Univerzita Pardubice
Fakulta ekonomicko-správní
Ústav systémového inženýrství a informatiky

Možnosti vyhodnocení uživatelských datových požadavků

Martin Pátek

Bakalářská práce
2019

Univerzita Pardubice
Fakulta ekonomicko-správní
Akademický rok: **2018/2019**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Martin Pátek**
Osobní číslo:	**E16053**
Studijní program:	**B6209 Systémové inženýrství a informatika**
Studijní obor:	**Informatika ve veřejné správě**
Název tématu:	**Možnosti vyhodnocení uživatelských datových požadavků**
Zadávající katedra:	**Ústav systémového inženýrství a informatiky**

Zásady pro vypracování:

Cílem práce je charakterizovat možnosti pro vyhodnocení uživatelských datových požadavků. Práce bude zaměřena na zdroje datových a funkčních požadavků s důrazem na možnosti jejich zhodnocení ve smyslu relevantnosti, priorit, aj.

Osnova:
- Základní pojmy související se zpracovávanou problematikou.
- Identifikace zdrojů.
- Hodnocení požadavků.

Rozsah grafických prací:

Rozsah pracovní zprávy: **cca 35 stran**

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

BRUCKNER, Tomáš. *Tvorba informačních systémů: principy, metodiky, architektury*. **Praha: Grada, 2012. Management v informační společnosti. ISBN 978-80-247-4153-6.**
KEŘKOVSKÝ, Miloslav. *IS/IT strategie krok za krokem: teorie pro praxi*. **V Praze: C.H. Beck, 2015. C.H. Beck pro praxi. ISBN 978-80-7400-272-4.**
SVOZILOVÁ, Alena. *Zlepšování podnikových procesů*. **Praha: Grada, 2011. Expert (Grada). ISBN 978-80-247-3938-0.**

Šimonové

Vedoucí bakalářské práce: **doc. Ing. Stanislava Šimonová, Ph.D.**
Ústav systémového inženýrství a informatiky

Datum zadání bakalářské práce: **3. září 2018**

Termín odevzdání bakalářské práce: **30. dubna 2019**

L.S.

doc. Ing. Romana Provazníková, Ph.D.
děkanka

doc. Ing. Pavel Petr, Ph.D.
vedoucí ústavu

V Pardubicích dne 3. září 2018

PROHLÁŠENÍ

Prohlašuji, že jsem tuto práci vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako Školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Beru na vědomí, že v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, a směrnicí Univerzity Pardubice č. 9/2012, bude práce zveřejněna v Univerzitní knihovně a prostřednictvím Digitální knihovny Univerzity Pardubice.

V Pardubicích dne 30. 4. 2019 Martin Pátek

PODĚKOVÁNÍ:

Tímto bych rád poděkoval celé své rodině za jejich podporu po celou dobu mého studia, jejich povzbuzování a cenné rady při zpracování této práce.

ANOTACE

Tato bakalářská práce se zabývá uživatelskými požadavky, jejich zdroji a možnostmi vyhodnocení. Mimo to se čtenář dozví, jaké existují techniky pro sběr požadavků a také s jakými riziky se může v souvislosti s požadavky setkat.

KLÍČOVÁ SLOVA

Požadavek, Hlas zákazníka, DFD, případ užití, matice CRUD, metoda MoSCoW, KANO model, Bublinkové řazení, QFD, Ishikawův diagram

TITLE

Options for evaluating user data requirements

ANNOTATION

This bachelor thesis deals with user requirements, their sources and possibilities of evaluation. In addition, the reader will learn what methods for collecting requirements exist and what risks they may enxounter in relation to requirements.

KEYWORDS

Requirement, Voice of the Customer, DFD, Use case, CRUD matrix, MoSCoW method, KANO model, Bubble sort, QFD, Ishikawa diagram

OBSAH

SEZNAM ILUSTRACÍ A TABULEK

SEZNAM ZKRATEK A ZNAČEK

CTQ	Critical to quality – Klíčová kritéria kvality
DFD	Data flow diagram – Diagram datových toků
HW	Hardware
IS	Informační systém
KUP	Katalog uživatelských požadavků
PC	Osobní počítač
QFD	Quality Function Deployment – Rozpracování funkcí kvality
SQFD	Software Quality Function Deplyment -
UML	Unified Modeling Language – Unifikovaný modelovací jazyk
VoC	Voice of the Customer – Hlas zákazníka

ÚVOD

Při tvorbě jakéhokoliv projektu jsou požadavky jeho nedílnou součástí. Bez správně vyjádřených a uchopených požadavků není možné vytvářet kvalitní produkty a poskytovat kvalitní služby. Z tohoto důvodu je potřeba zajistit veškeré dostupné zdroje a sesbírat veškeré požadavky, které se projektu týkají. Tato práce bude pojednávat jak o samotných požadavcích, přes jejich zdroje a techniky sběru až po samotné vyhodnocení a popis základních rizik s požadavky spojených. Práce bude rozdělena do 5 kapitol. V úvodní kapitole se čtenář dozví, co to požadavek je, jaké jsou jeho druhy, dále co je to katalog uživatelských požadavků a v neposlední řadě co jsou to data, která jsou stejně důležitá jako jakékoliv jiné požadavky, protože s nedostatečně zajištěnými daty nebude nikdy výsledný produkt pracovat správně, i když bude mít perfektně připravené funkce. Druhá kapitola bude věnována zdrojům požadavků. Půjde o zmapování i jiných možností, než jsou pouhé rozhovory se zákazníky. Ve třetí kapitole budou představeny jednotlivé techniky vhodné pro sběr požadavků a dále také metody, které mohou pomoci při hledání požadavků chybějících. Předposlední, čtvrtá kapitola, bude zaměřena na samotné vyhodnocení. V této kapitole si postupně představíme vybrané metody, jako jsou metoda MoSCoW, model KANO, bublinkové řazení a rozpracování funkcí kvality (QFD). Dále v této kapitole bude vytvořena srovnávací tabulka, porovnávající vybrané metody. Poslední kapitola se bude věnovat rizikům spojeným s požadavky. Dále v ní bude zmíněno použití Ishikawova diagramu, který je vhodný pro hledání chyb v požadavcích.

Cílem práce je charakterizovat možnosti pro vyhodnocení uživatelských datových požadavků. Práce bude zaměřena na zdroje datových a funkčních požadavků s důrazem na možnosti jejich zhodnocení ve smyslu relevantnosti, priorit, aj.

1 DRUHY POŽADAVKŮ

Pro popis požadavku neexistuje žádná přesně stanovená definice. Jedním z možných vysvětlení pojmu požadavek je názor Karla E. Wiegerse, který si požadavky vysvětluje jako dané vlastnosti, kterými musí systém disponovat, aby byl využitelný pro některého z uživatelů systému. Další možnou definicí pro popis požadavku je výrok, že *„Požadavky jsou popis toho, co všechno by se mělo implementovat. Popisují chování systému a jeho vlastnosti a mohou představovat nějaká omezení procesu vývoje systémů“* (Sommerville a Sawyer, 1997) [28].

Dle slovníku softwarové terminologie z roku 1990 se požadavek definuje jako [28]:

- *„Podmínka nebo funkce, kterou uživatel potřebuje pro řešení problému nebo dosažení nějakého cíle.“*
- *„Podmínka nebo funkce, kterou musí systém nebo jeho část splňovat, aby vyhověl smlouvě, standardu, specifikaci nebo jinému dokumentu, jenž se na něj formálně vztahuje.“*
- *„Dokumentovaná podoba některého z předchozích dvou bodů.“*

Podobně jako samotné stanovení významu slova požadavek je na tom i rozdělení jednotlivých typů požadavků, které jsou v různých publikacích děleny jinak. Požadavky lze rozdělit do tří základních skupin. Jedná se o požadavky podnikatelské, uživatelské a funkční. Navíc, kromě těchto skupin, každý systém obsahuje seznam parametrických požadavků, které jsou závislé na prostředí systému [28].

1.1 Podnikatelské požadavky

Podnikatelské požadavky jsou základními požadavky na nejvyšší úrovni. Definují nám důvody, dle kterých se organizace rozhodla daný systém pořídit, a vymezují cíle, kterých chce firma dosáhnout tímto zavedením systému [28].

Tyto základní požadavky většinou vychází od hlavního investora, zákazníka, marketingového oddělení a dalších [28].

1.2 Uživatelské požadavky

Pomocí uživatelských požadavků dochází k popisu cílů uživatelů systému a jejich úkolů, které musí systém umožnit provést. Pro zápis těchto požadavků se využívají případy užití,

scénáře, nebo tabulky, které popisují reakce na možné události. Za příklad takového případu užití můžeme považovat rezervaci knih v Univerzitní knihovně [28].

