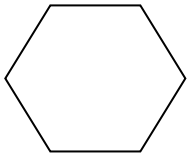
SEZNAM OBRÁZKŮ

SEZNAM DIAGRAMŮ

SEZNAM PŘÍLOH

Příloha č. 1: FTA symboly a jejich význam

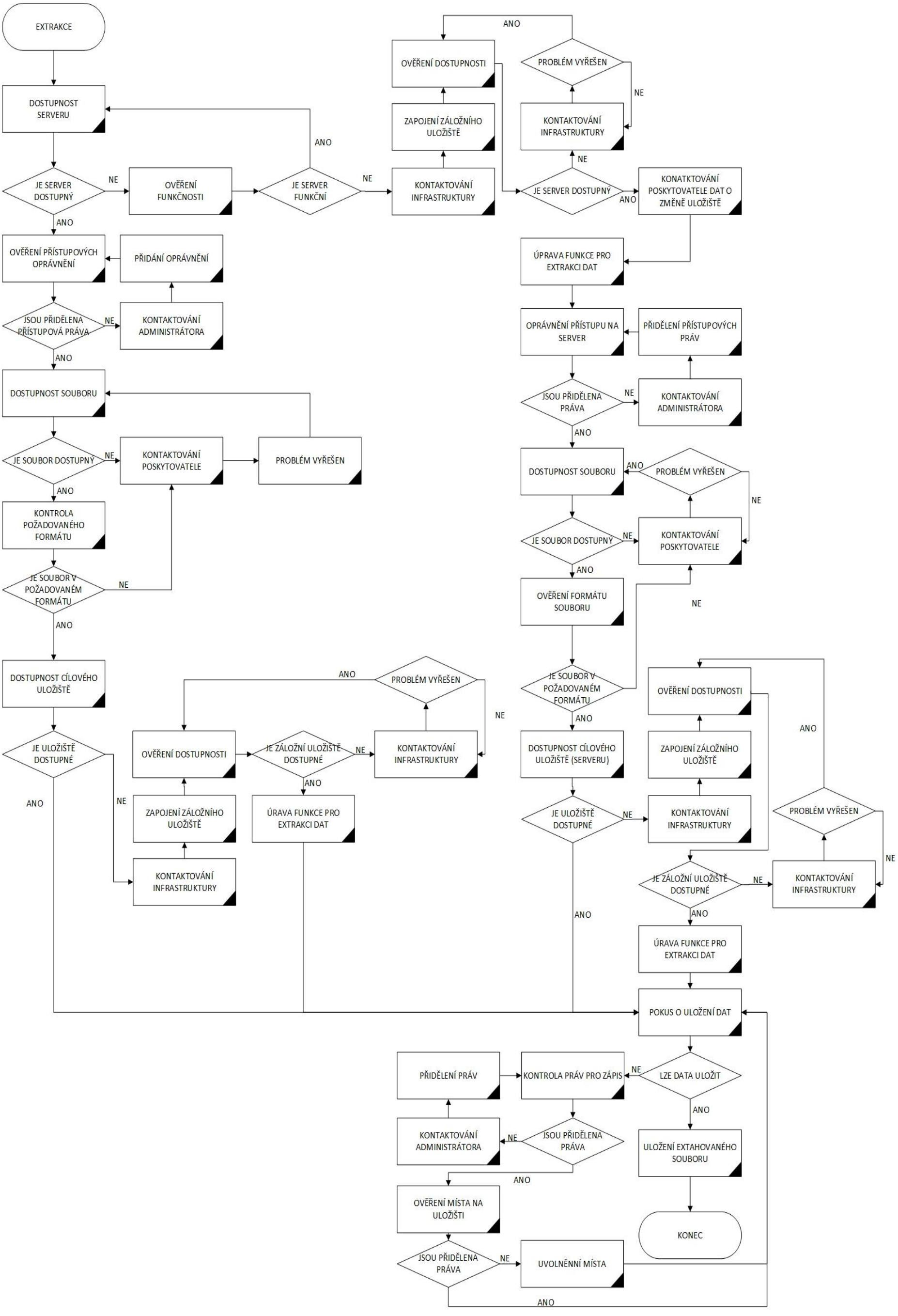
Příslušný symbol	Alternativní symbol	Význam symbolu
		Blok označující vrcholovou událost; začátek analýzy stromu chyb
		Blok označující událost; obsahuje pravděpodobnost četnosti jevu
		Koncová událost, která dále nepokračuje
		Událost, kterou nechceme dále rozvíjet, není předmětem požadované analýzy
		Jev, který je rozvíjen v jiném stromu chyb
		Pomocný symbol pro přehlednost analýzy; označení, že problém je analyzován jinde

