VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA PODNIKATELSKÁ
ÚSTAV INFORMATIKY

FACULTY OF BUSINESS AND MANAGEMENT
INSTITUTE OF INFORMATICS

NÁVRH DÍLČÍ ČÁSTI INFORMAČNÍHO SYSTÉMU FIRMY

PROPOSAL OF PART OF COMPANY INFORMATION SYSTEM

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — VÁCLAV BŘEZINA
AUTHOR

VEDOUCÍ PRÁCE — Ing. JAN LUHAN, Ph.D.
SUPERVISOR

BRNO 2015

Vysoké učení technické v Brně
Fakulta podnikatelská

Akademický rok: 2014/2015
Ústav informatiky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Březina Václav

Manažerská informatika (6209R021)

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách, Studijním a zkušebním řádem VUT v Brně a Směrnicí děkana pro realizaci bakalářských a magisterských studijních programů zadává bakalářskou práci s názvem:

Návrh dílčí části informačního systému firmy

v anglickém jazyce:

Proposal of Part of Company Information System

Pokyny pro vypracování:

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

Seznam odborné literatury:

BEGG, C., R. HOLOWCZAK a T. CONOLLY. Mistrovství - Databáze: Profesionální průvodce tvorbou efektivních databází. Praha: Computer Press, 2009. 584 s. ISBN 978-80-251-2328-7.
BRUCKNER, T., J. VOŘÍŠEK, A. BUCHALCEVOVÁ A. a kol. Tvorba informačních systémů: Principy, metodiky, architektury. Praha: Grada Publishing a.s., 2012. 360 s. ISBN 978-80-247-4153-6.
ŘEPA, V. Analýza a návrh informačních systémů. 1. vyd. Praha: EKOPRESS, 1999. 403 s. ISBN 80-86119-13-0.
SCHWALBE, K. Řízení projektů v IT: Kompletní průvodce. Praha: Computer Press, 2011. 632 s. ISBN 978-80-251-2882-4.

Vedoucí bakalářské práce: Ing. Jan Luhan, Ph.D.

Termín odevzdání bakalářské práce je stanoven časovým plánem akademického roku 2014/2015.

L.S.

doc. RNDr. Bedřich Půža, CSc.	doc. Ing. et Ing. Stanislav Škapa, Ph.D.
Ředitel ústavu	Děkan fakulty

V Brně, dne 28.2.2015

Abstrakt

Tato bakalářská práce řeší návrh části informačního systému pro společnost ABB s. r. o. V první části jsou popsána teoretická východiska práce a nastíněna tak pozadí zkoumané problematiky. Další části jsou zaměřeny na analýzu současného stavu podniku a návrhu informačního systému dle požadavků. Cílem bakalářské práce je vytvořit databázi pro informační systém a jeho grafické rozhraní.

Abstract

This bachelor thesis solves the design of section of information system for company ABB s. r. o. In the first part of thesis, I will deal with theoretical formulation of basic concepts. Next part objective is analysis of current state of the company and creation of information system. The goal of the thesis is to create database and web interface of the IS.

Klíčová slova

SQL, databáze, normalizace, informační systém, informace, návrh IS

Key words

SQL, database, normalization information system, information, creation of an information system

Bibliografická citace bakalářské práce

BŘEZINA, V. *Návrh dílčí části informačního systému firmy.* Brno: Vysoké učení technické v Brně, Fakulta podnikatelská, 2015. 53 s. Vedoucí bakalářské práce Ing. Jan Luhan, Ph.D..

Čestné prohlášení

Prohlašuji, že předložená bakalářská práce je původní a zpracoval jsem ji samostatně. Prohlašuji, že citace použitých pramenů je úplná, že jsem ve své práci neporušil autorská práva (ve smyslu Zákona č. 121/2000 Sb., o právu autorském a o právech souvisejících s právem autorským).

V Brně dne …….. ……

podpis studenta

Poděkování

Touto cestou bych rád poděkoval panu Ing. Janu Luhanovi, Ph.D. za odborné vedení bakalářské práce, ochotný přístup a cenné rady. Dále firmě ABB s. r. o. za poskytnutí potřebných informací. V neposlední řadě také rodině, která mě podporovala po dobu mého studia.

OBSAH

ÚVOD

Základem každého podniku je sdílení informací. Pokud jde o malou firmu, kde si lze tyto informace sdělit osobně, odpadá mnoho problémů. V případě velké firmy a velkého objemu dat je toto sdílení informací o mnoho složitější. Nelze komunikovat se stovkami kolegů současně. A právě sdílením interních informací mezi odděleními se zabývá tato bakalářská práce.

V této bakalářské práci bude navrhnut informační systém firmě ABB s. r. o., konkrétně divizi Přístrojových transformátorů a senzorů. Firma ABB působí v České republice prostřednictvím svých výrobků a služeb již od roku 1970. Formální vznik společnost v České republice je datován k roku 1992.

Celá práce je rozdělena do několika částí. První část obsahuje teoretické informace zabývající se informačními a databázovými systémy a technologiemi, které jsou použity v praktické části práce. Druhá část analyzuje současný stav firmy ABB s. r. o. a definuje požadavky na informační systém. Třetí část obsahuje návrh informačního systému.

Vznik tohoto infomačního systému usnadní sdílení informací mezi všemi odděleními divize Přístrojových transformátorů a senzorů firmy ABB s. r. o., ale také zefektivní proces vyhledávání interních informací. Informační systém obsahuje dva moduly. Modul s pracovním názvem „Odkazy“ a modul s názvem „Smlouvy“,

Jelikož firma disponuje sofistikovanými informačními technologiemi, není potřeba dalších investic. Informační systém bude pracovat v prostředí webového prohlížeče, který je standartní aplikací každé pracovní stanice ve firmě.

1. VYMEZENÍ PROBLÉMU A CÍLE PRÁCE

Ve firmě, pro kterou je informační systém navrhován, dosud nic podobného neexistuje a není využíváno. V současné době probíhá hledání informací procházením složek síťového uložiště a v případě odkazu na stránky třetích stran hledáním cílové adresy. Toto „procházení a hledání“ zabere spoustu času, které zaměstnance zdržuje ve své práci a pokud zaměstnanec vyhledává tyto data poprvé, nemusí být úspěšný z důvodu jejich neznámého umístění, nebo nelogického pojmenování souboru. Dalším problémem je sdílení smluv. Většina papírových smluv je uložena u odpovědných osob. Ale lokace jsou neznámé. Proto jsou v informačním systému uloženy základní údaje o smlouvě, včetně odpovědné osoby a také je zde uložena elektronická verze smlouvy.

Na základě analýzy současného stavu informačních technologií a požadavků vedoucích jednotlivých oddělení je cílem práce návrh informačního systému s uživatelsky příjemným rozhraním, které usnadní vyhledávání informaci a zkrátí dobu vyhledávání, pomocí rozhraní webového prohlížeče.

Dílčí cíle

- Seznámení se s problematikou informačních a databázových systému
- Návrh databáze a webové stránky
- Srovnání výsledného IS s požadavky

2. TEORETICKÉ VÝCHODISKO PRÁCE

2.1. Informační systém

2.1.1. Systém a informační systém

Systém lze definovat jako množinu prvků a vazeb. Prvky systému na dané úrovni jsou chápány jako nedělitelné. Komunikace mezi prvky systému může probíhat jednosměrně nebo obousměrně. Charakteristikou jednosměrné komunikace je role komunikátora a komunikanta. Komunikátor vysílá sdělení a komunikant je pouze přijímá. Opakem jednosměrné komunikace je obousměrná komunikace, kde oba prvky systému mají roli komunikátora a komunikanta.(1)

Je-li toto definice systému, informační systém je chápán jako uspořádání vztahů mezi lidmi, datovými a informačními zdroji a jejich zpracování za účelem dosažení stanovených cílů. Z hlediska informačního obsahu lze zmínit rozlišení mezi daty a informacemi. Na nejnižší úrovni jsou rozpoznávány signály, které slouží jako nosiče dat. Signály jsou chápány jako něco, co už je pevně dané a mění se v čase. Daleko důležitějšími pojmy jsou data a informace. Data jsou rozpoznané signály, které vypovídají o situacích a stavech sledovaných a řízených objektů. Zpracováním těchto dat se z nich stávají informace. Informace jsou používány uživateli k rozhodování. Stejná data mohou pro různé uživatele mít různý význam a tím také mohou být různými informacemi. (1)

Kdybychom měli shrnout informace zmíněné výše a popsat co to vlastně informační systém je, tak informační systém lze chápat jako soubor lidí, technologických prostředků a metod, který uchovává a zpracovává data pro potřeby uživatelů.

2.2. Databáze

Databáze lze chápat jako organizovaný systém informací. Předchůdcem dnešních elektronických informací byly papírové kartotéky. V těchto „papírových databázích“ byla data zakládána a kategorizována podle jednotlivých dat. Všechny tyto operace byla prováděna lidskou obsluhou.(2)

Příchodem výpočetní techniky bylo uchovávání dat přeneseno z papírové do elektronické podoby. Ale tato data nejsou v databázi nepřehledně uložena. Databáze mají svoji strukturu, které umožňují snadné vyhledávání a třídění. Toto umožňuje data efektivně zpracovávat a vyhodnocovat. Také to umožňuje se lépe rozhodnout a činit lepší rozhodnutí na základě dat obsažených v databázích. Dnešní databáze mohou obsahovat velké množství dat a velkou výhodou je že mohou být používána současně mnoha uživateli.(3)

2.2.1. Databázové systémy

Uživatelé pomoci těchto systémů vytváří a spravují databáze a také generují potřebné informace z dat uložených v databázi. Tyto systémy mohou být klasickými aplikacemi, nebo v dnešní době typičtějšími online aplikacemi. Tyto aplikace lze psát v programovacích jazycích jako je např. Java, C++ nebo C#.(4)

2.3.Databázové modely

Před vznikem prvních databázových modelů, byla data ukládána do nesouvisejících polí. Pro programátory byla extrakce těchto dat velmi složitá. Jejich programy musely provádět složité algoritmy a vytvářet vztahy.

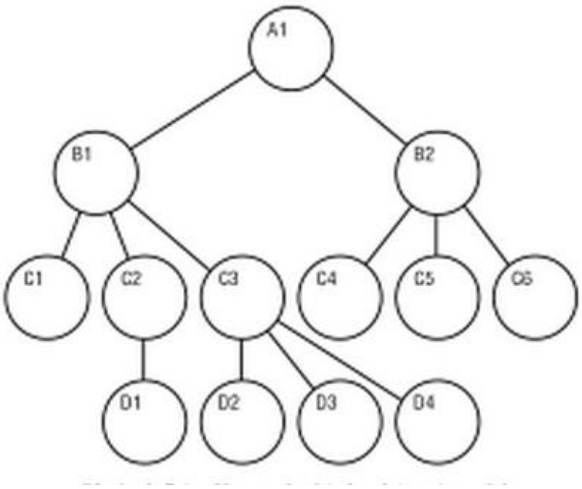
Vznikem databázových modelů byly vyvinuty standarty pro přístup a aktualizaci dat. Pomocí prostřednické vrstvy mezi aplikací a daty se nemusí programátor zabývat přístupem k datům, ale věnovat se vývoji aplikace. (5)

Historicky prvním modelem byl tzv. hierarchický model. Standarty tohoto modelu byly definovány v roce 1970. Zobecněním tohoto modelu vznikl model síťový. Tento model odstraňoval nedostatky předchozího modelu. Změnou struktury databáze přišel relační datový model založený na teoretických matematických základech. (6)

Níže budou vysvětleny principy výše zmíněných modelů a nastínění jejich dostatků a nedostatků. Největší důraz bude kladen na relační datový model, který byl zvolen pro praktické řešení této bakalářské práce.