1.3 Funkční požadavky

Behaviorální požadavky neboli požadavky na chování, jak někdy bývají funkční požadavky označovány, popisují softwarovou funkčnost, která musí být vývojáři do systému zavedena, aby jeho uživatelé mohli vykonávat svou práci, a tím plnit podnikatelské požadavky [28].

Příkladem funkčního požadavku může být zaslání potvrzovacího e-mailu studentovi nebo jinému uživateli knihovny o rezervaci jím zvolené knížky [28].

Jiným pohledem je základní dělení požadavků na funkční a nefunkční. Jedná se o rozdělení, které v literatuře vyskytuje častěji než druhy uvedené výše. Definice funkčních požadavků se shoduje s již uvedeným popisem. Jedná se tedy o požadavky popisující funkce nebo služby, které jsou od systému očekávány.

V případě nefunkčních požadavků můžeme mluvit a jakési nadstavbě požadavků funkčních. Jedná se o nezbytné vlastnosti každého softwaru. Jsou to vlastnosti, které jsou potřebné v závislosti na prostředí a souvislostech [33].

Mezi nefunkční požadavky se řadí např. požadavky na [33]:

- Spolehlivost
- Bezpečnost
- Výkonnost
- Podporu během provozu

1.4 Katalog uživatelských požadavků

Během vývoje softwaru může docházet k občasnému odklonu od zákaznických požadavků, zde mohou nastat situace, že nějaké požadavky jsou plněny pouze okrajově a některé dokonce nejsou plněny vůbec. Namísto plnění zadaných požadavků se poté dodavatelé zabývají problémem automatizovaných kontrol dat, navzdory tomu, že o nich zákazníci vůbec nehovořili. Jedná se o tzv. uměle vytvořené požadavky. Takto dosazené automatické kontroly mohou často omezit využívání informačního systému [15].

Pro omezení tohoto odklonu v průběhu vývojového cyklu softwaru lze využít rozšíření KUP. Při tvorbě softwaru se katalog využívá hlavně pro popis chování systému a datové struktury [15].

Obvyklý KUP obsahuje [15]:

- Popis okolí systému
- Požadavky na chování systému
- Požadavky na data systému
- Ostatní požadavky jako jsou bezpečnost, spolehlivost, výkon a další

Výše uvedené části KUP jsou popsány textem formou tabulky, kde každý z požadavků je popsán sadou atributů.

Základní atributy požadavků						
Číslo	Název	Popis	Priorita	Stav řešení	Datum poslední změny	Autor

Obrázek 1: Sada atributů požadavku

Zdroj: zpracováno dle [15]

Základní verze KUP se využívá především ve fázi analýzy a návrhu. Rozšířená verze slouží ke zvýšení kvality i v dalších fázích. Pro každou fázi je dobré vytvořit nový atribut v katalogu požadavků, který bude obsahovat odkaz na danou fázi [15].

Odkazem v dané fázi se myslí [15]:

- analýza – odkaz na kapitolu dokumentu s popisem rozhraní, procesů, tříd,
- návrh a implementace – odkaz na modul, třídu, databázovou tabulku,
- testování – odkaz na testovací scénář,
- uživatelská dokumentace – odkaz na část s popisem obsluhy požadavku.

1.5 Data, informace, znalost

Stejně tak, jako potřebujeme, aby daný systém uměl provádět veškeré operace s daty, které ke své práci potřebujeme, jsou důležitá i data samotná. Data představují základní údaje popisující realitu. Jedná se o fyzicky zaznamenané výsledky, fakta, poznatky… Veškerá data musí existovat a jsou uložena buď v lidské mysli, na papíře nebo na jiných nosičích [22][35].

Pojem data se využívá pro údaje [6]:

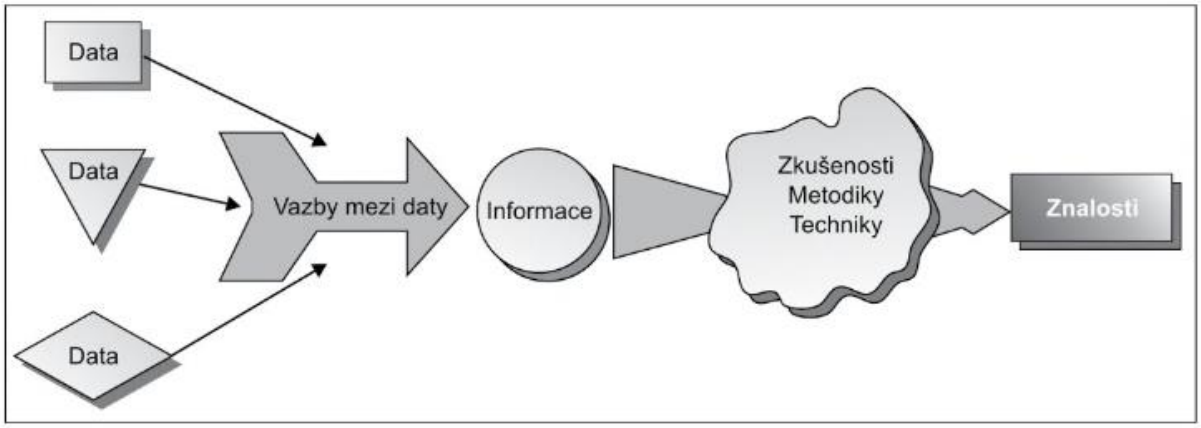
- Numerické
- Textové

- Obrazové
- Další

Jedním z jiných možných dělení dat jsou data [35]:

- Kvantitativní – číselné hodnoty, které vyjadřují charakteristiku jevu (např. hmotnost, cena a další)
- Kvalitativní – nečíselné hodnoty, které vyjadřují charakteristiku jevu (např. spokojenost uživatelů)

Pokud data pro podnik mají vyšší vypovídající kvalitu, označují se jako informace. Informace přiřazují datům význam. S informacemi dále pracujeme v informačním systému, kde požadované informace zpracováváme, uchováváme a shromažďujeme. Pokud pomocí zkušeností odvozujeme něco z informací, jedná se o znalost [22].



Obrázek 2: Data, informace, znalosti

Zdroj: [6]

2 ZDROJE POŽADAVKŮ

Proces získávání uživatelských požadavků není lehká činnost. V praxi se totiž setkáváme se dvěma různými skupinami uživatelů. První skupinu tvoří uživatelé, kteří mají konkrétní představu o tom, jak by měl nový systém fungovat, a tím pádem je pro nás snadné získat od nich přesné požadavky. Druhá skupina uživatelů vlastní představu nemá a nechává veškeré snahy na tvůrcích systému a žádají systém, který uspokojí jejich potřeby, aniž by museli spolupracovat na získávání požadavků [13].

Pro získání všech uživatelských požadavků se využívají různé zdroje v závislosti na typu projektu a vývojářském prostředí. Abychom získali veškeré, správné a úplné požadavky, je třeba využít více různých pohledů a zdrojů, což představuje složitý komunikační princip práce s požadavky. Mezi často využívané zdroje patří [28]:

2.1 Rozhovory s uživateli

Rozhovory s koncovými uživateli systému jsou základním procesem pro zjištění skutečných potřeb těchto uživatelů. Uživatelé informačního systému se od sebe navzájem odlišují určitými faktory, mezi které můžeme zařadit například [28]:

- Četnost využívání IS
- Zkušenosti s výpočetní technikou
- Využívané funkce
- Úkoly ve firmě
- Oprávnění

Dle zmíněných faktorů lze uživatele rozdělit do uživatelských tříd, kdy jeden uživatel může patřit i do více tříd najednou. Každá z vytvořených tříd má své vlastní požadavky, které jsou závislé na úkolech, které plní ve firmě uživatelé dané třídy. Za uživatelskou třídu nemusíme považovat pouze uživatele, ale mohou se v ní vyskytovat například i další aplikace nebo HW komponenty, které přijdou do styku se systémem [28].

2.2 Dokumentace popisující stávající nebo konkurenční systém

Z dokumentace jiných IS lze vyčíst firemní a průmyslové standardy, legislativní nařízení, nebo zákony, které musíme dodržet při tvorbě nového systému, pokud takovéto údaje poskytují. Z dokumentů dále lze odvodit stávající nebo dokonce i budoucí podnikatelské procesy. Co se

týká konkurenčních systémů, lze využít například recenze v tisku k získání konkurenční výhody [28].

2.3 Specifikace systémových požadavků

Jedná se o obecnou systémovou specifikaci, jejíž pomocí se IS popisuje jako celek [28]. Např. chceme-li využívat nějakou aplikaci ve svém osobním počítači, stanoví nám systémový požadavek, jaký procesor by PC měl mít, kolik místa aplikace zabere, s jakým operačním systémem je kompatibilní a další požadavky (rozlišení obrazovky, RAM paměť, …).