&		Jevy nastávají za předpokladu, že nastávají společně
>=1		Nastane alespoň jedna vstupní událost
>=m	m/n	Jev nastane za předpokladu, že nastane alespoň m z n událostí
		Událost nastane pod podmínkou, že nasstane vstupující událost a současně bude spolněna podmínka, která je uvedena ve vnitřku tohoto symbolu

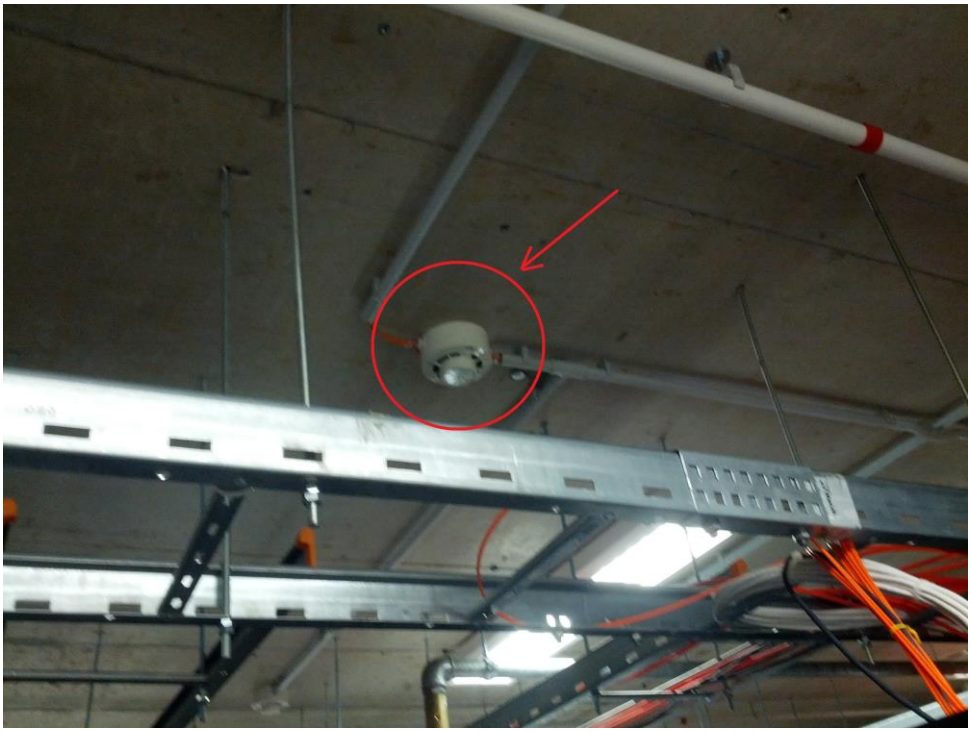
Příloha č. 2: Vzorový formulář pro vypracování analýzy rizik pomocí metody FMEA

FMEA - formulář	odpovědný obor: příslušný obor: příslušný dodavatel:										Jméno součásti: Číslo součásti: Vyhotovil: List č.: z Datum:				
Atribut	pot. chyba	pot. následky chyby	pot. příčiny chyby	současný stav preventivní a kontrol. opatření	P	V	O	PRČ	doporučené hav. opatření	závazný termín	zlepšený stav vhodná opatření	P	V	O	PRČ