2.3.1. Hierarchický model

Je nejstarším databázovým model. Jeho podobu je možné si představit jako převrácený strom. Základním prvkem tohoto modelu je role rodič – potomek. Rodič může mít více potomků, ale potomek může mít pouze jednoho rodiče. Příkladem pro lepší představu si lze představit souborový adresář. Adresář nejvyšší úrovně (kořenový adresář) obsahuje další adresáře (podadresáře) a například soubory. Tyto podadresáře mohou obsahovat další podadresáře. Oproti předcházejícímu modelu nesouvisejících polí, byl tento model krok vpřed, ale obsahuje spoustu nevýhod. Zachycuje vhodně vztah one-to-many, kdy rodič má mnoho potomků (např. firma má mnoho poboček). Vztah many-to-many se však obtížně v tomto modelu implementuje. Další nevýhodou je flexibilita modelu. Přidání nových vztahů může způsobit změnu existující struktury. Tato změna může znefunkčnit již existující aplikace a je nutná jejich úprava. Při vývoji aplikace musí programátor znát dobře strukturu modelu, aby mohl přistupovat k datům. Aby dostal požadovaná data, musí znát řetězec mezi oběma prvky. Např. mezi A1 a D4 je tento řetězec A1, B1, C3 a D4. Tento příklad je vidět na obrázku níže. Je zde také vidět proč se změnou struktury aplikace stávají nefunkčními.(5)

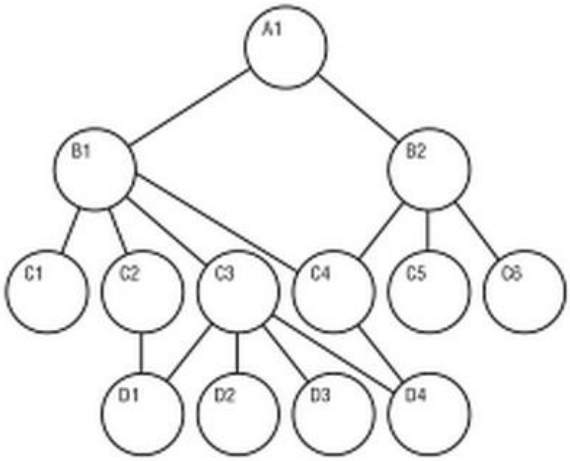


Obrázek 1 Hierarchický databázový model *[zdroj: 5, str. 232]*

2.3.2. Síťový model

Síťový databázový model řeší nedostatky předcházejícího modelu. Hlavně jeho flexibilitu oproti hierarchickému modelu. V tomto modelu už nemusí mít potomek pouze

jednoho rodiče. Potomky označujeme jako členy a rodiče jako vlastníky. Díky těmto vlastnostem může tento model modelovat složitější vztahy. Toto řeší vztah many-to-many. Nevýhodou tohoto modelu je, že se obtížněji implementuje, a i když je pružnější než hierarchický model, stále není tolik flexibilní. Nelze všechny vztahy řešit přiřazením nového vlastníka, a proto programátoři musejí rozumět dobře struktuře dat, aby bylo dosaženo maximálního výkonu. (5)



Obrázek 2 Síťový databázový model*[zdroj: 5, str. 233]*

2.3.3. Relační model

Relační model byl poprvé zveřejněn v roce 1971. To bylo krátkou dobu po vzniku hierarchického modelu. Tento model publikoval pracovník firmy IBM, matematik Dr. Codd. Tento model je založen na teoretických matematických základech. Relační model byl definován jako matematické zobecnění pojmu soubor pomoci matematické relace. Pomocí matematických definic definoval také pojmy jako entita, vazba, atribut a struktura databáze. Tyto pojmy jsou vysvětleny v následující části.

Pojem matematická relace je vhodné si představit jako dvourozměrnou tabulku. V této tabulce každý řádek odpovídá jedné entitě, naopak sloupec představuje atribut. Název každého atributu v rámci relace musí být jedinečný. Databázové relace, na rozdíl od těch matematických, jsou proměnné v čase. Umožňují nám v databázi zachytit změny, které se dějí v reálném světě. Tyto relace mají tyto tabulkové vlastnosti:

- Homogenita sloupce

- Každý údaj je atomickou položkou
- Na pořadí řádků nezáleží
- Na pořadí sloupců nezáleží
- Každý řádek tabulky je jednoznačně identifikovatelný

Charakteristickou vlastností tohoto modelu je realizace vazeb pomoci relace. Relační model realizuje vazbu pomoci primárního klíče. Tyto vazby lze vždy realizovat pomocí vazební tabulky. V případě některých vazeb, jako je 1:1 nebo 1:M, nemusí být použita vazební tabulka, ale pouze je doplněn atribut v tabulce jako primární klíč.

Výhodou relačního modelu je možnost přidání nepředvídatelných vazeb do existující struktury relací bez jejich změny. Pokud je potřeba evidovat něco nového, je definována pouze nová vazební relace pomocí primárních a cizích klíčů. S těmito relacemi se poté pracuje pomocí běžných příkazů.

Relační model představuje implementaci reálného světa vytvořeného podle pravidel relačního modelu, který by měl splňovat 12 pravidel Dr. Codda, která publikoval v roce 1985. Tato pravidla vyjadřují spíše ideální cíl, kterého by měl dosáhnout relační databázový model. (6)

1. *Pravidlo informace: Všechny informace v relační databázi se na logické úrovni reprezentují explicitně hodnotami v tabulkách.*
2. *Pravidlo zaručeného přístupu: Musí být zajištěno, aby úplně každý údaj v relační databázi byl logicky přístupný použitím názvu tabulky, hodnoty primárního klíče a názvu sloupce.*
3. *Systematické ošetření prázdných hodnot: Prázdné hodnoty (nikoliv nuly nebo prázdné řetězce) jsou systematicky plně podporovány RDBMS pro reprezentaci chybějících informací a neplatných informací nezávisle na datovém typu. (Typicky řešeno hodnotou NULL)*
4. *Popis struktury založený na relačním modelu: Popis databáze se na logické úrovni reprezentuje stejně jako běžná data, tzn. v relacích, na které se mohou oprávnění uživatelé dotazovat stejně jako na jakoukoliv jinou relaci.*
5. *Pravidlo komplexního datového jazyka: Relační systémy mohou podporovat více jazyků a režimů přístupu, ale musí existovat minimálně jeden jazyk, jehož*

příkazy jsou vyjádřitelné nějakou dobře definovanou syntaxí jako řetězce znaků, který podporuje:

   a. *Definice dat*
   b. *Definice pohledu*
   c. *Manipulace s daty*
   d. *Omezení integrity*
   e. *Autorizace*
   f. *Vymezení transakce*

6. *Aktualizace pohledu: Všechny aktualizovatelné pohledy je možno aktualizovat systémově.*
7. *Vysokoúrovňová manipulace s daty: Zpracování základní či odvozené relace se jako jediný operand aplikuje jak na vyhledávání, tak na vložení a změnu dat.*
8. *Fyzická datová nezávislost: Aplikace a terminály zůstávají logicky nedotčeny změnami v reprezentaci uložiště nebo v přístupových metodách.*
9. *Logická datová nezávislost: Aplikace a terminály jsou logicky nedotčeny, pokud jsou v tabulkách provedeny změny v uchování informací.*
10. *Nezávislost integrity: Integritní omezení musí být definovatelné v datovém jazyku v databázi, nikoliv v aplikaci.*
11. *Distribuční nezávislost: Databázový jazyk musí být schopen manipulovat s daty umístěnými na jiném počítačovém systému.*
12. *Pravidlo nenarušení: Pokud je v systému více jazyků, žádný z nich nesmí mít možnost manipulovat s daty v rozporu s integritními omezeními.*

Prakticky žádný ze současných systémů nesplňuje všechna pravidla naprosto dokonale a spoustu programátorů databázových aplikací nevyužívá možností databáze. (Týká se hlavně bodu 10). (6, str. 31)

V závislosti na relačním databázovém modelu definoval Dr. Codd jazyk, podle něhož se vyhledávají informace. Manipuluje se s daty. Tento jazyk známe pod zkratkou SQL. Tento pojem bude objasněn níže. (6)

2.4. Fáze návrhu databáze

2.4.1. Konceptuální návrh databáze

Úplně prvotním krokem v návrhu databáze je konceptuální návrh. Jedná se o návrh struktury vazeb a všech potřebných charakteristik. V tomto návrhu se nezohledňují fyzické, resp. technologické charakteristiky databáze, ale je zaměřen na strukturu dat, která není závislá na logickém uspořádání dat zvoleného softwaru pro řízení databází a parametrech paměťových zařízení.(7)

- *Konceptuální datový model musí co nejvěrněji zachytit realitu, např. podniku nebo jeho části či věcné oblasti, pro které je datový model navrhován, tj. všechny jeho typy objektů (entity), vazby a jejich podstatné vlastnosti, ale bez technických detailů.*
- *Konceptuální datový model musí být maximálně přehledný a názorný, aby se v něm analytik i uživatel snadno orientoval. V naprosté většině tedy využívá grafického zobrazení modelované reality.*
- *Na druhé straně musí být konceptuální model konstruován tak, aby byl následně relativně snadno transformován do logického a fyzického návrhu databáze a následně implementován ve zvoleném databázovém modelu.* (7, str. 283)

2.4.2. Logický návrh databáze

Na základě konceptuálního návrhu se řeší logický návrh databáze. V tomto kroku je již vymezena struktura databáze na již zvoleném databázovém modelu. V tomto návrhu jsou definovány struktury databázových tabulek, jejich atributy, primární klíče a vazby mezi entitami. (7)

2.4.3. Fyzický návrh databáze

Z logického návrhu je definován fyzický datový model. Tento návrh specifikuje již potřebné implementační charakteristiky ve vztahu k příslušnému databázovému prostředí, v němž bude databáze realizována. Tím jsou chápány formáty jednotlivých

položek, jejich rozsah, definování cizích klíčů, povolení NULL, či jiné charakteristiky. (7)

2.5.Terminologie

Tato část se týká terminologie datového relačního modelu, který je stěžejní pro praktickou část této bakalářské práce.

2.5.1. Entita

„*Entita je objekt reálného světa, který je schopen nezávislé existence a je jednoznačně odlišný od ostatních objektů. Příkladem entity může být například občan Josef Novák, narozený 22. 2. 1960 v Příbrami, bytem v Olomouci, rodné číslo 600222/1234.“*(8, str. 52)

2.5.2. Vazba mezi entitami

Vazba je spojení mezi entitami. Tato vazba obsahuje dvě základní vlastnosti: kardinalitu a volitelnost. Kardinalita značí, zda vazba jedné entity k druhé nastane pouze jednou nebo vícekrát. Pak se jedná o vztah 1:1, resp. 1:M. U volitelnosti je sledováno, zda vazba mezi entitami musí nastat a je tedy povinná, nebo nemusí nastat a je volitelná. (9)

Vztah 1 : 1 lze chápat jako vazbu, kdy jednomu záznamu z tabulky A odpovídá pouze jeden záznam z tabulky B. Naopak vztah 1 : M lze chápat jako vazbu, kdy jednomu záznamu z tabulky A může odpovídat jeden nebo více (M) záznamů z tabulky B. Poslední možnost, která může nastat, je vztah N: M, kdy M záznamům z tabulky A může odpovídat N záznamům z tabulky B. Tuto vazbu lze zapsat pouze pomoci tabulky C a vznikne vztah 1 : M mezi tabulkou A a C.

2.5.3. Klíče

Kandidátní klíč

Kandidátním klíčem se rozumí takový sloupec tabulky nebo kombinace sloupců, který má unikátní hodnotu v každém záznamu. Musí být nenulový. Jeden z kandidátních klíčů je primárním klíčem.(10)

Primární klíč

Každá tabulka databáze obsahuje takový sloupec, který jednoznačně identifikuje záznam v tabulce. Primární klíč má dvě základní vlastnosti. Jedinečnost a ne-NULL-ovou hodnotu. Primární klíč může být kombinací několika sloupců tabulky. V tomto případě se jedná o složeném klíči. (2)

Cizí klíč

Je-li primární klíč přiřazen do pole jiné tabulky, jedná se o cizí klíč. Tento klíč má stejné vlastnosti a hodnotu jako primární klíč. Pomocí těchto záznamů se vytváří vztahy mezi tabulkami. (10)

2.5.4. Normalizace

„*Normalizování databází je technika, která Vám pomůže vyhnout se datovým anomáliím a dalším problémům se správou dat. Skládá se z transformování tabulky různými fázemi.*

- *1. normalizovaná forma*
- *2. normalizovaná forma*
- *3. normalizovaná firma*
- *Boyce-Coddova normalizovaná forma*
- *4. normalizovaná forma*
- *5. normalizovaná forma "(*10)

1. normalizovaná forma

Tabulka je v první normalizované formě, splňuje-li tato pravidla:

- Nejsou tu žádné opakující se skupiny.
- Jsou definovány všechny klíčové atributy.
- Všechny atributy závisejí na primárním klíči.

Kdyby byly informace zmíněné výše shrnuty, záznam je v 1. normální formě, obsahuje-li jeho atribut již nedělitelnou hodnotu. Příkladem může být atribut „jméno“. První normální formou vzniknou atributy „křestní jméno“ a „příjmení“. Dalším krokem je definování primárního klíče a zajištění, aby zde nebyly žádné opakující se skupiny dat. (5)

2. normalizovaná forma

Tabulka je ve druhé normalizované formě, naplňuje-li tato pravidla:

- Nachází se v 1. normalizované formě.
- Nezahrnuje žádné částečné závislosti (kdy nějaký atribut závisí jen na části primárního klíče).

Druhá normalizovaná forma se týká pouze těch záznamů, kde je primární klíč složen z více než jednoho atributu. Pokud je primární klíč složen pouze z jednoho atributu, záznam je automaticky v druhé normalizované formě. Častým řešením je rozpad na dvě tabulky. (5)

3. normalizovaná forma

Tabulka je ve třetí normalizované formě, naplňuje-li tato pravidla:

- Je-li ve druhé normalizované formě.
- Neobsahuje žádné tranzitivní neboli přechodné závislosti (když nějaký neklíčový atribut závisí na primárním klíči prostřednictvím jiného neklíčového atributu).