2.4 Chybová hlášení a požadavky na zlepšení stávajícího IS

Chybová hlášení patří k cenným zdrojům požadavků v případě, že firma již nějaký IS vlastní. Využívají se zde znalosti technické podpory, která má přehled o problémech, se kterými se uživatelé potkávají během své práce se současným systémem a dále mohou přicházet s nápady, které během komunikace s uživateli zaslechli jako nápady na zlepšení systému [28].

2.5 Průzkumy trhu a uživatelské dotazníky

Pomocí průzkumů a dotazníků lze nasbírat značné množství dat od mnoha různých uživatelů. Zároveň průzkum slouží k porovnání s aktuální představou požadavků. Při provádění průzkumu je důležité ho předem vyzkoušet, protože může docházet k situacím, kdy jsme položili otázku nejednoznačně, nebo jsme dokonce na nějaké otázky úplně zapomněli [28].

2.6 Sledování uživatelů při práci

Sledováním uživatelů lze správně pochopit jejich práci se systémem. Analytik tráví čas s uživateli aktuálního systému, čímž si může lehce ověřit informace, které získal během předchozích rozhovorů, dále může najít problémy stávajícího systému nebo vysledovat jakým způsobem by měl nový IS podporovat práci uživatelů [28].

Důležité ale je, aby analytik vysledované činnosti zobecňoval, abychom se ujistili, že se požadavky týkají celé skupiny uživatelů a nejen jednotlivců [28].

2.7 Analýza uživatelských úkolů

Z úkolů, které jsou vyžadovány od uživatelů, aby je mohli pomocí systému plnit, můžeme vyvodit funkční požadavky. K tomu se využívá analýza případů užití. Je důležité zapsat veškeré informace, které se objevují jak na vstupu, tak i na výstupu úkolu a v neposlední řadě také zdroje těchto informací [28].

2.8 Události a odpovědi

Jedná se o velmi vhodný přístup využívaný pro real-time systémy, které načítají a zpracovávají data, chybové kódy a řídící signály z vnějších zařízení. Patří sem seznam všech vnějších událostí, které mohou nastat a jejich vhodných odpovědí [28].

2.9 Hlas zákazníka (VoC)

Hlasem zákazníka se rozumí termín používaný k popisu stanovených a nestálých potřeb nebo požadavků zákazníka. Hlas zákazníka lze zachytit různými způsoby [36]:

- Přímé rozhovory
- Dotazníky
- Specifikace zákazníků
- Pozorování a další

Sbírání hlasu zákazníka je velmi důležité, protože plnění jeho přání je základním předpokladem pro jeho spokojenost. Bez požadavků určených zákazníkem nejsme schopni jeho potřebám plně porozumět. Zákaznická data lze sbírat dvěma způsoby. V případě aktivního sběru mluvíme o oslovování zákazníků, zasílání dotazníků či provádění průzkumu na trhu. Při pasivním sběru, který bývá označován jako reaktivní, řešíme například stížnosti a reklamace od zákazníků [26].

Rozlišujeme dva základní typy zákazníků [26]:

- Interní – jedná se např. o další oddělení společnosti
- Externí – koncový zákazník, ke kterému jde konečný podnikový výstup

Veškeré zákaznické požadavky se týkají funkcionality navrhovaného systému. Pokud bychom pracovali na IS, který bude schopen reagovat i na podnikatelské požadavky a tím napomoci plnění poslání firmy, budeme hledat nejen hlas zákazníka, ale také i hlas podniku (Voice of Business), to znamená potřeby, které mají jasnou vazbu na strategii podniku [21].

3 SBĚR POŽADAVKŮ

Sběr požadavků je základní činností při práci s požadavky, jedná se o hledání především uživatelských potřeb a omezení kladených na SW. Jak bylo již dříve uvedeno, uživatelské požadavky se týkají popisu cílů uživatelů systému a jejich úkolů, které musí systém umožnit provést [28].

Sbírání požadavků je s velkou pravděpodobností nejsložitější, nejdůležitější, a tím pádem i komunikačně nejnáročnější část v procesu vývoje jakéhokoliv SW. Základem úspěšného sběru požadavků je spolupráce uživatelů s vývojáři. Z následujících tabulek lze vyčíst základní práva a povinnosti zákazníků/uživatelů [28].

Tabulka 1: Seznam základních práv softwarového zákazníka

1) Očekávat, že s vámi analytik bude mluvit vaší řečí.
2) Očekávat, že se analytik poučí o fungování vaší firmy a vašich cílech.
3) Očekávat, že analytik získané požadavky zpracuje do podoby psané specifikace.
4) Nechat si od analytika vysvětlit všechny výstupy získané během zpracovávání požadavků.
5) Očekávat, že se k vám analytici a vývojáři budou chovat s respektem a během vašich vzájemných setkání budou udržovat profesionální a vstřícný přístup.
6) Očekávat, že vám analytici a vývojáři budou předkládat nápady a možné varianty řešení vašich požadavků a jejich implementace.
7) Zadat požadavky na systém tak, aby se dobře používal.
8) Dostat příležitost upravit své požadavky tak, aby se při řešení dal použít nějaký existující software.
9) Získat při každé změně co nejupřímnější odhad nákladů, dopadů a kompromisů, které tato změna způsobí.
10) Dostat systém, který splňuje všechny vaše nároky na funkce a kvalitu do té míry, na které jste se shodli s vývojáři.

Zdroj: [28]

Tabulka 2: Seznam základních povinností softwarového zákazníka

1) Vysvětlit analytikům a vývojářům procesy ve své firmě a vysvětlit jim firemní žargon.
2) Obětovat tolik svého času, kolik bude potřeba pro sdělení, vyjasnění a postupné upřesnění všech požadavků.
3) Být při popisování požadavků co nejkonkrétnější a nejpřesnější.
4) Rozhodovat o požadavcích.
5) Respektovat vývojářův odhad ceny a složitosti implementace vašich požadavků.
6) Ve spolupráci s vývojáři rozdělit prioritu jednotlivých funkčních požadavků, funkcí systému a případů užití.
7) Důkladně přečíst specifikaci požadavků a vyzkoušet prototypy.
8) Hlásit změny požadavků včas.
9) Provádět změny v požadavcích podle procesu, který mají vývojáři pro tyto případy nachystaný.
10) Respektovat procesy, které analytik používá pro práci s požadavky.

Zdroj: [28]

Prvním krokem fáze sběru požadavků je samotné plánování sběru, ve kterém se stanovují zdroje požadavků, termíny sběru a jeho výsledky. Pouze naplánováním těchto aspektů dosáhneme toho, že se uživatelé nebudou sběru vyhýbat. Takovýto plán může obsahovat například cíle sbírání požadavků, kam můžeme zařadit ověření dat o trhu nebo zkoumání případů užití. Strategie a procesy, do kterých řadíme průzkumy, workshopy, rozhovory a jejich kombinace a další techniky. Typy výstupů řešící podrobnou specifikaci požadavků, vypracování seznamu případů užití nebo také analýzu výsledků průzkumu. Dalším bodem je odhad termínů a potřebných zdrojů, kde uvádíme např. zástupce jak z řad vývojářů, tak i uživatelů a odhadu času. V neposlední řadě je dobré si před samotným sběrem také stanovit rizika, která by mohla ohrozit zmíněné aktivity a také možné způsoby jejich řešení [28].

3.1 Techniky shromažďování požadavků

Individuální rozhovory a setkání

Rozhovory a setkání jsou velmi využívanou metodou pro shromažďování požadavků. Individuální rozhovory vyžadují určité plánování a přípravy před samotným pohovorem a

následné dokumentování výsledků. Technika spočívá v pokládání různých otázek jak otevřených, tak uzavřených, které nám pomohou objasnit požadavky uživatelů nebo jiných zainteresovaných osob. Hlavní výhodou je, že pomocí rozhovorů můžeme získat důležité informace od uživatelů, ti se necítí omezeni tím, co potřebují nebo nedostatkem jejich znalostí. Dalším kladem této techniky je, že někteří uživatelé spíše sdělí své názory v soukromí než při setkání více lidí. Největší nevýhodu můžeme spatřit v rychlosti sběru informací, protože promlouvat s jednotlivými uživateli může být velmi zdlouhavé [18].

Skupinové rozhovory a setkání

Při skupinovém setkání nebo rozhovorech se jedná o stejný princip jako u první zmíněné techniky. Oproti ní má ale své výhody i nevýhody spojené s vyšším počtem zúčastněných osob. Výhodou samozřejmě je, že s vyšším počtem osob lze získat více požadavků než při individuálních rozhovorech a za předpokladu, že jsou zúčastněné osoby na podobné úrovni znalostí nebo jsou ze stejného oddělení, může být skupinové setkání produktivnější. S tím souvisí i nevýhody techniky, které můžeme pozorovat při přípravách na setkání z důvodu jejich větší náročnosti anebo skutečnost, že je obtížné udržet setkání zaměřené na dané téma [18].