Příloha č.3: Vývojový diagram – proces extrakce získání zdrojových dat



Příloha č.4: Fotodokumentace zabezpečení serverovny proti požáru



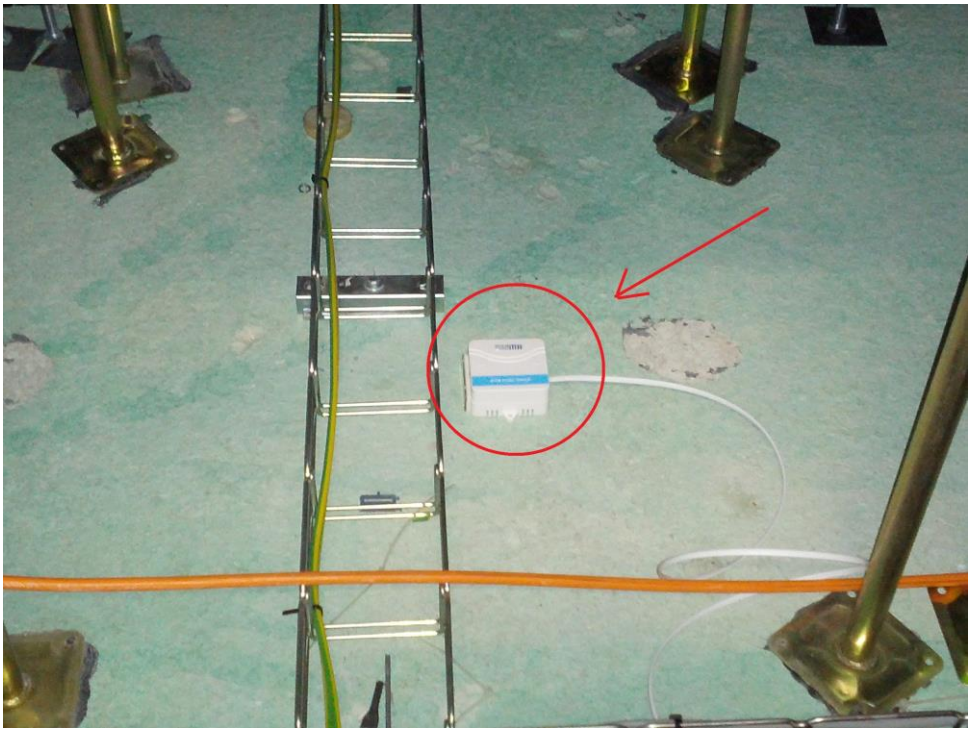
Fotografie 1: Detektor požáru

Fotografie 2: Ústředna požární signalizace a hasícího zařízení

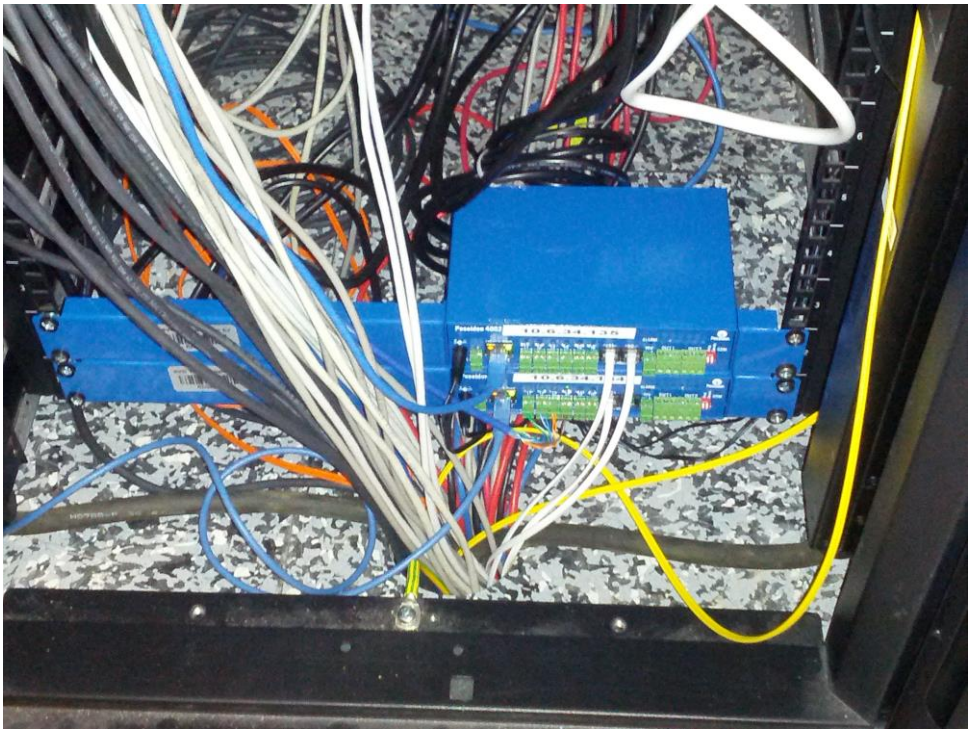
Fotografie 3: Hasící zařízení

Fotografie 4: Odvětrávací šachta

Příloha č.5: Fotodokumentace zabezpečení serverovny proti povodni



Fotografie 5: Čidlo detekující vodu v serverovně



Fotografie 6: Hlásící zařízení

Příloha č. 6: Extrakce – získání dat ze zdrojového uložiště

```
:: ------------------------------------------------------------------------
:: Copy file from %1 to %2
::
::
:: USAGE NOTES:
::
:: PARAMETERS:
:: %1 ... source folder
:: %2 ... destination folder
::
:: EXAMPLE:
::copy.cmd "source folder" "destination folder"
:: ------------------------------------------------------------------------
copy %1 %2
```

Příloha č. 7: Extrakce – nahrání zdrojových dat do source databáze – SQL kód

```
USE [SRC]
GO

CREATE PROC [dbo].[LOAD_SRC_DATA] @fordate datetime = null, @filename sysname,
@fileformat nvarchar(10), @DataFolder varchar(1024), @debug bit=1, @ProjectName
varchar(255)
AS

DECLARE @bulk_insert varchar (max),
             @extract_fileX  varchar(max),
             @err_message nvarchar(512) = NULL,
             @sqlstm VARCHAR(MAX) = '',
             @extract_date VARCHAR(10)

-- ProjectName
IF @ProjectName IS NULL
SET @ProjectName = 'DefaultProject'
-- Get data folder
IF @DataFolder IS NULL