Obsahuje-li tabulka pouze jeden neklíčový atribut, je logické, že se automaticky nachází ve třetí normalizované formě. Příkladem závislosti více neklíčových atributů může být „PSČ“ a „Město“. Bude-li „PSČ“ závislé na našem primárním klíči, „Město“ je závislé na PSČ. Tato závislost porušuje třetí normalizovanou formu. Řešením této normalizace by byl rozpad na více tabulek a tyto tabulky propojit pomocí primárních a cizích klíčů. (5)

Boyce-Coddova normalizovaná forma

Platí, že tabulka je v Boyce-Coddově normalizované formě, když naplňuje tyto podmínky:

- Nachází se ve třetí normalizované formě.
- Každý determinant je kandidátním klíčem.

Aby bylo možné blíže pochopit, co je to Boyce-Coddova normalizovaná forma, měl by být objasněn pojem determinant. Determinant je atribut, který určuje (determinuje) hodnotu jiného atributu. Takže pokud atribut určuje jednoznačnost záznamu a není

kandidátním klíčem, je tato forma porušena. Nemělo by být zapomenuto na předchozí normalizované formy, abychom je neporušili. Řešením je rozpad na více tabulek. (5)

4. normalizovaná forma

Tabulka je ve 4. normalizované formě, naplňuje-li tato pravidla:

- Je v Boyce-Coddove normalizované formě.
- Neobsahuje více než jednu vícehodnotovou závislost.

Čtvrtá normální forma se zabývá vztahy uvnitř složeného primárního klíče. Ve složeném primárním klíči se může stát, že hodnoty v klíči jsou na sobě nezávislé. Tím, že tvoří složený primární klíč, vzniká falešná souvislost mezi těmito hodnotami a nemohou existovat nezávisle na sobě. Tímto porušují modelovou realitu. Postup pro dodržení čtvrté normalizované formy je rozdělení do více tabulek.(5)

5. normalizovaná forma

Tabulka se nachází v páté normalizované formě, splňuje-li tyto podmínky:

- Je ve čtvrté normalizované formě.
- Není možné do ní přidat další atribut tak, aby se vlivem skrytých závislostí rozpadla na několik dílčích relací.

Tato normalizovaná forma se nachází spíše v akademické rovině. Problémy řešené touto formou se objevují v praxi málokdy. Tato forma se týká složeného primárního klíče, který se skládá ze tří nebo více atributů. Existuje-li mezi těmito atributy párová cyklická závislost, tak je potřeba tuto závislost extrahovat do samostatných tabulek, ale v některých případech je potřeba zachovat původní tabulku. Mohlo by nastat při rozdělení a opětovném spojení nebo vymazání atributů, že budou ztracena původní data. (5)

Normalizovat je určitě potřeba. Čím složitější databáze a čím více dat, tím více je potřeba normalizovat. Ale i tady platí všeho s mírou. Například u 3. NF by firma s několika desítkami zaměstnanců asi neměla potřebu dávat PSČ do další tabulky a bylo by to zbytečné. V tabulce zákazníků některého z mobilních operátorů s milióny zákazníků to už význam určitě má.

2.6.Jazyk SQL

Co je to SQL? Zkratka SQL (Structured Query Language) by se dala volně přeložit jako „Strukturovaný dotazovací jazyk“. Tento jazyk je používán k dotazování na relační databáze. Na základě tohoto dotazu nám je poskytnuta odpověď. Řadí se mezi tzv. deklarativní jazyky. Pouze definujeme, jaké výsledky si přejeme získat, ale nedefinujeme, jak jich chceme dosáhnout. Příkladem může být funkce AVG. Nemusíme počítat hodnoty a počet těchto hodnot, jazyk SQL tyto operace udělá automaticky. (11)

V rámci standardizace jazyka SQL byly definovány následující skupiny příkazů. Toto členění je pouze zjednodušené a ve skutečnosti je skupin více. Budou zmíněny pouze základní skupiny. (12)

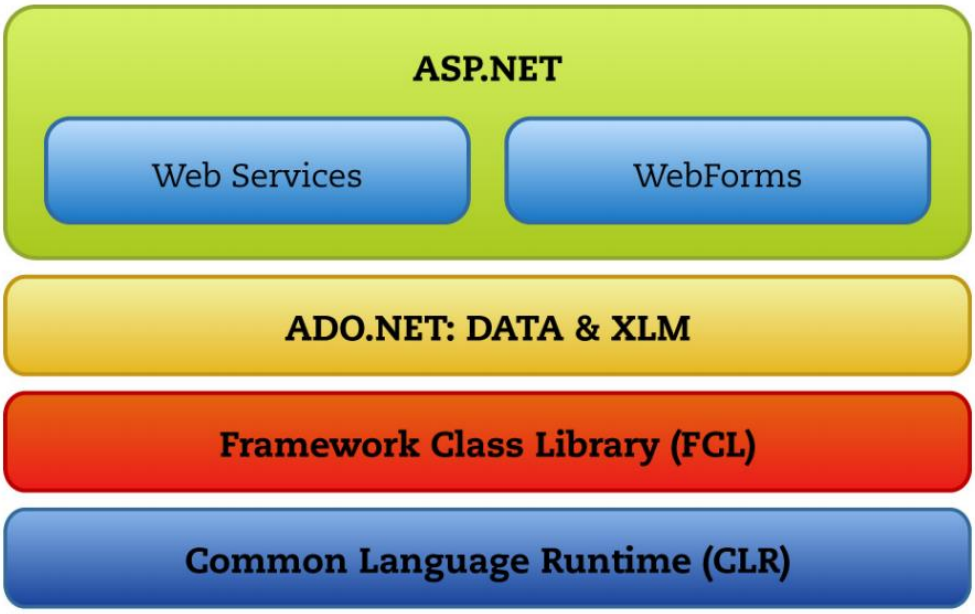
- *Příkazy pro manipulaci s daty (DML, Data Manipulation Language) – pro získání dat z databáze a jejich změny*
- *Příkazy pro definici struktury databáze (DDL, Data Definition Language) – pro definování schématu databáze a její změny*
- *Příkazy pro řízení dat (DCL, Data Control Language) – definování přístupových práv k databázi a datům a také řízení transakcí*
- *Ostatní příkazy – pro definování uživatelů, formulací pravidel, např. způsob řazení dat podle národních abeced apod.* (12, str. 254)

2.7. ASP. NET

Technologie ASP.NET umožňuje vytvořit webové stránky s dynamickým obsahem. Obsah této webové stránky může být uložen v databázi a z této databáze je obsah načítán. Načítání není jediná možnost. Lze vyvinout aplikaci nebo webovou stránku pro tvorbu objednávek a takto do databáze i zapisovat. (13)

Zkratka ASP znamená Active Server Pages (aktivní serverové stránky). ASP dovoluje vkládat do běžných HTML stránek speciální tagy, které umožňují vykonávat funkce, které by s HTML nebyly možné. Tyto funkce jsou vykonávány na serveru a klientovi se zobrazí pouze výsledná HTML stránka. Jedná se o tzv. technologii tenkého klienta. Pod touto funkcí si lze představit např. vyhledávání města dle PSČ. Do formuláře je zadána

hodnota, na serveru proběhne hledání v databázi a je vygenerovaná stránka. Tato stránka se jako HTML stránka odešle klientovi. (14)



Obrázek 3:Schéma architektury ASP. NET *[zdroj:14, str. 10]*

Na obrázku č. 3 je k vidění základní schéma architektury ASP. NET. Obsahuje ASP. NET – Web Services a WebForms, ADO.NET, Framework Class Library (FCL) a Common Language Runtime (CLR).

ADO. NET je zkratka pro ActiveX Data Object. Jedná se o model pro práci s daty. Lze díky němu spojit naši aplikaci nebo webovou stránku s databází. Podporuje jak relační databáze, tak umí pracovat s XML daty.

Framework Class Library (FCL) je knihovna tříd, která umožňuje přistupovat k vlastnostem operačního systému. Lze si jej představit jako strom logicky setříděných funkcí.

Common Language Runtime(CLR) funguje nad operačním systémem. Jedná se o virtuální prostředí, které poskytuje mnoho služeb (např. zavedení a spuštění kódu, správu paměti, zpracování výjimek apod.). (14)

3. ANALÝZA PROBLÉMU A SOUČASNÉ SITUACE

3.1. Základní údaje

Adresa sídla společnosti:

ABB s.r.o.

Štětkova 1638/18

140 00 Praha 4

Tel.: 234 322 110

Obrázek 4: Logo ABB *[zdroj: http://www.abb.com/]*

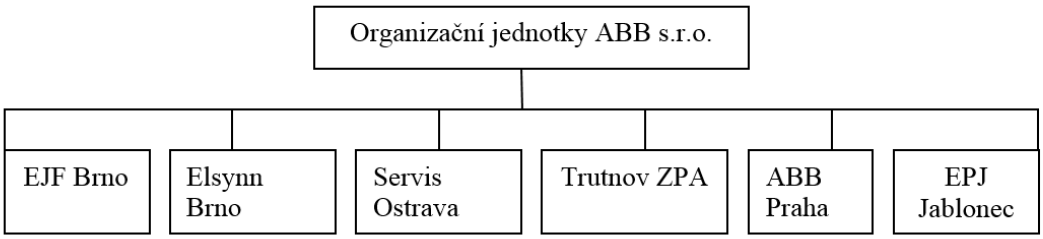
IČO: 49682563

DIČ: CZ49682563

Společnost zapsaná v obchodním rejstříku vedeném Městským soudem v Praze v oddílu C, vložka 79391, se sídlem Štětkova 1638/18, PSČ 140 00 Praha 4, Česká republika

3.2. Organizační struktura podniku

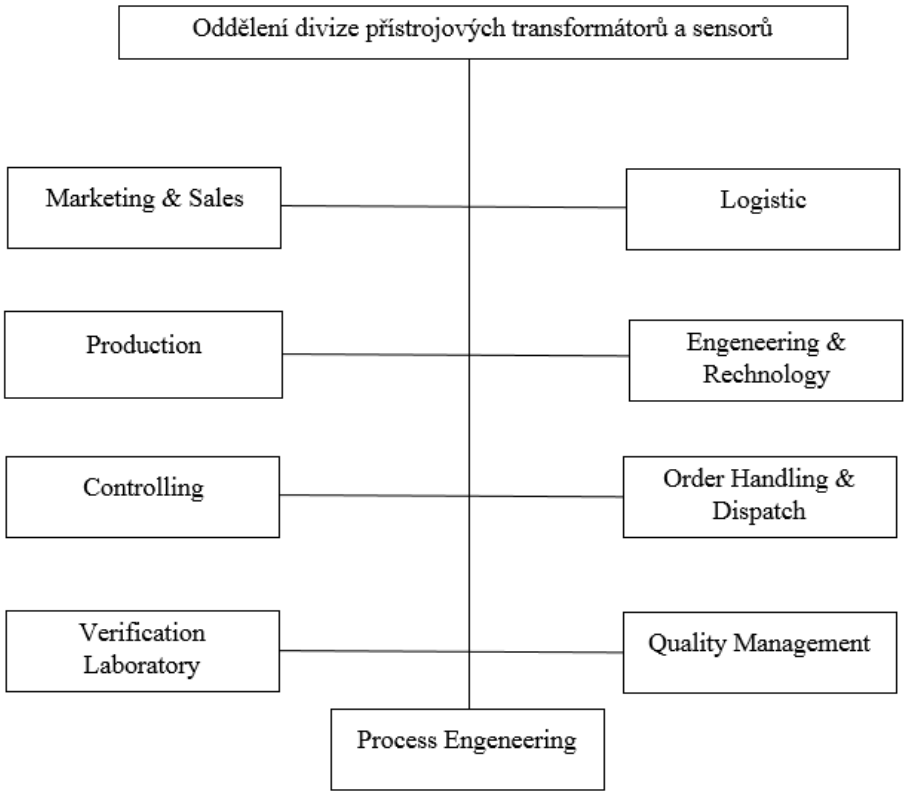
Z pohledu celorepublikového rozdělení se firma ABB nachází v pěti českých městech a je rozdělena na šest samostatně hospodařících společností.



Obrázek 5: Organizační jednotky ABB v ČR *[zdroj: vlastní zpracování]*

Každá organizační jednotka firmy ABB zmíněná výše se liší vyráběným sortimentem. Jednotka EPJ sídlící v Jablonci nad Nisou se zabývá výrobou elektroinstalačního materiálu (např. domovní spínače, zásuvky apod.). Jednotka ABB

Praha vyrábí výkonné polovodičové součástky. Jednotka ZPA Trutnov od roku 2004 dodává rozvaděče a řídicí systémy pro energetiku. Jednotka Servis Ostrava se zabývá opravou elektromotorů, elektromagnetů a suchých transformátorů a také servisem robotů. Jednotka Elsynn Brno dodává prostředky pro rozvod elektrické energie s nízkým napětím a elektrické pohony. Poslední organizační jednotkou je EJF Brno. Zabývá se výrobou rozvaděčů středního a vysokého napětí, přístrojových transformátorů a sensorů. Tuto jednotku tedy lze ještě rozdělit na Divizi přístrojových transformátorů a senzorů a Divizi rozvaděčů. A právě Divizí přístrojových transformátorů a sensorů se budeme zabývat ještě podrobněji. Organizační struktura znázorněna níže.