Workshopy

Workshopů se může zúčastnit 6-10 uživatelů nebo i více, a to za účelem získání požadavků. Workshopy mají spíše definovanou dobu trvání než cíl a mohou být opakovány za účelem objasnění již získaných požadavků nebo pro získání větších podrobností. Jedná se o rychlejší techniku, než jakou jsou skupinové rozhovory, využívá se zejména pro běžné nebo systémové požadavky. Na druhou stranu je zde nutné více příprav, vedení této techniky vyžaduje více dovedností a mohou zde být zapotřebí osoby z oblasti IT, které požadavky nebo jejich detaily zaznamenávají. Z počátku se také může stát, že konverzace není plynulá [18].

Brainstorming

Brainstorming lze provádět jak individuálně, tak i formou skupin. Jedná se o shromáždění myšlenek, které mohou být dále analyzovány a případně zahrnuty i do samotných požadavků na systém. Myšlenky mohou pocházet z toho, co uživatelé viděli u profesně zkušenějších uživatelů nebo také ze systémů, se kterými pracovali v předchozích zaměstnáních. Může se jednat o velmi efektivní způsob, jak mohou uživatelé definovat své požadavky, nebo dokonce v rámci brainstormingu můžeme přijít na velmi inovativní nápady a požadavky. Pro některé uživatele může být nevýhodou, že je pro ně těžké přemýšlet v rychlosti a přijít na nové nápady [18].

Studie proveditelnosti

Studie proveditelnosti nebo jiné nedávné studie stávajících systémů mohou poskytnout přehled o základních požadavcích. Protože se jedná o nedávné, ale hlavně také zdokumentované informace, přinášejí nám relevantní údaje pro specifikaci požadavků. Tyto studie nejsou ale ve většině případů příliš podrobné, proto mohou být nedostatečné pro naše potřeby [18].

Aktuální systémová dokumentace

Za předpokladu, že máme nějaký stávající IS, můžeme jako vstupy požadavků na systém nový využít již existující dokumentaci starého systému. Takováto dokumentace může obsahovat podrobnosti o rozhraní, uživatelské příručky a příručky dodavatele softwaru. Největší výhodou je spousta potenciálních informací a jejich snadný přenos do nového dokumentu o požadavcích na systém. Rizikem v tomto případě je zastaralost dokumentace, která nemusí odpovídat novým potřebám a požadavkům. Proto je důležité si dokumentaci pečlivě pročíst a posoudit, zda odráží skutečnost, kterou potřebujeme [18].

Dotazníky

Dotazníky mohou být užitečné pro získání informací o systémových požadavcích od uživatelů, kteří často se systémem nepracují nebo od uživatelů, kteří jsou geograficky vzdáleni. Návrh samotného dotazníku a stanovení otázek vyžaduje péči, protože špatným sestavením může dojít k ovlivnění odpovědí, a tím k získání irelevantních informací. Výhodou je, že se jedná o velmi levnou techniku, kterou lze praktikovat na velmi vysokém počtu uživatelů. Jedná se o efektivní způsob získávání informací od uživatelů, kteří se nenacházejí v místě tvorby systému. Jelikož se jedná o písemné odpovědi, přináší to výhodu v podobě usnadnění práce a času při zpracování a analýze. I přes tyto nezpochybnitelné výhody existuje i několik nevýhod. Mezi hlavní lze zařadit fakt, že dotazníky mohou být pomalé a nemusíme dostat dobrou odezvu, protože se často stává, že uživatelé neradi dotazníky vyplňují [18].

Případy užití

Případy užití popisují, jak fungují určité procesy, kterých se týkají systémové požadavky. Popisují systém z pohledu uživatele a pro některé z nich může být jednodušší popsat, s čím a jak pracují než specifikovat požadavky. Tyto procesy je ale potřeba ještě analyzovat a zdokumentovat v požadavcích. Je potřeba se těmto procesům věnovat a v případě, že se zaměřujeme na stávající procesy, nemusí být informace tak důležité [18].

Pozorování lidí

Pozorování uživatelů nebo části jejich práce může poskytnout informace o existujících procesech, vstupech a výstupech. Jedná se však o velmi pomalý postup, který je zaměřený spíše na existující procesy než na procesy nové. Na druhou stranu je tato technika užitečná, pokud uživatel není schopen jasně vysvětlit, jaký je přesný popis jeho práce nebo není schopen specifikovat své požadavky na systém nový. Z tohoto pozorování lze v případě schopného analytika vyčíst nápady pro zlepšení procesů nebo odstranění zbytečných činností v novém systému [18].

Prototyping

Prototypování zahrnuje shromažďování požadavků a jejich použití při tvorbě prototypu. To umožňuje uživatelům vidět potenciální řešení a získat lepší cit pro to, co od nového systému vyžadují. Na základě prototypu pak uživatelé přidávají nebo upravují své požadavky, kdy lze prototyp postupně přepracovávat, dokud nejsou veškeré požadavky splněny. Mezi hlavní výhody této metody bezpochyby patří možnost ověřit si, jak by mohl fungovat určitý požadavek na softwarovou funkci nebo jedinečný SW na míru v praxi. Dále také dokážeme identifikovat problémy s požadavky a můžeme zlepšit kvalitu požadavků a tím i konečný systém. Prototypování není vhodné pro velké IS, dále není vhodné pro počáteční identifikaci požadavků, kdy na začátku samotné identifikace může docházet k velkým chybám. V neposlední řadě je také nutné si uvědomit, že se jedná o velmi nákladnou techniku [18].

3.2 Hledání chybějících požadavků

Požadavky, které nebyly v rámci sběru definovány z důvodu jejich opomenutí, jsou nejčastějším typem chyb. Při přezkoumávání dokumentací a zápisů je ale nenalezneme, protože tam chybí. Z tohoto důvodu je pro hledání doposud přehlížených požadavků nutné využít nějaké jiné techniky [28].

Základní technikou je rozvedení obecných požadavků do větších podrobností, aby bylo všem jasné, co přesně dané požadavky znamenají. Pokud jsou stanovené požadavky příliš obecné a neurčité, nechávají tím příliš mnoho prostoru čtenářově fantazii a tím se můžeme dostat do situace, kdy se výsledná implementace vývojáři může značně lišit od představ samotného zákazníka. Dále je také potřebné si ověřit, že jsme zdárně obdrželi požadavky od všech uživatelů, a že u každého případu užití máme alespoň jednoho aktéra [28].

Velmi důležité je rovněž nutnost pohlídat si tzv. hraniční případy. Příkladem takového případu může být situace, kdy požadavky stanovují, že pokud je teplota nižší než 20 °C kontrolka svítí

zeleně a pokud je teplota vyšší než 20 °C kontrolka svítí červeně. V tomto případě nevíme, co má kontrolka dělat v případě, že teplota dosahuje přesně 20 °C, tento požadavek nám chybí [28].

Při hledání chybějících požadavků se také využívají grafické zápisy požadavků již zdokumentovaných, protože textová dokumentace je méně přehledná. Při zkoumání grafického modelu si lze snadno povšimnout např. chybějící šipky od jednoho obdélníku ke druhému, tato chybějící šipka nám ukazuje na chybějící požadavek [28].