SET @DataFolder = (SELECT [DATAFOLDER] FROM dbo.PROJECT WHERE PROJECT_NAME =
@ProjectName)

-- Fordate
IF @fordate IS NULL
       SET @fordate = (SELECT CONVERT(datetime, GETUTCDATE()))
--

--
SET @extract_date =
CONVERT(varchar,@fordate,112)+SUBSTRING(CONVERT(varchar,@fordate,114),1,2)
SET @extract_fileX = @DataFolder + @filename + '.' + @fileformat
--

--Bulk insert
SET @bulk_insert = 'BULK INSERT [dbo].[SRC_DATA_'+@filename+']
                           FROM '''+@extract_fileX+'''
                           WITH (DATAFILETYPE  = ''char'',
                           KEEPNULLS,
                    TABLOCK,
                    FIRSTROW = 1,
                           ROWS_PER_BATCH =100000,
                           FIELDTERMINATOR = '','',
                           ROWTERMINATOR =''0x0A'',
                           MAXERRORS = 1
                           )'

--
--
SELECT @bulk_insert = REPLACE(@bulk_insert,'YYYYMMDDHH',@extract_date)
--
--
SET @sqlstm = 'TRUNCATE TABLE [dbo].[SRC_DATA_'+@filename+']'
--
-- Run/Print statement
IF @debug = 0
       BEGIN
              -- truncate table
              EXEC(@sqlstm)
```

```
        -- execute bulk insert command
        BEGIN TRY
                EXEC(@bulk_insert)
                IF @@ROWCOUNT = 0
                BEGIN
                RAISERROR('Trying to load 0 rows. Check extract file.',16,1)
                END
        END TRY
                BEGIN CATCH
                        SET @err_message = (SELECT ERROR_MESSAGE())
                        RAISERROR(@err_message, 16, 1)
                END CATCH
END
```