Obrázek 6: Oddělení divize přístrojových transformátorů a sensorů *[zdroj: vlastní zpracování]*

3.3.Historie EJF Brno

Společnost začíná svou historii již v roce 1887, ve kterém vznikl první elektrotechnický závod na Moravě pod názvem Bartelmus – Donát. V roce 1927 v období blížící se hospodářské krize byl tento elektrotechnický závod zakoupen společností ŠKODA a před druhou světovou válkou zde byl vyráběn široký sortiment elektrických přístrojů

nízkého i vysokého napětí včetně měřících transformátorů a rozvaděčů. V roce 1950 se v rámci znárodnění průmyslu závod osamostatnil a byl přejmenován na Elektrotechnické závody Julia Fučíka – EJF Brno. O devět let později byl podnik začleněn do sdružení elektrotechnických podniků s názvem Závody silnoproudé elektrotechniky Praha (ZSE Praha).

V roce 1990 se závod EJF Brno znovu osamostatnil a roku 1993 bylo v rámci privatizace českého průmyslu 100 procent akcií zakoupeno nadnárodní společností ABB (Asea Brown Boveri), jejíž sídlo je v Curychu ve Švýcarsku. Závod byl přejmenován na ABB EJF, a.s.. V roce 1998 byly ze společnosti vyčleněny divize zabývající se výrobou nízkonapěťových přístrojů a zařízení a podnik se tak stal výrobcem výhradně silnoproudé elektrotechniky v oblasti vysokého napětí.

V prosinci roku 2000 došlo ke sjednocení samostatných společností v rámci ABB a vznikla tak jedna velká společnost ABB s.r.o., skládající se z několika organizačních jednotek. Společnost se přeměnila z akciové společnosti na společnost s ručením omezeným, neboť chtěla zvýšit svoji flexibilitu vůči předpokládaným změnám v její struktuře. V průběhu uplynulých čtyř let totiž společnost provázely výrazné změny, při kterých docházelo k odprodávání jednotlivých organizačních jednotek nebo jejich částí (např. divize prvovýroby ABB EJF v Brně, kterou společnost odprodala a v současnosti od ní nakupuje komponenty v rámci outsourcingu). (15)

3.4. Současný stav

V současné době neexistuje podobný nástroj, který by usnadňoval vyhledávání souborů na síťovém disku. Pro představu, síťový disk obsahuje data o velikosti 310 GB, tj. 21 000 složek a 250 000 souborů.

3.5. Informační technologie

3.5.1. Pracovní stanice

Pracovní stanice uživatelů, ať už se jedná o stolní počítače nebo notebooky, jsou výhradně od výrobce Lenovo. U stolních počítačů se jedná o počítače Lenovo ThinkCenter řady M. Jedná se o řadu firemních počítačů, které nabízejí spolehlivý chod a dostatečný výkon. Výhodou této řady je provedení Small Form Factor, které je vhodné

do malých prostor a umožňuje orientaci na výšku nebo na šířku. Další výhodou je možnost připojení až čtyř monitorů. Tato funkce usnadňuje organizaci pracovní plochy a zvyšuje produktivitu práce. Konkrétní technická specifikace hardwaru jednotlivých pracovních stanic se může lišit kus od kusu. Existuje mnoho variant jak stolní počítač sestavit, může se lišit v operační paměti, velikosti pevného disku nebo zda obsahuje čtečku paměťových karet. Naopak každá pracovní stanice obsahuje DVD mechaniku a obsahuje operační systém Windows 7 Professional. Příslušenství ke stolním počítačům, jako jsou monitory a vstupní zařízení, již nejsou výhradně od výrobce Lenovo, ale stále převažují. Lze objevit monitory různých rozměrů i od jiných výrobců. Nejrozšířenějším monitorem je však monitor Lenovo ThinkVision LT2223p. Jedná se o monitor o uhlopříčce 21,5“ s poměrem stran 16:9. Propojení s pracovní stanicí je možno pomocí VGA vstupu, HDMI vstupu nebo pomocí DisplayPortu. Navíc obsahuje ještě USB port. Dalším nejrozšířenějším příslušenstvím jsou polohovací zařízení od výrobce Lenovo. Jedná se o klasické laserové myši připojené pomocí USB. Tyto myši mají rozlišení 2000 DPI. Dalším příslušenstvím je klávesnice s USB portem s českou lokalizací.

Obrázek 7: Informační technologie *[zdroj:http://www.lenovo.com/cz/cs/]*

Co se týče notebooku, je situace podobná jako u stolních počítačů. Konfigurace jednotlivých zařízení se liší, jak v hardware který obsahují, tak ve velikosti monitorů. Toto je většinou závislé na datu pořízení notebooku. Součástí každého notebooku je dokovací stanice, která umožní z notebooku během pár vteřin udělat stolní počítač, aniž bychom museli zdlouhavě připojovat všechny monitory a polohovací zařízení. A naopak během pár vteřin být opět mobilní v případě potřeby. Příslušenství je stejné jako u stolních pracovních stanic.

Obrázek 8: Dokovací stanice Lenovo[zdroj:http://www.lenovo.com/cz/cs/]

Na všech pracovních stanicích a noteboocích je naistalován operační systém společnosti Microsoft, Windows 7 Professional. Co se týče aplikačního vybavení jednotlivých stanic, obsahují standartní aplikace schválené firmou ABB, tj. WinZip, antivirový software McAfee, Adobe Reader, Corel, Bussines Explorer, ABB Remote Control a jiné. Odlišnosti se nacházejí poté v jednotlivých odděleních a u různých uživatelů podle jejich pracovního zaměření. Jsou to například programy pro vývojáře a programátory SQL Managment Studio,Visual Studio, Report Builder atd. Pro inženýry je to např. SolidWorks.

Pro komunikaci mezi kolegy se ve firmě, v době mého nástupu, používala aplikace od IBM Lotus Notes, ale firma celosvětově přechází na aplikaci Lync, která je součástí Office 365. V současné době běží tyto dva komunikační nástroje souběžně, avšak podpora Lotus Notes by měla být do konce roku ukončena.

V rámci bezpečnosti není uživatelům povoleno instalovat aplikace sami. Musí zadat požadavek na IT oddělení pomocí příslušného nástroje, a pokud je požadavek schválen a aplikace není v rozporu s pravidly ABB, je tato aplikace naistalována IT oddělením.

3.5.2. Síťové technologie

Firma ABB disponuje sofistikovanými síťovými technologiemi, které podporují chod informačních systémů. Veškeré tyto technologie jsou pod správou IT oddělení. Konkrétní informace nejsou k dispozici. Jedná se však o virtuální servery a tak je těžké konkrétní hardwarové specifikace jednotlivých serverů určit. O chod se stará operační systém Windows Server 2012.

3.6. Informační systémy

SAP

SAP je softwarový produkt, který slouží pro řízení podniku (Enterprise resources planning – ERP). ERP je informační systém, který integruje a automatizuje velké množství procesů souvisejících s činnostmi podniku. Typicky se jedná o výrobu, logistiku, distribuci, správu majetku, prodej, fakturaci a účetnictví.

SAP je vnitřně rozčleněn do jednotlivých celků, takzvaných modulů, které se kryjí s logickým členěním typické firmy. SAP se skládá z následujících modulů:

- FI (Financial Accounting) Finanční účetnictví
- CO (Controlling) Kontrola
- AM (Asset Management) Evidence majetku
- PS (Project system) Plánování dlouhodobých projektů
- WF (Workflow) Řízení oběhu dokumentů
- IS (Industry Solutions) Specifická řešení různých odvětví
- HR (Human Resources) Řízení lidských zdrojů
- PM (Plant Maintenance) Údržba (16)

V Brně se nachází tzv. sdílené účtárny (share services) používající systém SAP, ve kterých se shromažďují a sjednocují účetní výkazy jednotlivých organizačních jednotek. Vyhodnocuje výsledky jednotlivých organizačních jednotek v různých oblastech (výroba, distribuce, prodej apod.) a výsledky ABB s.r.o. jako celku.

CpmPlus Enterprise Connectivity (MES)

Divize přístrojových transformátorů a senzorů disponuje systémem MES (Manufacturng Execution Systems). Tato zkratka by se dala přeložit jako výrobní informační systémy. Tento systém byl naprogramován konkrétně pro ABB IT&S a jmenuje se cpmPlus Enterprise Connectivity. Je propojen s ERP systémem SAP. Tento systém pracuje na pracovních stanicích umístěných na pracovištích, podle logické segmentace procesu výroby.

Klíčové oblasti systému cpmPlus Enterprise Connectivity:

- Dostupnost výrobních dat v reálném čase
- Aktualizace výrobních dat v reálném čase, což vede k přesnějšímu plánování
- Stálá aktualizace výrobního statusu až do ukončení výroby
- Dostupnost potřebných informací (instrukce, seznam materiálu, oznámení)
- Snadno přístupná všechna kritická data

Hlavní funkce systému cpmPlus Enterprise Connectivity

- Propojení (smazání, přidání, úprava) výrobních a prodejních zakázek se systémem SAP, včetně důležitých informací (sériové číslo, list materiálu apod.)
- Aktualizace priorit výrobních zakázek v systému MES na základě aktualizace vyrovnávací paměti systému SAP
- Aktualizace výrobních postupů
- Kontrola spuštění, ukončení, pozastavení a stornování kroků výroby s předdefinovanými důvody a komentáři
- Zachování historických dat (číslo šarže, sériová čísla vstupního materiálu apod.)
- Podpora zasílání pracovních instrukcí v reálném čase elektronicky (lepší komunikace mezi modrými a bílými límečky – obousměrná komunikace)
- Integrace RFID zařízení, čteček čárových kódů a tiskáren čárových kódů
- Integrace výrobních strojů a databází na konkrétní výrobní kroky
- Sledování postupů výroby v reálném čase
- Měření výkonnosti
- Měření skutečné spotřeby materiálu
- Kompletní report o produktivitě osoby (17)

Service Manager (HPSM)

Jedná se o tiketový nástroj pro hlášení požadavků na IT oddělení. HPSM je náhradou tiketového nástroje za Lotus Notes aplikace Service Desk. Tento nástroj je dostupný přes web portál. Do tohoto systému funguje automatické přihlášení SSO (Single Sign On), nebo je zde i možnost manuálního přihlášení (Manual login). Zakládají se v něm požadavky na IT: ISA (podpora SAP), ISI (IT infrastruktura).

Service Manager lze rozdělit z dvou pohledů:

ESS - Employee Self Service

- zde mají přístup všichni uživatelé informačních technologií
- možnosti:
    - založení tiketu
    - schválení změnového požadavku (klíčový uživatel, vlastník procesu, IS manažer)

SMC - Service Management Centre

- zaměstnanci IT oddělení
- možnosti:
    - řešení incidentu
    - změny
    - přesměrovaní
    - žádost o podrobnější informace

4. VLASTNÍ NÁVRH ŘEŠENÍ INFORMAČNÍHO SYSTÉMU

4.1. Požadavky na systém

Požadavky na systém byly definovány vedoucími jednotlivých oddělení. Toto je logický krok, protože je důležité, aby systém splnil představu samotných uživatelů. Z několika porad, které byly vedeny formou diskuze a návrhů, vzešla obecná definice informačního systému.

Z technického hlediska bylo zvoleno řešení webové stránky na inside portálu firmy ABB. Tato webová stránka je přístupná pouze v pracovní síti ABB. Jelikož internetový prohlížeč je standartním nástrojem každého uživatele, je systém dostupný všem bez potřeby instalace nového softwaru. Kvůli snadné správě a editaci obsahu systému bylo zvoleno databázové řešení. Tyto úpravy lze provádět v databázi, kterou spravuje naše oddělení a na kterou je informační systém napojen. Naopak zbytek systému poběží na serverech, které spadají pod oddělení IT. Na tyto servery nelze přistupovat a veškerá správa nebo aktualizace obsahu by znamenala další administraci v podobě zadání ServiceDesku s požadavkem o aktualizaci. V případě každodenní aktualizace obsahu informačního systému by to nebylo vhodné řešení.

Na samém začátku měl být informační systém tvořen pouze modulem odkazů. Z grafického hlediska nebyl žádný prostor pro úpravu, jelikož systém musel odpovídat vnitřním nařízením firmy ABB. Vzhled systému byl tedy již předem definován. Jako nejdůležitější vlastnost informačního systému je přehlednost a logické uspořádání zobrazovaných odkazů. Aby byla tato vlastnost docílena, jsou zobrazované odkazy kategorizovány z dvou různých hledisek. Prvním hlediskem bylo zvoleno kategorizování pomocí oddělení. Toto rozdělení by mělo sloužit ke snadnějšímu sdílení informací mezi kolegy jednoho oddělení. Bylo definováno 9 oddělení a to tyto: Marketing & Sales, Engineering Technology, Order Handling, Process Engineering, Supply Chain & Logistics, Quality Management, Production, Director & Controling, Verification Laboratory. Druhým pohledem je rozdělení z hlediska kategorií. Toto by mělo usnadnit sdílení odkazů na úrovni celé divize IT&S. Bylo definováno 7 základních kategorií.