3.2.1 Diagram datových toků (DFD)

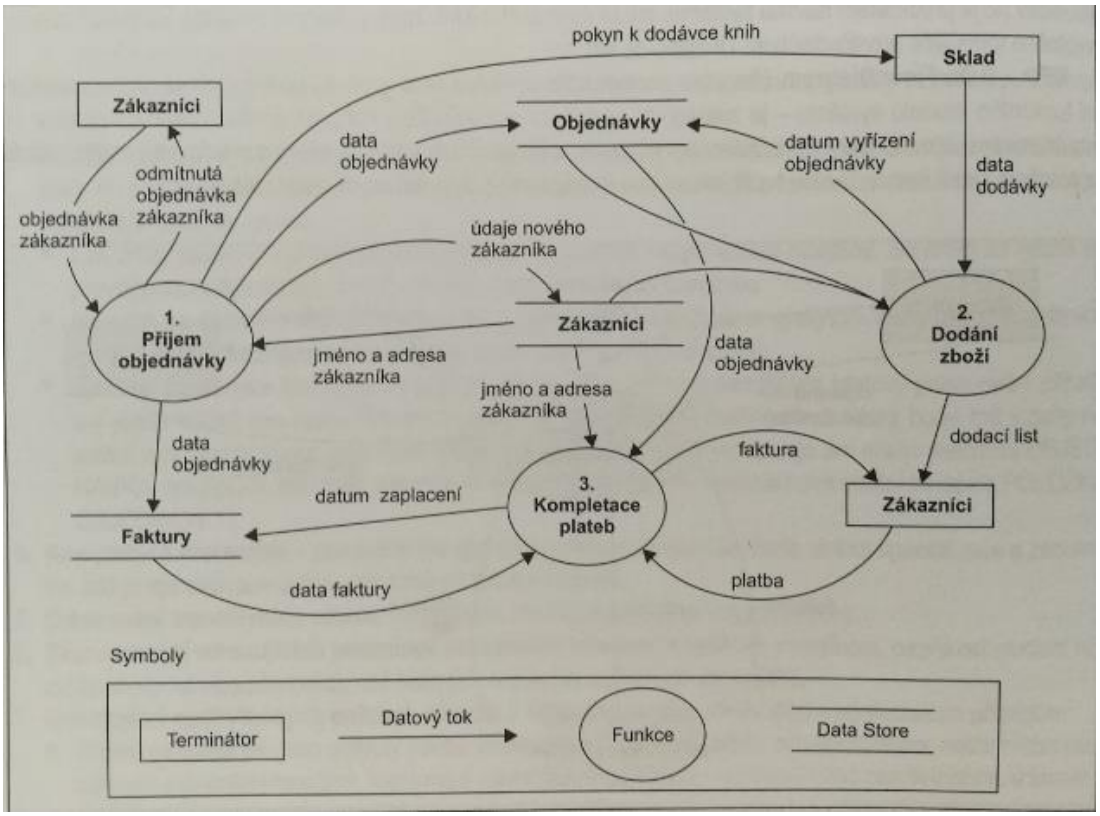
Diagram datových toků je často využívaným nástrojem např. v agilním modelování. Základní výhodou tohoto diagramu je schopnost vyjádřit datové toky mezi systémem a jeho okolím a mezi samostatnými funkcemi příslušného systému navzájem. DFD dále také umožňuje systémem uchovávaná data vyjádřit [6].

DFD je grafickým prostředkem pro vyjádření návrhu a zobrazení funkčního modelu systému. Funkční model popisuje funkce, které mají tvořit IS, aby se stal věrným modelem reality, která se skládá z funkcí a jejich vstupů a výstupů. Při tvorbě tohoto diagramu se využívají čtyři základní prvky [6]:

- funkce – funkce je informační proces (zpracování dat), jehož pomocí modelujeme reálné dění, znázorňuje transformaci vstupu na požadovaný výstup. Značí se pomocí elipsy, musí mít název a být označena jedinečným ID, které se skládá z ID nadřazené funkce a přiděleného čísla v rámci dané úrovně ve které se nachází,
    - Datová funkce – hlavním úkolem je produkce výstupů za pomocí zpracování vstupních dat
    - Řídící funkce – algoritmus řízení procesů v nějaké části systému, nezpracovává žádná data, slouží k zachycení charakteristik systému v reálném čase
- datový tok – znázorňuje se šipkou, která symbolizuje přesun dat mezi jednotlivými částmi systému nebo mezi systémem a okolím. Datové toky obsahují data, která systém ukládá nebo zpracovává. Název datového toku má za úkol přesně vystihovat přenášený obsah,
- datové úložiště – datové úložiště symbolizuje místo, kde se patřičná data uchovávají. Znázorňuje se pomocí dvou rovnoběžných čar, mezi kterými se nachází název samotného úložiště, který by měl být uváděn v množném čísle. Každé datové úložiště

musí mít nejméně jeden datový tok dovnitř a jeden ven a data musí vždy proudit přes funkci,

- terminátor – pomocí obdélníku znázorňuje objekty, které jsou součástí okolí systému, a ne jeho přímou součástí, tím myslíme externí zdroj dat nebo místo jejich určení. Jedná se tedy o objekt v okolí IS, se kterým systém komunikuje. Jedná se především o lidi nebo jiné systémy.



Obrázek 3: Diagram datových toků

Zdroj: [6]

Pravidla tvorby DFD

Při vytváření diagramu datových toků existují určitá pravidla, která musí být dodržována. Tato pravidla umožňují, aby DFD byl smysluplný a srozumitelný. Mezi základní pravidla patří [20]:

- veškeré datové toky musí proudit buď z funkce, nebo do funkce, datový tok nemůže přímo propojit datové úložiště a terminátory, terminátory spolu nemohou přímo komunikovat, data se nemohou přesouvat mezi datovými úložišti bez předchozího zpracování,

- proces musí mít alespoň jeden vstupní tok a jeden výstupní tok. Když má proces vstupní tok, ale žádný výstupní tok, nazývá se „černá díra“. Když má proces výstupní toky, ale žádné vstupní toky, nazývá se „zázrak“,
- vstupy do procesu musí být dostatečné k produkci výstupních toků. “Šedá díra” je, když výstupy procesu jsou větší než součet jeho vstupů. Pokud je například jméno a adresa zákazníka vstupem, jejich bankovní údaje nemohou být výstupem, protože proces nemá k dispozici dostatek informací,
- procesy musí transformovat data. Při pojmenování datových toků by měla být použita adjektiva, která ukazují, jak zpracování dat změnilo tok dat,
- toky dat se nemohou navzájem křížit. K překonání tohoto problému lze duplikovat datové úložiště a entity. Procesy však nelze duplikovat. Datové toky musí být jednosměrné,
- subjekty musí být označeny malými písmeny.

3.2.2 Případy užití

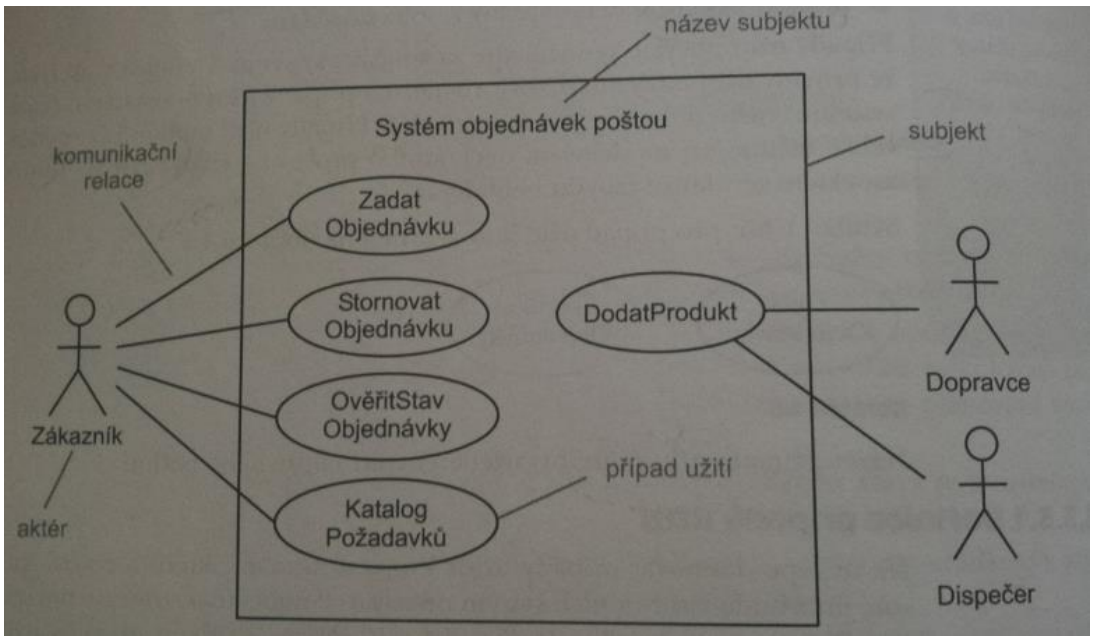
Případy užití zachycují funkčnost informačního systému a tím vymezují rozsah prací. Každý z případů užití popisuje právě jednu požadovanou funkci, tedy způsob využití IS. Funkce, která není popsána žádným takovým případem užití, nebude v systému implementována [13].

Případy užití definují interakce mezi aktéry a systémem k dosažení konkrétních cílů. Existují tři základní prvky, které tvoří případ užití [39]:

- Aktér – uživatel, který se systémem pracuje,
- Systém – případy zachycují funkční požadavky, které určují zamýšlené chování systému,
- Cíle – případy použití jsou obvykle iniciovány uživatelem, aby splnily aktivity a varianty, které mají být zahrnuty při dosahování cíle.

Diagram případu užití

V diagramu případu užití znázorňujeme pomocí rámečku hranice systému, který je modelován pomocí případů užití. Aktéry znázorňujeme symbolem postavičky, pod kterou je její název a umisťujeme je mimo hranice systému. Samotné případy užití umisťujeme uvnitř hranic systému a znázorňujeme je pomocí elipsy, ve které je uveden název daného případu. Vztah mezi jednotlivými případy užití a aktéry jsou znázorněny pomocí plné čáry (v jazyce UML symbol přiřazení) [3].

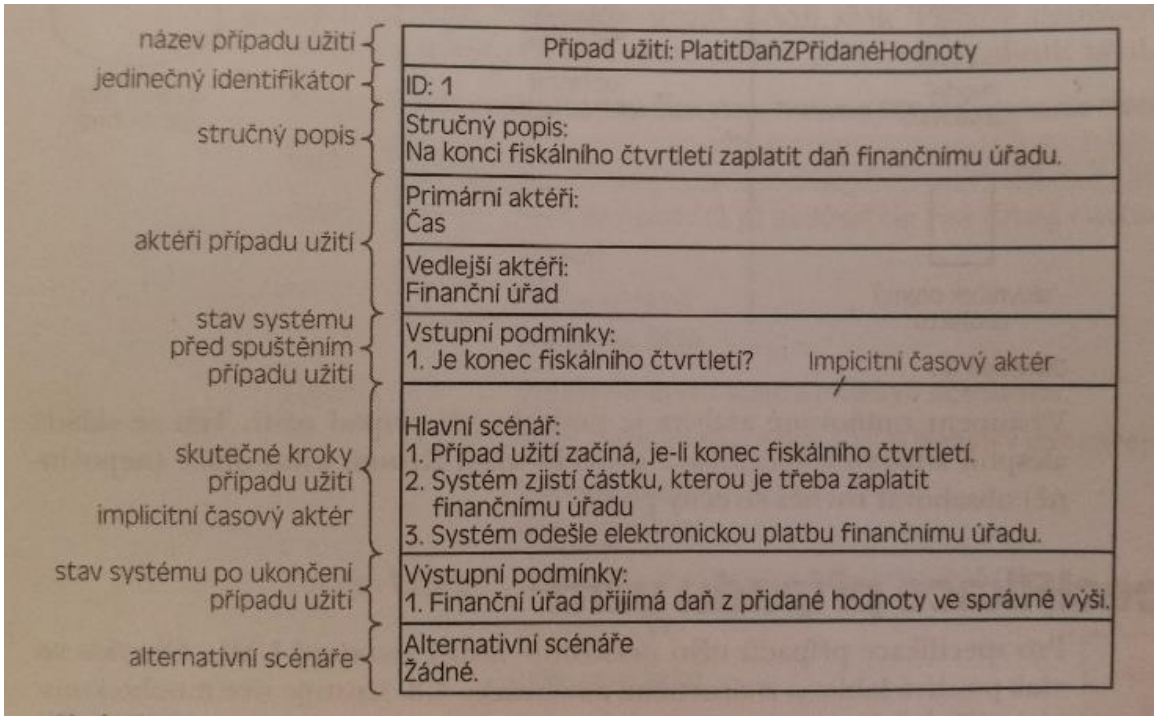


Obrázek 4: Diagram případu užití

Zdroj: [3]

Scénáře

Scénář je do detailů rozpracovaný průběh části funkčnosti daného IS. Jedná se o posloupnost činností, které vedou k naplnění dané funkčnosti. Výstupem správného scénáře je pro uživatele zisk dané funkce. Každý případ užití může být popsán více různými scénáři, které mohou zachycovat i alternativní stavy systému, popřípadě stavů chybových [22].



Obrázek 5: Scénář případu užití

Zdroj: [3]

3.2.3 CRUD matice

Matice CRUD (Create, Read, Update, Delete) je tabulka, která ukazuje vazby mezi procesy a daty nebo mezi procesy a zdroji. Pokud existuje odkaz, ukazuje, zda proces provádí operaci vytváření, čtení, změny nebo mazání na datech nebo zdroji [37]. Mimo těchto čtyř základních operací někteří lidé uvádějí navíc k této matici také písmeno L z anglického slova List, což znamená výpis daných dat [28].

Když matice CRUD ukazuje vazby mezi procesem a daty, nazývá se datovou CRUD maticí, pokud ale ukazuje vazby mezi procesem a zdrojem, nazývá se CRUD maticí zdrojů. Data, která jsou uložena a uchovávána v testovaném systému, mají životní cyklus. Ten začíná, když je entita vytvořena a končí, když je odstraněna. Mezitím je entita aktualizována nebo čtena. Pomocí této matice můžeme snadno získat přehled o životním cyklu dat nebo entit. V této matici se na vodorovné ose (ve sloupcích) zobrazují entity a na vertikální ose (v řádcích) se zobrazují funkce/případy užití [38].

Následně se do matice postupně vyplňují jednotlivá písmena C, R, U, D, případně L, jestliže funkce/případ užití vykonává danou akci ve spojení s entitou [38].

entita / případ užití	Objednávka	Chemikálie	Zásobovač	Dodavatelský katalog
Vytvoření objednávky	C	R	R	R, L
Změna objednávky	U, D		R	R, L
Správa chemického inventáře		C, U, D		
Výpis seznamu objednávek	R	R, L	R, L	
Změna Zásobovače				

Obrázek 6: Příklad CRUD(L) matice – systém pro evidenci chemikálií

Zdroj: [28]

Když je matice hotová, podíváme se na vypsaná písmena u každé entity a v případě, že v některém sloupci nějaké písmeno chybí, je potřeba se zamyslet, jestli opravdu danou činnost s entitou provádět nepotřebujeme, nebo jsme daný požadavek při sběru zapomněli [28].

4 VYHODNOCENÍ POŽADAVKŮ

Velké softwarové systémy mohou mít několik set až tisíce požadavků. Ne všechny zdokumentované požadavky je ale implementační tým schopen realizovat. Existuje několik omezení, jako jsou omezené zdroje, rozpočtová omezení, časová krize, proveditelnost atd., které je třeba upřednostnit [23].

Dobře provedené rozdělení požadavků dle priorit je nezbytné pro všechny dobře řízené projekty. Zajišťuje, že se projekt zaměřuje nejprve na nejdůležitější prvky, a že každý chápe a souhlasí s tím, co jsou nejdůležitější prvky systému. Dobré stanovení priorit také zajistí, že inženýři, programátoři a databázoví analytici budou vyvíjet nejdůležitější prvky projektu v synchronizaci s obchodními potřebami [43].

Technik pro vyhodnocení softwarových požadavků existuje mnoho. Článek v odborném periodiku Information and Software Technology identifikuje celkem 49 technik pro prioritizaci softwarových požadavků [29].

V této kapitole si postupně představíme vybrané techniky, jejichž pomocí můžeme naše požadavky rozdělit. Jako první bude představena metoda MoSCoW, kterou jsem vybral z důvodu její jednoduchosti při samotné tvorbě. Další metodou bude model KANO, ten jsem vybral z důvodu, že jeho pomocí lze rozdělit požadavky dle jejich priority, ale také lze identifikovat nerelevantní požadavky. Třetí zvolenou metodou je bublinkové řazení, technika byla vybrána z důvodu, že se jedná o nejjednodušší metodu využívající algoritmus. Poslední metodou, zvolenou jako zástupce komplexnějších metod, je Rozpracování funkcí kvality (QFD). Na závěr kapitoly budou vybrané metody porovnány.

4.1 Metoda MoSCoW

Jako první zvolenou metodu si představíme metodu MoSCoW. Princip této metody spočívá v jasném rozdělení požadavků do jednotlivých kategorií v závislosti dle jejich důležitosti pro konečný systém. Název této prioritizační metody je odvozen z jednotlivých názvů kategorií, do kterých požadavky dělíme [41].

Must have (musí mít)

Jak název napovídá, tato kategorie se skládá z iniciativ, které jsou pro náš tým „nutností“. Představují potřeby pro daný projekt nebo produkt, o kterých se nevyjednává. Pokud například vydáváte aplikaci pro zdravotní péči, iniciativou, která musí mít nutnost, mohou být bezpečnostní funkce, které pomáhají udržovat shodu. Cokoliv v kategorii „musí mít“ je pro tým

považováno za povinné. Pokud si nejsme jisti, zda něco patří do této kategorie, můžeme se sami sebe zeptat na následující otázky [16]:

- Co se stane, když spustíme systém bez dané funkce (požadavku)?
- Existuje jiné řešení nebo jednodušší způsob, jak toho dosáhnout?
- Funguje systém bez tohoto požadavku?

Pokud produkt nebude fungovat bez takového požadavku nebo se jeho vypuštěním stane zbytečným, požadavek je s největší pravděpodobností nutností [16].

Should have (měl by mít)

Tyto požadavky by měly být jen o krok níže než požadavky nutné. Jsou důležité pro konečný software, ale nejsou životně důležité. Pokud IS nebo jiný produkt nebude takovými požadavky disponovat, bude stále funkční. Pokud jsou však tyto požadavky zahrnuty, přidávají systému významnou hodnotu [16].

Požadavky „měl by mít“ se liší od požadavků „musí mít“, protože mohou být připraveny na budoucí vydání, aniž by to mělo dopad na současnost. Například zlepšení výkonu, drobné opravy chyb nebo nové funkce mohou být v této kategorii [16].

Could have (mohl by mít)

Jedná se o požadavky, které budou implementovány pouze v případě, že máme dostatek času a zdrojů. Zařazení těchto požadavků do konečného systému není vůbec povinné, ale jedná se o velmi vítané řešení [4].