Management & Finance, Sales & Marketing, Order Processing, Technical Area, Supply Chain, Quality Management, Operation.

Pro přehlednost systému byly definovány ještě další vlastnosti, které obsahuje každý odkaz. Jedná se o vlastnost „ToolTip“. Tento krátký text se zobrazí po najetí myši na odkaz. Tato vlastnost pomáhá uživatelům ověřit, zda se opravdu jedná o odkaz, který hledají a chtějí otevřít. V případě hledání může být tento text nápomocný. Druhou vlastností je „Ikona“. U každého odkazu se zobrazí malá ikona formátu souboru, na který odkaz vede. Byly definovány tyto formáty: Excel, Word, PowerPoint, Web, Aplikace, LotusNote, OneDrive, Složka a Jiné. Díky této vlastnosti uživatel ihned uvidí, na jaký typ souboru bude přesměrován.

Dalším požadavkem, který vzešel ze společné diskuze, byla možnost sdílení a ukládání elektronických verzí papírových smluv, které mají u sebe jednotliví uživatelé. Uživatel by vyplnil veškeré povinné náležitosti smlouvy, vybral by např. naskenovanou verzi smlouvy ve formátu pdf ve svém počítači a uložil by ji do systému. Systém by uložil veškeré údaje do databáze, vybraný soubor by nakopíroval na server a uložil si cestu k němu. Seznam všech uložených smluv by se uživatelům zobrazoval s možností náhledu elektronické verze smlouvy. Samozřejmostí je možnost vyhledávání v již uložených smlouvách. Jako povinné údaje a údaje, podle kterých lze vyhledávat byly definovány Číslo smlouvy, Předmět smlouvy, Datum podpisu a Jméno zaměstnance u kterého je smlouva fyzicky uložena, popř. kdo ji uzavřel. Z těchto definic vznikl modul Smlouvy.

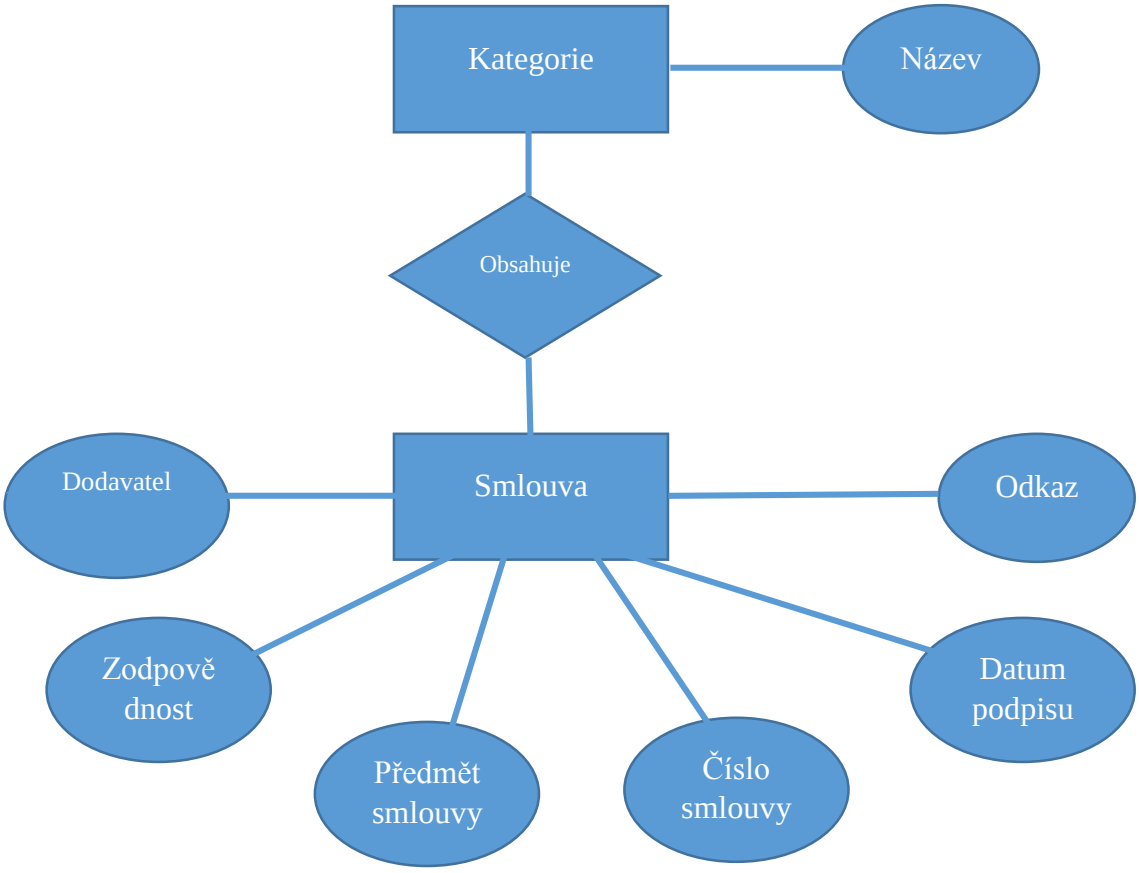
Dalším požadavkem, ne tak důležitým z pohledu uživatele, ale ne méně důležitým z pohledu funkčnosti systému, bylo uchovávání výjimek, které nastaly při běhu systému.

4.2. Databázové řešení

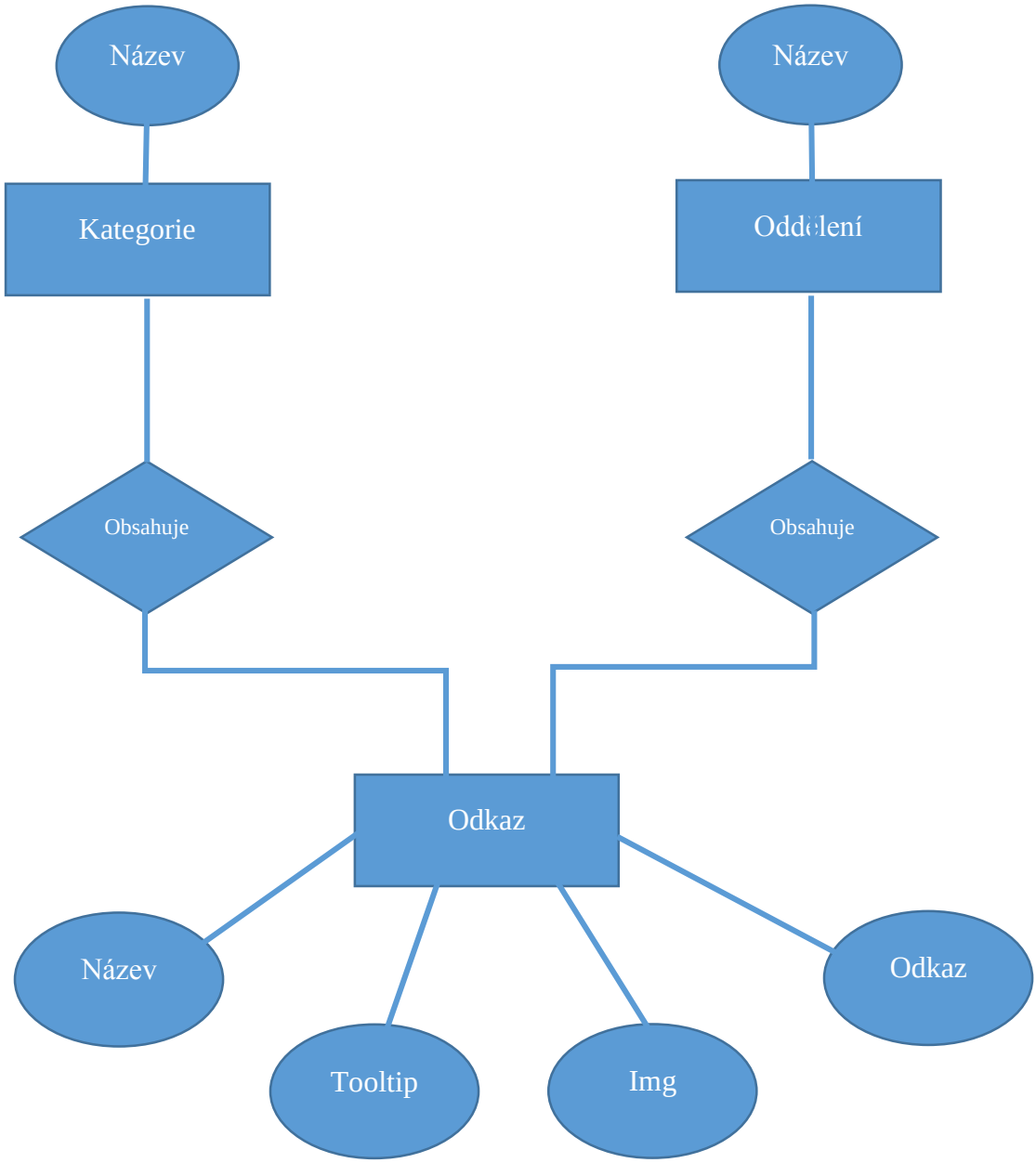
Nejdůležitější částí navrhovaného informačního systému je databáze a tak je nutno zachytit všechny entity a hodnoty, které budeme v databázi uchovávat. Poté zvolit vhodný databázový model a nakonec i databázi vytvořit.

4.2.1. Konceptuální návrh databáze

Pomocí grafického návrhu jsou níže zachyceny entity, které vystihují všechny objekty reality a znázorňují vazby mezi nimi. V grafickém návrhu jsou zachyceny také vlastnosti jednotlivých objektů, které budeme uchovávat a budou nám tak vlastně tvořit záznamy v databázi.



Obrázek 9: grafický návrh databáze I. *[zdroj: vlastní zpracování]*



Obrázek 10: Grafický návrh databáze II. *[zdroj: vlastní zpracování]*

4.2.2. Logický návrh databáze

V logickém návrhu databáze identifikujeme všechny entity a relace a charakterizujeme jejich vlastnosti. Budeme vycházet z našeho grafického návrhu v konceptuálním návrhu databáze.

Entita Odkaz

První ze dvou nejdůležitějších entit celého systému je entita Odkaz. Tato entita slouží k evidování té nejdůležitější součásti systému, tj. odkazu. Obsahuje atribut „ID_Odkaz“ Tento atribut je primárním klíčem a určuje jeho jednoznačnou identifikaci. Dalším atributem je „Nazev“. Tento atribut obsahuje text, který odkaz pojmenovává. Tento atribut musí byt vyplněn, nesmí obsahovat hodnotu NULL. Dalším atributem je „Tooltip“. Tento atribut nemusí být vyplněn. Atribut „Img“ obsahuje cestu k ikoně odkazu, podle typu soboru, na který odkazuje. Atribut „Odkaz“ obsahuje absolutní cestu. Tento atribut nesmí obsahovat hodnotu NULL.

Tabulka 1: Entita Odkaz *[zdroj: vlastní zpracování]*

Odkaz			
Atribut	**Popis**	**Datový typ**	**Null**
ID_Odkaz	Unikátní hodnota určující jednoznačně odkaz	Int	Ne
Nazev	Název odkazu, který je zobrazován uživatelům	Varchar(200)	Ne
Tooltip	Text, který je zobrazován jako tooltip u zobrazených odkazů	Varchar(400)	Ano
Img	Cesta, která určuje zobrazenou ikonu u záznamu	Varchar(100)	Ne
Odkaz	Cílová cesta, na kterou daný záznam odkazuje	Varchar(400)	Ne

Entita Kategorie_odkaz

Každá entita Odkaz patří do jedné nebo více entit Kategorie. Entita Kategorie obsahuje atribut „ID_Kategorie“. Je primárním klíčem a jednoznačně identifikuje záznam. Atribut „Nazev“ určuje název kategorie. Oba tyto atributy musí být vyplněny a nesmí obsahovat hodnotu NULL.

Tabulka 2:Entita Kategorie_odkaz *[zdroj: vlastní zpracování]*

Kategorie_odkaz			
Atribut	**Popis**	**Datový typ**	**Null**
ID_Kategorie	Unikátní hodnota určující jednoznačně kategorii odkazu	Int	Ne
Nazev	Název kategorie odkazu	Varchar(50)	Ne

Entita Oddeleni_odkaz

Každá entita Odkaz patří do jedné nebo více entit Oddeleni. Entita Oddeleni obsahuje atribut „ID_Oddeleni“. Je primárním klíčem a jednoznačně identifikuje záznam. Atribut „Nazev“ určuje název oddělení. Oba tyto atributy musí být vyplněny a nesmí obsahovat hodnotu NULL.