Ve srovnání s požadavky předchozí kategorie mají mnohem menší dopad na výsledek, pokud budou vynechány. Požadavky, které jsou umístěny v kategorii „mohl by mít“, jsou často první, které mají být vynechány, pokud jsou první dvě kategorie větší, než se očekávalo [16].

Won´t have this time (nebude mít v daném čase)

Jednou z výhod metody MoSCoW je, že zařazuje některé požadavky do kategorie „nebude mít v daném čase“. To nám pomáhá zvládnout očekávání o tom, co (ne)bude zahrnuto do konkrétní verze našeho IS. Jedním ze způsobů, jak zabránit šíření rozsahu, je umístění požadavků v této kategorii. Pokud jsou potřeby zařazeny v této kategorii, tým ví, že pro tento konkrétní časový rámec nemají být prioritou. Některé požadavky v této skupině budou mít v budoucnu prioritu, zatímco jiné se pravděpodobně nebudou implementovat vůbec [16].

Tato technika je skvělý způsob, jak sestavit rozhovory s klienty, porozumět tomu, co chtějí, a co je pro ně dobré v systému mít. Tímto způsobem neztrácíte čas a je to výhodné pro obě strany [8].

Písmena „O“ jsou v názvu metody přidána, aby byla zkratka dobře vyslovitelná, a jsou často psaná malým písmem, aby bylo zřejmé, že nic neoznačují. Na rozdíl od systému číslování pro nastavení priorit slova něco znamenají a usnadňují diskusi o tom, co je důležité. Nutné požadavky musí poskytnout ucelené řešení a samy o sobě vést k úspěchu projektu [45].

Priorita MoSCoW je efektivní metodou stanovení priorit pro týmy, které by chtěly do svého procesu zapojit zástupce celé organizace. Zahrnutím účastníků z různých funkčních oddělení lze zachytit širší perspektivu. Dalším důvodem, proč je dobré použít tuto metodu, je umožnění danému týmu zjistit, kolik úsilí jde do každé kategorie [16].

Tipy pro úspěšnou MoSCoW prioritizaci [41]:

- Celý tým by měl jasně definovat, co znamenají jednotlivé kategorie pro daný produkt
- Využívat veškeré kategorie
- Je dobré spravovat celkový rozsah povinných požadavků
- Pokud je to možné, provádět tuto prioritizaci co nejvíce vizuálně
- Celý tým by měl diskutovat o tom, jak jim technika vyhovuje a jak jsou spokojeni s jednotlivými kritérii

4.2 Kano model

Model Kano vyvinul v 80. letech japonský pedagog, lektor, spisovatel a konzultant v oblasti managementu Noriaki Kano. Tento model umožňuje klasifikovat funkce produktu v závislosti na hodnotě, kterou poskytují svým uživatelům [25].

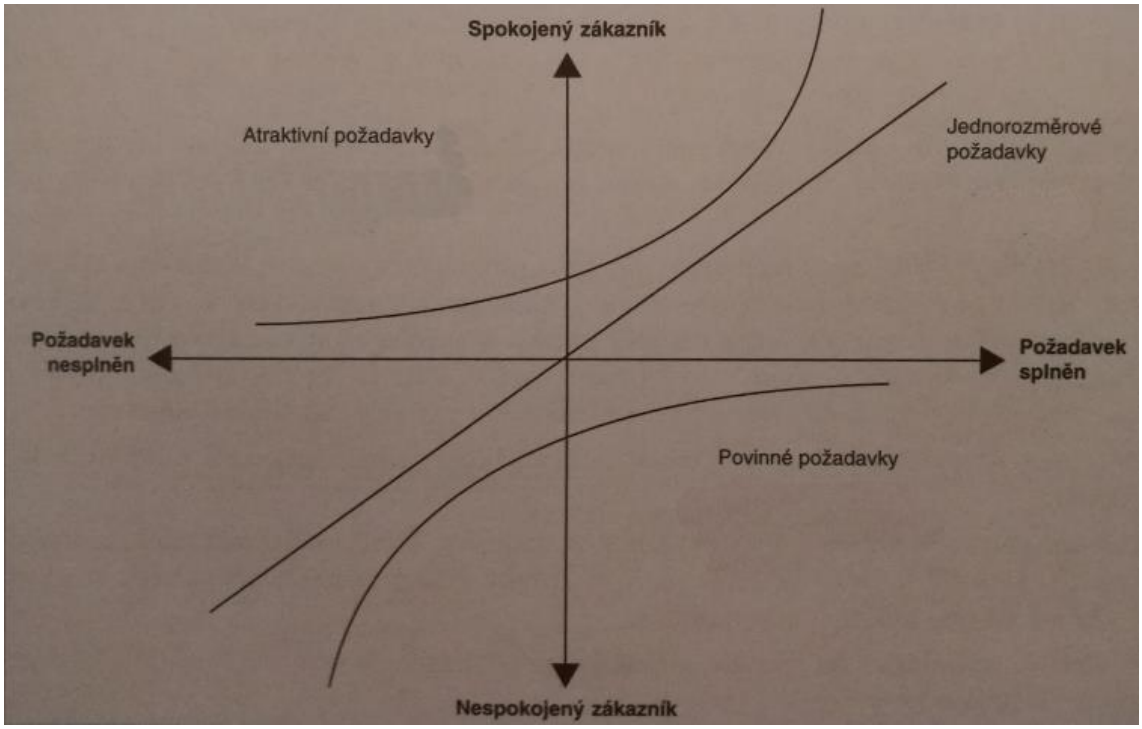
Kano říká, že produkt nebo služba je mnohem více než jen funkčnost. Je to také o emocích zákazníků. Například všichni zákazníci, kteří si koupí nové auto, očekávají, že se zastaví, když šlápnou na brzdy, ale mnozí z nich budou potěšeni jeho hlasově aktivovaným systémem asistence při parkování [14].

Profesor klasifikoval 5 unikátních kategorií požadavků zákazníků, z nichž 3 by měly skončit v konečné nabídce a další 2 by měly být vyřazeny. Hlavním cílem modelu Kano je pomoci týmům porozumět, klasifikovat a integrovat tyto 3 hlavní kategorie požadavků do produktů nebo služeb, které vyvíjejí. 5 kategorií požadavků zákazníků je klasifikováno v závislosti na jejich schopnosti vytvářet spokojenost zákazníků nebo způsobit jejich nespokojenost. Vědět, do jaké

kategorie zákazníků požadavky spadají a důležitost každého požadavku může pomoci upřednostnit rozvojové aktivity a určit, co zahrnout do naší nabídky a jak nejlépe utratit zdroje, které by zlepšily tyto požadavky [44].

V závislosti na funkčnosti a spokojenosti zákazníka jsou požadavky umístěny v jedné z následujících kategorií [25]:

- Musí být (Must-Be)
- Jednorozměrné (One-Dimensional)
- Atraktivní (Attractive)
- Neutrální (Indifferent)
- Přesně opačné (Reverse)



Obrázek 7: Kano model spokojenosti zákazníků

Zdroj: upraveno [12]

Must-Be

Jak již název napovídá, povinné požadavky jsou takové, které uživatelé považují za nezbytné, aby produkt jako celek fungoval podle očekávání. Povinné požadavky musí být přítomny, nebo produkt nebude fungovat a v očích zákazníka nebude mít žádnou hodnotu. Mezi příklady takových požadavků (funkcí) můžeme uvést [25]:

- Fungující volant v autě

- Možnost telefonovat na novém smartphonu
- Knoflíky na knoflíkové košili a další

Pokud jsou takovéto požadavky splněny, určitě nás to neudělá šťastnými, ale pokud naopak splněny nejsou, zajisté nás to rozzlobí. Na obr. 7 si můžeme všimnout, jak se chová křivka spokojenosti povinných požadavků. Dokonce i sebemenší kousek investic vede ke zvyšování spokojenosti. Ale také si lze všimnout, že spokojenost nikdy nedosahuje ani pozitivní reakce. Bez ohledu na to, co do této funkce investujeme, zákazníci nebudou spokojeni. Dobrou zprávou je, že jakmile dosáhneme základní úrovně očekávání, nemusíme do takových funkcí nadále investovat. [24].

One-Dimensional

Jak ukazuje obrázek 7, míra spokojenosti, kterou poskytuje jednorozměrná funkce, přímo souvisí s úrovní funkčnosti uvedeného prvku. Pro další objasnění těchto požadavků můžeme použít příklady takových funkcí, jako jsou [25]:

- Spotřeba vozidla
- Velikost pevného disku smartphonu
- Rychlost zpracování operací počítače nebo notebooku

Každý z těchto příkladů jsou případy „čím více, tím lépe“ (na rozdíl od „buď / nebo“, jak je tomu u povinných funkcí). Obecně řečeno, čím více kilometrů řidič ujede za stejný objem paliva, nebo čím více místa bude na pevném disku smartphonu, tím vyšší bude úroveň spokojenosti zákazníka [25].