Tabulka 3: Entita Oddeleni_odkaz [zdroj: vlastní zpracování]

Oddeleni_odkaz			
Atribut	**Popis**	**Datový typ**	**Null**
ID_Oddeleni	Unikátní hodnota určující jednoznačně oddělení odkazu	Int	Ne
Nazev	Název oddělení odkazu	Varchar(50)	Ne

Entita Smlouva

Druhou nejdůležitější entitou systému je entita Smlouva. Obsahuje atribut „Dodavatel“. Identifikuje protistranu s kterou je smlouva uzavřena. Nesmí obsahovat hodnotu NULL. Atribut „Cislo_Smlouvy“ jednoznačně identifikuje smlouvu, ale je nevhodným atributem pro primární klíč, protože může obsahovat dlouhé texty. Musí být vyplněn a nesmí obsahovat hodnotu NULL. Atribut „Předmět_Smlouvy“ jako krátký text popisuje, čeho se smlouva týká a nesmí obsahovat hodnotu NULL. Atribut „Datum_Podpis“ zaznamenává, kdy byla smlouva podepsána. Nesmí obsahovat hodnotu NULL. Atribut „Zodpovednost“ uchovává jméno, u koho je smlouva uložena a kdo za ni nese zodpovědnost. Nesmí obsahovat hodnotu NULL. Posledním atributem, který musí být vyplněn, je atribut „Odkaz“. Uchovává cestu k elektronické formě podepsané smlouvy. Poslední atributy „Uzivatel“ a „Datum“ nemusí být vyplněny a jsou spíše informativní. Uchovávají login uživatele, který smlouvu do systému uložil a datum, kdy se tak stalo.

Tabulka 4: Entita Smlouva [zdroj: vlastní zpracování]

Smlouva			
Atribut	**Popis**	**Datový typ**	**Null**
ID_Smlouva	Unikátní hodnota určující jednoznačnost záznamu	Int	Ne
Dodavatel	Název druhé strany uzavírané smlouvy	Varchcar(150)	Ne
Cislo_Smlouvy	Identifikační číslo smlouvy	Varchar(150)	Ne

Předmět_Smlouvy	Název oblasti, které se smlouva týká	Varchar(150)	Ne
Datum_Podpis	Datum podpisu smlouvy	Datetima	Ne
Zodpovednost	Jméno osoby, u které je smlouva uložena a nese za ní odpovědnost	Varchar(50)	Ne
Uzivatel	Login uživatele, který nahrál záznam do systému	Varchar(15)	Ano
Datum	Datum nahrání smlouvy do systému	Datetime	Ano
Odkaz	Cesta k PDF souboru, kde je smlouva elektronicky uložena	Varchar(300)	Ne

Entita Kategorie

Každá entita Smlouvy patří do jedné entity Kategorie. Entita Kategorie obsahuje atribut „ID_Kategorie“. Je primárním klíčem a jednoznačně identifikuje záznam. Atribut „Nazev“ určuje název kategorie. Oba tyto atributy musí být vyplněny a nesmí obsahovat hodnotu NULL. Další dva atributy „Uzivatel“ a „Datum“ nemusí byt vyplněny.

Tabulka 5: Entita Kategorie *[zdroj: vlastní zpracování]*

Kategorie			
Atribut	**Popis**	**Datový typ**	**Null**
ID_Kategorie	Unikátní hodnota určující jednoznačně kategorii	Int	Ne
Nazev	Název kategorie	Varchar(50)	Ne
Uzivatel	Login uživatele, který tuto kategorii založil	Varchar(20)	Ano
Datum	Datum, kdy byla kategorie založena	Datetime	Ano

Entita Zadost_odkaz

Tato entita obsahuje veškeré atributy jako entita Odkaz. Slouží k evidenci žádostí o přidání nových odkazů do systému.

Tabulka 6: Entita Zadost_odkaz [zdroj: vlastní zpracování]

Zadost_odkaz			
Atribut	**Popis**	**Datový typ**	**Null**
ID_Zadost	Unikátní hodnota určující jednoznačnost záznamu	Int	Ne
Nazev	Název odkazu, pod kterým má byt do systému uložen	Varchar(150)	Ne
Tooltip	Tooltip, který má být u odkazu zobrazován	Varchar(300)	Ne
Cesta	Cesta k odkazu	Varchar(300)	Ne
Kategorie	Kategorie, do které má být odkaz zaražen	Varchar(50)	Ne
Oddeleni	Odděleni, do kterého má být odkaz zařazen	Varchar(50)	Ne
Datum	Datum, kdy byla žádost do systému přidána	Datetime	Ano
Uzivatel	Login uživatelžá který zadost přidal	Varchar(15)	Ano

Entita Error

Entita Error obsahuje 3 atributy. První atribut se jmenuje „ID_Error“. Tento atribut je primárním klíčem a jednoznačně identifikuje záznam. Dalším atributem je „Error_text“. Pokud ve funkční části systému nastane neočekávatelná chyba, je kód a popis této chyby uložen do databáze. Posledním atributem je „Datum“. Uchovává datum vložení záznamu do databáze.

Tabulka 7: Entita Error [zdroj: vlastní zpracování]

Error			
Atribut	**Popis**	**Datový typ**	**Null**
ID_Error	Unikátní hodnota určující jednoznačnost záznamu	Int	Ne
Error_text	Text error message, když v systému chyba nastane	Text	Ne
Datum	Datum, kdy byl záznam uložen	Datetime	Ano

Relace Smlouva – Kategorie

Vztah mezi entitami Smlouva a Kategorie můžeme nazvat rozdělení. Entita Smlouva vždy musí patřit do jedné kategorie. Naopak entita Kategorie může obsahovat N smluv. Tento vztah je znázorněn v tabulce níže.

Tabulka 8:Relace Smlouva - Kategorie [zdroj: vlastní zpracování]

Název vztahu	Entita	Vztah		Entita
Rozdělení	Smlouva	je součástí	1..1	Kategorie
	Kategorie	obsahuje	1..N	Smlouva

Relace Odkaz – Kategorie

Vztah mezi entitami Odkaz a Kategorie můžeme nazvat Odkaz-Kategorie. Entita Odkaz může patřit do N kategorií. Entita Kategorie obsahuje N odkazu. Tato relace je znázorněna v tabulce níže.

Tabulka 9: Relace Odkaz - Kategorie [zdroj: vlastní zpracování]

Název vztahu	Entita	Vztah		Entita
Odkaz-Kategorie	Odkaz	patří	1..N	Kategorie
	Kategorie	obsahuje	1..N	Odkaz

Relace Odkaz – Oddeleni

Vztah mezi entitami Odkaz a Oddeleni můžeme nazvat Odkaz-Oddeleni. Entita Odkaz může patřit do N oddělení. Entita Oddeleni obsahuje N odkazu. Tato relace je znázorněna v tabulce níže.

Tabulka 10: Relace Odkaz - Oddeleni [zdroj: vlastní zpracování]

Název vztahu	Entita	Vztah		Entita
Odkaz-Oddeleni	Odkaz	patří	1..N	Oddeleni
	Oddeleni	obsahuje	1..N	Odkaz

4.2.3. Fyzický návrh databáze

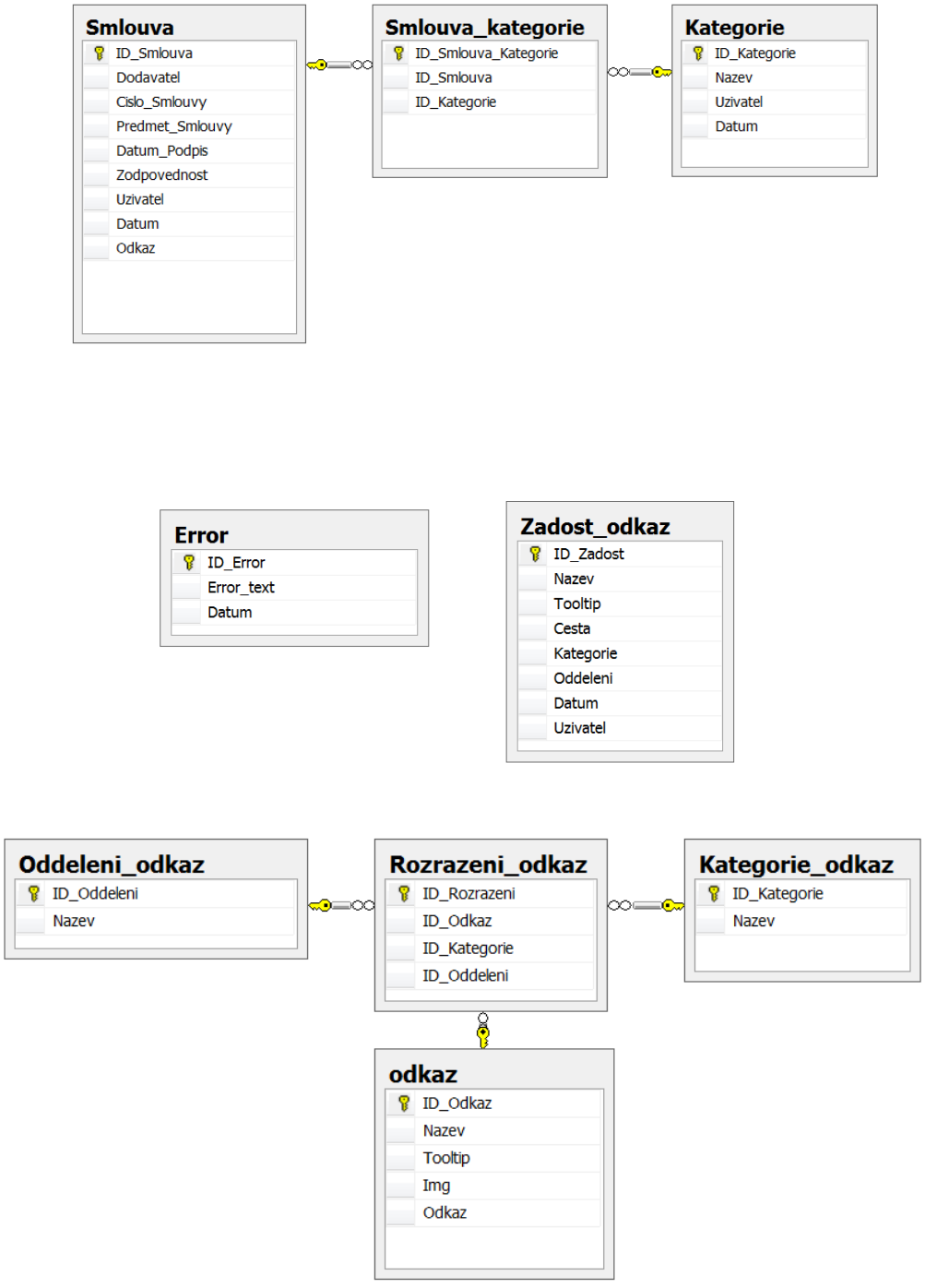
Pomoci nástroje SQL Server Management Studio a jazyka SQL byla fyzicky vytvořena naše databáze. Tato databáze vycházela ze zpracovaného logického návrhu. SQL Server Management Studio nám nabízí více možností, jak databázi vytvořit. Lze ji vytvořit pomoci grafického rozhraní, které tento nástroj nabízí. Tato volba nabízí možnost tvorby databáze i pro méně znalé jazyka SQL. Druhá možnost je programováním. Databáze byla vytvořena druhým způsobem. Zápisem jazyka SQL a následným zpracováním SQL serverem. Na níže znázorněném příkladu vysvětlím použité příkazy jazyka SQL.

```
create table Smlouva_kategorie
(
ID_Smlouva_Kategorie int identity not null,
ID_Smlouva int not null,
ID_Kategorie int not null,

constraint PK_Smlouva_Kategorie primary key(ID_Smlouva_Kategorie),
constraint FK_Smlouva foreign key (ID_Smlouva) references Smlouva
(ID_Smlouva),
constraint FK_Kategorie foreign key (ID_Kategorie) references
Kategorie (ID_Kategorie)
)
```

Příkaz „create table“ vytvoří tabulku. V našem případě tabulku s názvem „Smlouva_kategorie“. Následující tři řádky tvoří atributy tabulky. Na začátku se nachází název atributu, následuje datový typ, v našem případě integer. Atribut „identity“ znamená automatickou inkrementaci primárního klíče. Parametr „not null“ značí, že hodnota atributu nesmí být nulová, ale musí obsahovat hodnotu. Následující řádky nám určují primární a cizí klíče. Parametr „constraint“ pojmenovává omezení. S těmito omezeními lze pracovat stejně jako s objekty. V našem případě pojícího se s primárním klíčem. V závorce je pojmenování atributu, který je primárním klíčem. Následující řádky naopak určují cizí klíč. V první závorce je název atributu, který bude cizím klíčem. Parametr „references“ odkazuje na tabulku a atribut v ní, který se bude s naším cizím klíčem svázán.

Toto je pouze vytvoření jedné tabulky. Celý SQL kód, pro vytvoření celé databáze je k nalezení v příloze č. 1. SQL Server Management Studio nabízí mnoho funkcí a jednou z ní je generování diagramu. Po fyzickém vytvoření databáze byl jeden takový diagram vygenerován a můžeme ho vidět níže. V diagramu lze vidět vazby mezi tabulkami, jejich primární klíče apod.



Obrázek 11: Schéma databáze *[zdroj: vlastní zpracování]*

4.2.3.1. Procedury

Při fyzickém návrhu databáze byly navrhnuty procedury, které slouží k zobrazování dat. V následující části je každá procedura popsána. SQL kód se nachází v příloze č. 1.

Procedura Kategorie_smlouva

Tato procedura má jeden vstupní parametr s názvem ID_kategorie datového typu integer. Tato procedura podle vstupního parametru, kterým je primární klíč kategorie smlouvy, vrátí záznam (entitu) smlouvy, která do dané kategorie patří. Tato procedura je použita k výpisu smluv dle kategorií.

Procedura Listbox

Tato procedura neobsahuje žádný vstupní parametr. Vrací nám pouze „ID“ a „Název“ každého záznamu z tabulky s názvem Kategorie. Tato procedura nám slouží k naplnění listboxu názvy všech kategorií smluv.

Procedura Listbox_kategorieodkazu

Tato procedura neobsahuje žádný vstupní parametr. Vrací nám pouze „ID“ a „Název“ každého záznamu z tabulky s názvem Kategorie_odkaz. Tato procedura nám slouží k naplnění listboxu názvy všech kategorií, do kterých může odkaz patřit.

Procedura Listbox_oddeleniodkazu

Tato procedura neobsahuje žádný vstupní parametr. Vrací nám pouze „ID“ a „Název“ každého záznamu z tabulky s názvem Oddělení_odkaz. Tato procedura nám slouží k naplnění listboxu názvy všech oddělení, do kterých může odkaz patřit.

Procedura odkaz_bykategorie

Tato procedura má jeden vstupní parametr s názvem ID_kategorie datového typu integer. Tato procedura podle vstupního parametru, kterým je primární klíč kategorie odkazu, vrátí záznam (entitu) odkaz, která do dané kategorie patří. Tato procedura je použita k výpisu odkazů dle kategorií.

Procedura odkaz_byoddeleni

Tato procedura má jeden vstupní parametr s názvem ID_oddeleni datového typu integer. Tato procedura podle vstupního parametru, kterým je primární klíč oddělení

odkazu, vrátí záznam (entitu) odkaz, která do daného oddělení patří. Tato procedura je použita k výpisu odkazů dle oddělení.

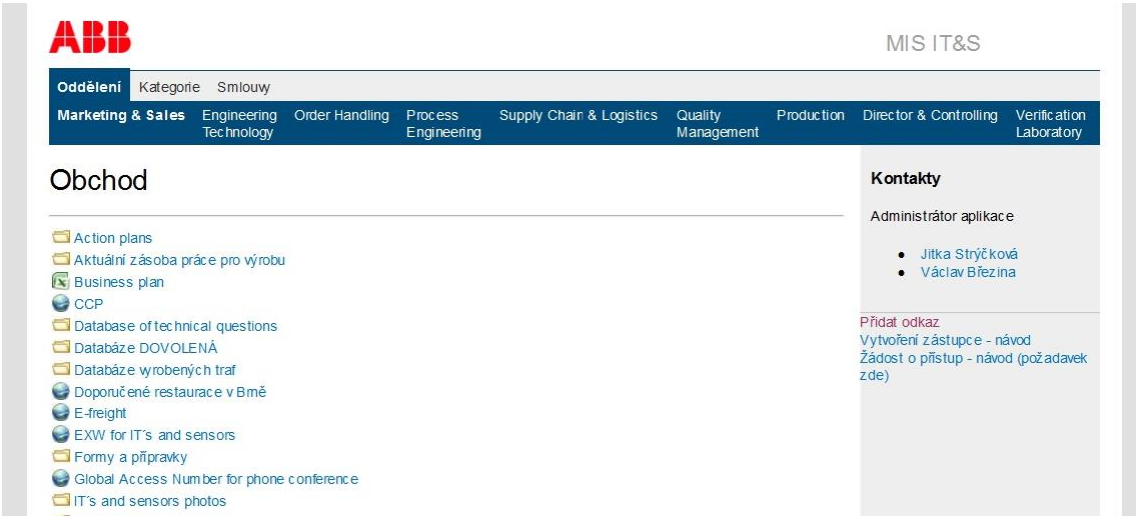
Procedura smlouva_nacti

Tato procedura má jeden vstupní parametr s názvem ID_smlouva datového typu integer. Tato procedura podle vstupního parametru, kterým je primární klíč smlouvy, vrátí záznam (entitu) smlouva podle ID. Tato procedura je použita k výpisu smluv.

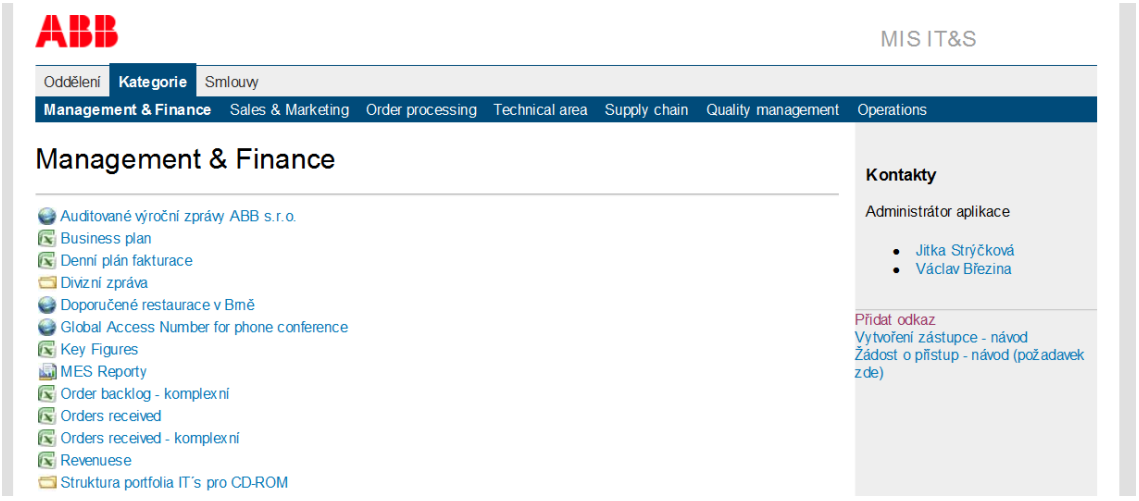
4.3.Řešení uživatelského rozhraní

Pouhá databáze je pouze půlka našeho informačního systému. Data v ní musíme také uživatelům zobrazit. Jak již bylo zmíněné výše, data budou zobrazena na webové stránce. Použijeme technologii ASP. NET, která je vhodná pro zobrazování proměnlivého obsahu webové stránky. V našem případě uložených dat v databázi. Vzhled webové stránky a rozložení jednotlivých prvků jsou znázorněny níže.

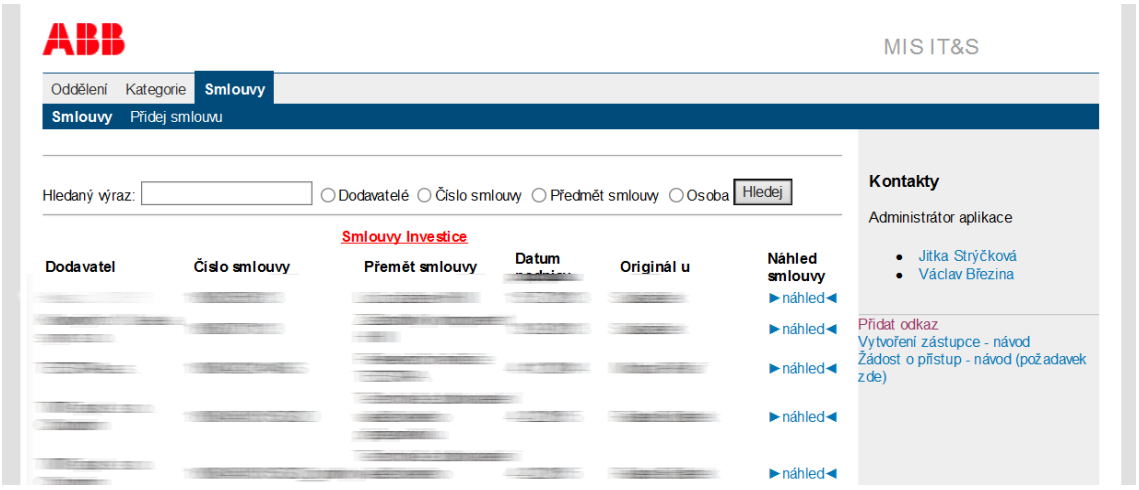
Ve vrchní části se nachází logo firmy ABB a název aplikace. Pod tímto záhlavím stránky se nachází první ze dvou menu. Toto menu obsahuje 3 položky. Dvě se týkají modulu „Odkazy“ a jedna modulu „Smlouvy“. První položka s názvem „Oddělení“ zobrazí druhé menu, nebo také sub menu, ve kterém se nachází seznam všech oddělení. Vybráním jedné konkrétní položky se již zobrazí požadované odkazy. Ukázku můžeme vidět na obrázku č. 12. Druhá položka s názvem „Kategorie“ zobrazí menu s názvy všech kategorií. Ukázku můžeme vidět na obrázku č. 13. Třetí položka, patřící do modulu „Smlouvy“, zobrazí v menu pouze dvě položky. A to položku „Smlouvy“ a „Přidat smlouvu“. Pokud vybereme položku „Smlouvy“, uvidíme seznam všech zobrazených smluv podle jednotlivých kategorií. Nad tímto seznamem se ještě nachází ovládací prvky, které nám umožňují filtrovat zobrazené smlouvy podle různých kritérií. Druhá položka, „Přidat smlouvu“, nás přesměruje na webovou stránku, která nám umožní přidat do systému novou smlouvu. Náhledy obou těchto stránek můžeme vidět na obrázku č. 14 a obrázku č. 15. Webová stránka neobsahuje žádné zápatí stránky. Na pravé straně stránky se nachází informační panel. Jsou zde zobrazeny kontakty na administrátory aplikace, přesměrování na webovou stránku se žádostí přidání nového odkazu (Obrázek č. 16) a odkazy na důležité informace pro uživatele.



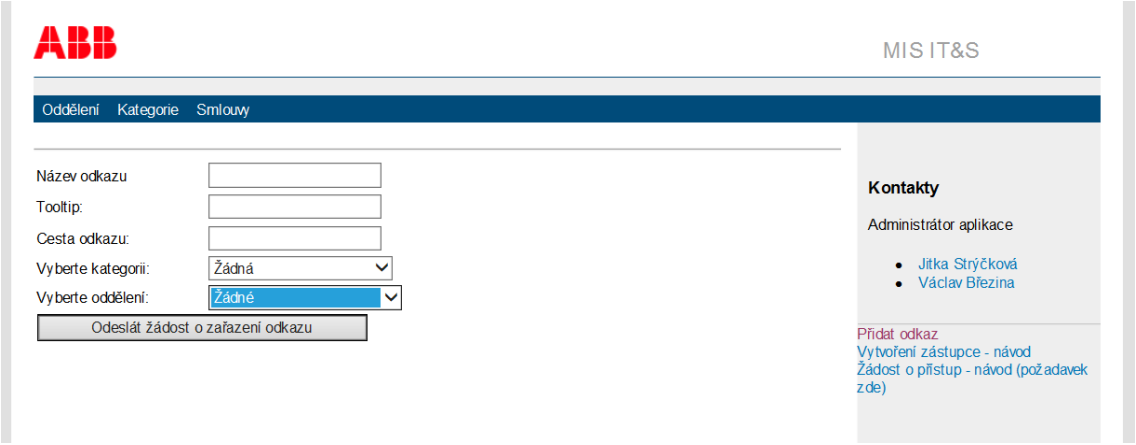
Obrázek 12: Grafický vzhled - Oddělení [zdroj: vlastní zpracování]



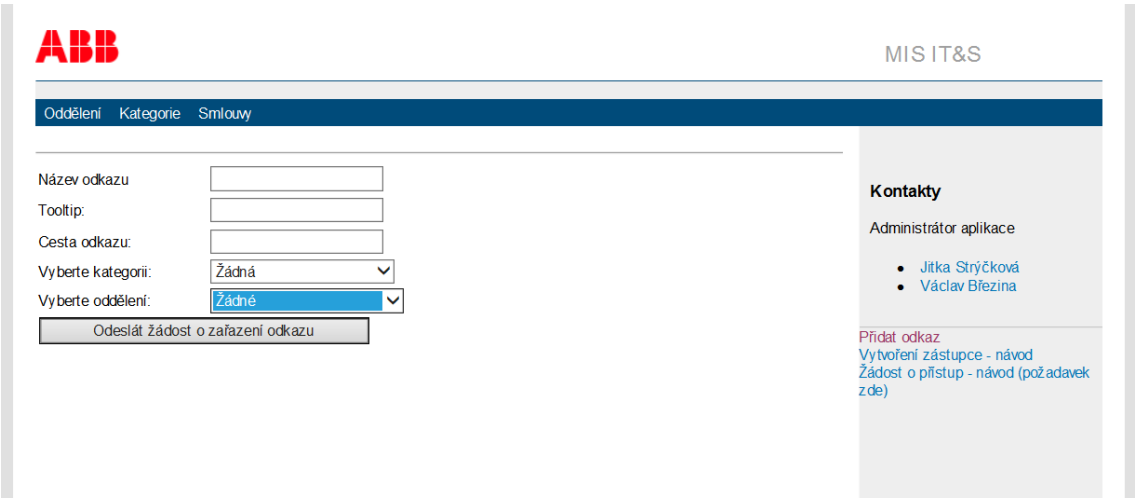
Obrázek 13: Grafický vzhled - Kategorie [zdroj: vlastní zpracování]



Obrázek 14: Grafický vzhled - Přehled smluv [zdroj: vlastní zpracování]



Obrázek 15: Grafický vzhled - Vložit odkaz [zdroj: vlastní zpracování]



Obrázek 16: Grafický vzhled - Vložit smlouvu [zdroj: vlastní zpracování]

4.4.Možnosti rozšíření

4.4.1. Login – úprava vlastních oblíbených odkazů

Každý uživatel využívá některé odkazy častěji než ostatní. Pokud se odkazy nachází v různých kategoriích nebo odděleních, uživatel sice už ví, kde má hledat, ale musí listovat v kategoriích či odděleních. Tento problém by mohla vyřešit stránka s oblíbenými odkazy. Každý uživatel by si vytvořil vlastní seznam oblíbených odkazů, které by měl zobrazené na jedné stránce, ač by spadaly do různých kategorií nebo oddělení. Aby toto mohlo fungovat, musel by se každý uživatel přihlásit. Toto by obnášelo rozšíření databáze o uživatelská jména a hesla. Toto by vyžadovalo větší stupeň zabezpečení. Lepším řešením by mohlo být propojení informačního systému s doménovým jménem a heslem

a použití SSO (Single Sign-On), jelikož je aplikace přístupná pouze v intranetu firmy. Po spuštění aplikace by byl hned uživatel identifikován a byly by zobrazeny jeho oblíbené odkazy.

4.4.2. Administrace – úprava přidávání odkazů

Veškeré úpravy (editace, smazání, přidání) odkazů je možné pouze upravením záznamu přímo v databázi. Díky tomuto nemůže běžný uživatel s daty manipulovat a o veškeré úpravy musí žádat správce aplikace. Výhodou tohoto řešení je úplná kontrola nad aplikací administrátorem. Nevýhodou je zatěžování správce aplikace častými úkoly k aktualizaci dat. Tento problém by mohla vyřešit stránka administrace odkazů. Tato stránka by byla zabezpečena uživatelským jménem a heslem. Toto by zabránilo možnosti editovat data všem uživatelům. Editovat by mohli pouze pověření uživatelé. Tímto jménem a heslem by disponoval pravděpodobně vedoucí oddělení nebo pověřená osoba z oddělení a měla by možnost editovat pouze odkazy, které patří do patřičného oddělení. V případě, že by se změnilo umístění cílového souboru, vedoucí oddělení by mohl cestu k odkazu aktualizovat sám, stejně tak jako změnit název, text tooltipu apod.

5. ZÁVĚR

Tato bakalářská byla zpracována pro firmu ABB s. r. o., konkrétně pro Divizi přístrojových transformátorů a senzorů (IT&S's) ve spolupráci s vedoucími jednotlivých oddělení.

Hlavním cílem této práce bylo vytvoření části informačního systému pro lepší sdílení interních informací mezi zaměstnanci. Součástí tohoto návrhu byla analýza firmy a jejich informačních technologií, definice požadavků na systém formou společné diskuze a samotný návrh informačního systému.

Řešením je návrh a implementace informačního systému pojmenovaného „MIS IT&S", jejíž celý název zní Management Information System Instrument Transformers and Sensors. Tento informační systém má formu webové stránky a splnil všechny definované požadavky. Díky tomuto informačnímu systému získala firma nástroj pro lepší sdílení interních informací, včetně papírových smluv.

SEZNAM POUŽITÉ LITERATURY

(1) VYMĚTAL, D. *Informační systémy v podnicích: teorie a praxe projektování*. Praha: Grada, 2009. ISBN 978-80-247-3046-2.

(2) OPPEL, J. A. *SQL bez předchozích znalostí*. Brno: Computer Press, 2012. ISBN 978-80-251-1707-1.

(3) PÍSEK, S. *Access 2013: podrobný průvodce*. Praha: Grada, 2013. ISBN 978-80-247-4746-0.

(4) CONOLLY, T., C. BEGG a R. HALOWCZAK. *Mistrovství – databáze: Profesionální průvodce tvorbou efektivních databází.* Brno: Computer Press, 2009. ISBN 978-80-251-2328-7.

(5) GILFILLAN, I. *Myslíme v jazyce MySQL 4*. Praha: Grada, 2003. ISBN 80-247-0661-X.

(6) PROCHÁZKA, D. *Oracle: průvodce správou, využitím a programováním nad databázovým systémem*. Praha: Grada Publishing, 2009. ISBN 978-80-247-2762-2.

(7) ŘEPA, V. *Vývojové trendy metodik vývoje informačních systémů* [online]. 2000 [cit. 2015-02-18]. Vysoká škola ekonomická v Praze. Dostupné z: http://nb.vse.cz/~repa/veda/EurOpen99%20Paper.pdf

(8) LACKO, Ľ. *1001 tipů a triků pro SQL*. Brno: Computer Press, 2011. ISBN 978-80-251-3010-0.

(9) HERNANDEZ, J. M. *Návrh databází*. Praha: Grada Publishing, 2006. ISBN 80-247-0900-7.

(10) KŘÍŽ, J. a P. DOSTÁL. *Databázové systémy*. Brno: Akademické nakladatelství CERM, 2005. ISBN 80-214-3064-8.

(11) HERNANDEZ, J. M. a J. L. VIESCAS. *Myslíme v jazyku SQL tvorba dotazů*. Praha: Grada Publishing, 2004. ISBN 80-247-0899-X.

(12) GÁLA, L. *Podniková informatika: počítačové aplikace v podnikové a mezipodnikové praxi, technologie informačních systémů, řízení a rozvoj podnikové informatiky*. Praha: Grada, 2006. ISBN 80-247-1278-4.

(13) PRICE, J. *C#: programování databází*. Praha: Grada, 2005. ISBN 80-247-0982-1.

(14) BABIUCH, Marek. *Programování aplikací pro internet II* [online]. 2007 [cit. 2015-05-05]. VŠB TU Ostrava. Dostupné z WWW: http://www.elearn.vsb.cz/archivcd/FS/PAI2/PaPi_UcebniText.pdf

(15) ABB. *Historie ABB ČR* [online]. 2011 [cit. 2014-11-21]. Dostupné z: http://new.abb.com/cz/o-nas/historie/historie-abb-cr

(16) MAASSEN, André. *SAP R/3: kompletní průvodce*. Brno: Computer Press, 2007. ISBN 978-80-251-1750-7.

(17) ABB. *ABB Brno IT Stakeholder Requests, version 1.0*. Brno: ABB, 2011.

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH

Příloha č. 1 Zdrojový kód SQL

```
create table Kategorie (

ID_Kategorie int identity not null,
Nazev varchar(50) not null,
Uzivatel varchar(20),
Datum datetime

constraint PK_Kategorie primary key (ID_Kategorie)
)

create table Smlouva (

ID_Smlouva int identity not null,
Dodavatel varchar(150) not null,
Cislo_Smlouvy varchar(150) not null,
Predmet_Smlouvy varchar(150) not null,
Datum_Podpis varchar(50) not null,
Zodpovednost varchar(50) not null,
Uzivatel varchar(15),
Datum datetime,
Odkaz varchar(300) not null

constraint PK_Smlouva primary key (ID_Smlouva)
)

create table Smlouva_kategorie(

ID_Smlouva_Kategorie int identity not null,
ID_Smlouva int not null,
ID_Kategorie int not null,

constraint PK_Smlouva_Kategorie primary key(ID_Smlouva_Kategorie),
constraint FK_Smlouva foreign key (ID_Smlouva) references Smlouva
(ID_Smlouva),
constraint FK_Kategorie foreign key (ID_Kategorie) references
Kategorie (ID_Kategorie)
)

create table odkaz(

ID_Odkaz int identity not null,
Nazev varchar(200) not null,
Tooltip varchar(400) not null,
Img varchar(100) not null,
Odkaz varchar(400) not null

constraint PK_Odkaz primary key (ID_Odkaz)
)

create table Kategorie_odkaz(

ID_Kategorie int identity not null,
Nazev varchar(50) not null

constraint PK_Kategorie_odkaz primary key(ID_Kategorie)
)
```

```
create table Oddeleni_odkaz(

ID_Oddeleni int identity not null,
Nazev varchar(50) not null

constraint PK_Oddeleni_odkaz primary key(ID_Oddeleni)
)

create table Rozrazeni_odkaz (

ID_Rozrazeni int identity not null,
ID_Odkaz int not null,
ID_Kategorie int not null,
ID_Oddeleni int not null

constraint PK_Rozrazeni_odkaz primary key (ID_Rozrazeni),
constraint FK_Odkaz foreign key (ID_Odkaz) references Odkaz(ID_Odkaz),
constraint FK_Kategorie_Odkaz foreign key (ID_Kategorie) references
Kategorie_odkaz(ID_Kategorie),
constraint FK_Oddeleni_odkaz foreign key (ID_Oddeleni) references
Oddeleni_odkaz(ID_Oddeleni)
)

create table Zadost_odkaz(

ID_Zadost int identity not null,
Nazev varchar(150) not null,
Tooltip varchar(300) not null,
Cesta varchar(300) not null,
Kategorie varchar(50)not null,
Oddeleni varchar(50)not null,
Datum datetime,
Uzivatel varchar(50)

constraint PK_Zadpst primary key (ID_Zadost)
)

create table Error(
ID_Error int identity not null,
Error_text text not null,
Datum datetime

constraint PK_Error primary key (ID_Error)
)

CREATE PROCEDURE odkaz_byoddeleni
@id_oddeleni INT
AS

SELECT o.Odkaz,o.Nazev,o.Tooltip,o.Img,o.ID_Odkaz,odd.Nazev AS
Nazev_oddeleni ,odd.ID_Oddeleni FROM Odkaz o,Oddeleni_odkaz odd,
Rozrazeni_odkaz ro
WHERE odd.ID_Oddeleni = @id_oddeleni and odd.ID_Oddeleni =
ro.ID_Oddeleni and o.ID_Odkaz = ro.ID_Odkaz
ORDER BY o.Nazev
```

```
CREATE PROCEDURE odkaz_bykategorie
@id_kategorie INT

AS

SELECT DISTINCT o.Odkaz,o.Nazev,o.Tooltip,o.Img,o.ID_Odkaz,ko.Nazev AS
Nazev_kategorie,ko.ID_Kategorie FROM odkaz o, Kategorie_odkaz ko,
Rozrazeni_odkaz ro
WHERE ko.ID_Kategorie = @id_kategorie and ko.ID_Kategorie =
ro.ID_Kategorie
and o.ID_Odkaz = ro.ID_Odkaz
ORDER BY o.Nazev

CREATE PROCEDURE listbox_oddeleniodkazu
AS

SELECT nazev FROM Oddeleni_odkaz ORDER BY nazev DESC

CREATE PROCEDURE listbox_kategorieodkazu
AS

SELECT nazev FROM Kategorie_odkaz ORDER BY nazev DESC

CREATE PROCEDURE listbox
AS

SELECT * FROM Kategorie

CREATE PROCEDURE kategorie_smlouva
@ID_kategorie INT
AS

SELECT s.ID_Smlouva, CONVERT(VARCHAR,s.Datum_podpis, 101) , s.Odkaz,
s.Predmet_smlouvy, s.Zodpovednost, s.Dodavatel,
s.Datum_podpis,s.Cislo_smlouvy, k.Nazev AS Kategorie, k.ID_kategorie
FROM Smlouva s, Smlouva_kategorie sk, Kategorie k
WHERE k.ID_Kategorie = @ID_kategorie and s.ID_Smlouva = sk.ID_Smlouva
and sk.ID_Kategorie = k.ID_Kategorie

CREATE PROCEDURE smlouva_nacti
@id_smlouva INT AS

SELECT s.Cislo_smlouvy,  s.Datum_podpis  AS
datum,s.Dodavatel,s.Odkaz,s.Predmet_smlouvy,s.Zodpovednost,lower(s.Dod
avatel) as hledej_dodavatel,lower(s.Cislo_smlouvy) as
hledej_cislo,lower(s.Predmet_smlouvy) as
hledej_predmet,lower(s.Zodpovednost) as hledej_osoba
FROM smlouva s
WHERE s.ID_smlouva = @id_smlouva
```