Attractive

Jedná se o rys, který by produkt mohl mít, který bude pro zákazníka trochu bonusem nebo příjemným překvapením, ale nezměnil by úroveň spokojenosti zákazníka, kdyby nebyl zahrnut [2].

Jako příklady atraktivních požadavků lze uvést [25]:

- Ukazatel spotřeby na palubní desce automobilu
- Smartphone s možností bezdrátového nabíjení
- Bezplatná dodávka od online prodejce

Indifferent

Neutrální požadavky (nebo funkce) jsou takové, které nejsou kategoricky ani špatné, ani dobré, a nezaručují ani spokojenost zákazníka, ani jeho nespokojenost [46].

Příkladem indiferentních požadavků můžou být [25]:

- Zdali-je plynová nádrž v automobilu umístěna na levé nebo pravé straně
- Typ plastu láhve džusu
- Barva krytu tiskárny

Reverse

Jedná se o přesně opačné požadavky (funkce) než jsou funkce jednorozměrné. Jinými slovy, se jedná o funkce takové, které ve skutečnosti snižují úroveň spokojenosti uživatele, protože zvyšují funkčnost produktu [25].

Příklady přesně opačných funkcí [25]:

- Příliš mnoho tlačítek na volantu, které vede k rozptýlení řidiče
- Pokročilé funkce SW, které jsou pro průměrné uživatele příliš komplikované

Postup tvorby KANO modelu má 4 základní kroky [12]:

1) Identifikace požadavků

   V rámci zjištění jednotlivých požadavků jsou ve většině případů využity dotazníky. Abychom úspěšně nalezli veškeré požadavky z hlediska spokojenosti zákazníků, lze použít následující otázky.

   „Jaké se Vám vybaví asociace, když používáte produkt X?“

   „Jaké problémy/závady si vybavíte v souvislosti v produktem X?“

   „Jaká kritéria berete v úvahu při zvažování nákupu produktu X?“

   „Jaké nové vlastnosti nebo služby byste přivítali v souvislosti s produktem X? Co byste na produktu X změnili?“

   Výsledek položení první otázky nám bohužel nepřinese konkrétní požadavky na daný produkt, protože bývají mlhavé. Mohou nám ale dobře posloužit jako inspirace pro budoucí inovace produktu. Pomocí druhé otázky dojdeme k přáním zákazníků a zjištění problémů, které jsou s produktem spojeny.

Odpovědi na třetí otázku bývají většinou jednorozměrové. Poslední čtvrtá otázka má za úkol určení požadavků, které jsou zajímavé pro zákazníky, ale samotní dodavatelé o nich nevědí.

2) Tvorba Kano dotazníku

Ve druhém kroku se dříve sesbírané požadavky rozdělují a zařazují do výše popsaných skupin. Princip spočívá v položení dvou otázek ke každému požadavku, kdy první otázka ukazuje reakci zákazníka v případě, že daný požadavek splněný je, a naopak druhá otázka sleduje reakci v případě, že splněn není.

Odpovědi na obě otázky by měly být podány formou jednoho z 5 stupňů od silného souhlasu přes neutrální reakci až po silný nesouhlas. Samotné zařazení probíhá pomocí klasifikační tabulky, ze které lze snadno vyčíst, o jaký typ požadavku se jedná (viz. tabulka 3).

Tabulka 3: Vyhodnocení zákaznických požadavků dle KANO modelu

Požadavky zákazníků		Negativně koncipovaná otázka				
		1. silný souhlas	2. musí být splněn	3. neutrální	4. lze to vydržet	5. silný nesouhlas
Pozitivně koncipovaná otázka	1. silný souhlas	Q	A	A	A	O
	2. musí být splněn	R	I	I	I	M
	3. neutrální	R	I	I	I	M
	4. lze to vydržet	R	I	I	I	M
	5. silný nesouhlas	R	R	R	R	Q

Zdroj: zpracováno dle [12]

3) Dotazníková šetření

V tomto kroku je hlavní správně rozhodnout a vybrat jakou formou budeme dotazování provádět. Jednou z nejlevnějších metod je použití poštovní distribuce, které má ale velkou nevýhodu v poměru získaných informací. Na druhou stranu nejúčinnějším způsobem je osobní dotazování či telefonické rozhovory.

4) Vyhodnocení a vyjádření

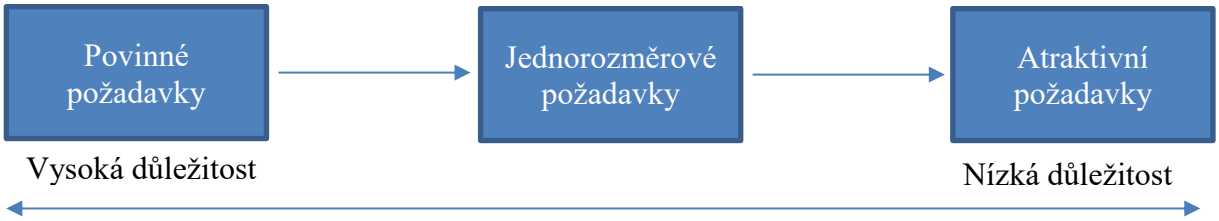
Při samotném vyhodnocení nejdříve vypracujeme tabulku, jejíž pomocí jednoduše určíme výslednou kategorii požadavků:

Tabulka 4: Kategorizace požadavků – příklad

Požadavek	A	O	M	I	R	Q	Součet	Kategorie
Pož. 1	8	22	**53**	7	0	10	100 %	M
Pož. 2	12	**48**	28	6	2	4	100 %	O
Pož. 3	**55**	20	10	12	0,5	2,5	100 %	A

Zdroj: přepracováno dle [12]

Čísla v tabulce znamenají procento zákazníků, dle toho, jak odpovídali ve druhém kroku. Současně lze vyhodnotit zákaznickou spokojenost s plněním požadavků pro daný produkt. Po těchto krocích následuje rozhodnutí, na které požadavky je vhodné se zaměřit nejdříve. Následující obrázek vyjadřuje posloupnost důležitosti tří základních kategorií.



Obrázek 8: Důležitost základních kategorií

Zdroj: zpracováno dle [10]

4.3 Bublinkové řazení

Bublinkové řazení je třídící algoritmus, který porovnává dvě sousední položky najednou, počínaje počátkem seznamu, a vyměňuje je, pokud jsou mimo pořadí. Cílem je tedy seřadit čísla od nejmenšího po nejvyšší. Každé srovnání postupně posouvá položku s nejvyšším číslem na konec seznamu [53].

Chceme-li řadit požadavky dle jejich priority pomocí této metody, vezmeme dva požadavky a porovnáme je. Pokud zjistíme, že jeden požadavek má větší prioritu než druhý, odpovídajícím způsobem je vyměníme. Tímto způsobem pokračujeme dále, dokud nedojde k řádnému setřídění do posledního požadavku. Výsledkem je seznam požadavků, které jsou seřazeny dle jejich priority [30].

Výhody [31]:

- efektivní metoda při malém množství položek v seznamu
- jednoduchá implementace
- efektivní využívání paměti
- výsledkem je setříděný seznam

Nevýhody [31]:

- časově náročné pro velké množství požadavků (položek seznamu) z důvodu, že s přibývajícími požadavky čas roste exponenciálně

4.4 QFD

Mnoho úspěšných organizací shromažďuje a integruje Hlas zákazníka (VOC) do designu a výroby svých výrobků. Aktivně navrhují kvalitu a hodnotu vnímanou zákazníkem do svých výrobků a služeb. Tyto společnosti využívají strukturovaný proces k definování přání a potřeb svých zákazníků a jejich transformaci do konkrétních návrhů produktů a výrobních plánů na výrobu produktů, které uspokojí potřeby zákazníka. Proces nebo nástroj, který používají, se nazývá QFD (Quality Function Deployment) [48].

QFD bylo vyvinuto tak, aby přineslo osobní rozhraní do moderní výrobní a obchodní činnosti. V dnešní průmyslové společnosti, kde vzrůstá vzdálenost mezi výrobci a koncovými uživateli, QFD spojuje potřeby zákazníka s konstrukčními, vývojovými, technickými, výrobními a servisními funkcemi [49].

Jedná se o přeměnu uživatelských požadavků do návrhů výrobků. Cílem QFD je vybudovat produkt, který dělá přesně to, co zákazník chce, místo toho, aby dodal produkt, který zdůrazňuje odbornost, kterou již uživatel má [17]. Dále se pomocí soustavy matic s korelační závislostí využívá pro analýzu a hodnocení vazeb mezi zákaznickými požadavky a samotnými vlastnostmi produktu [47].

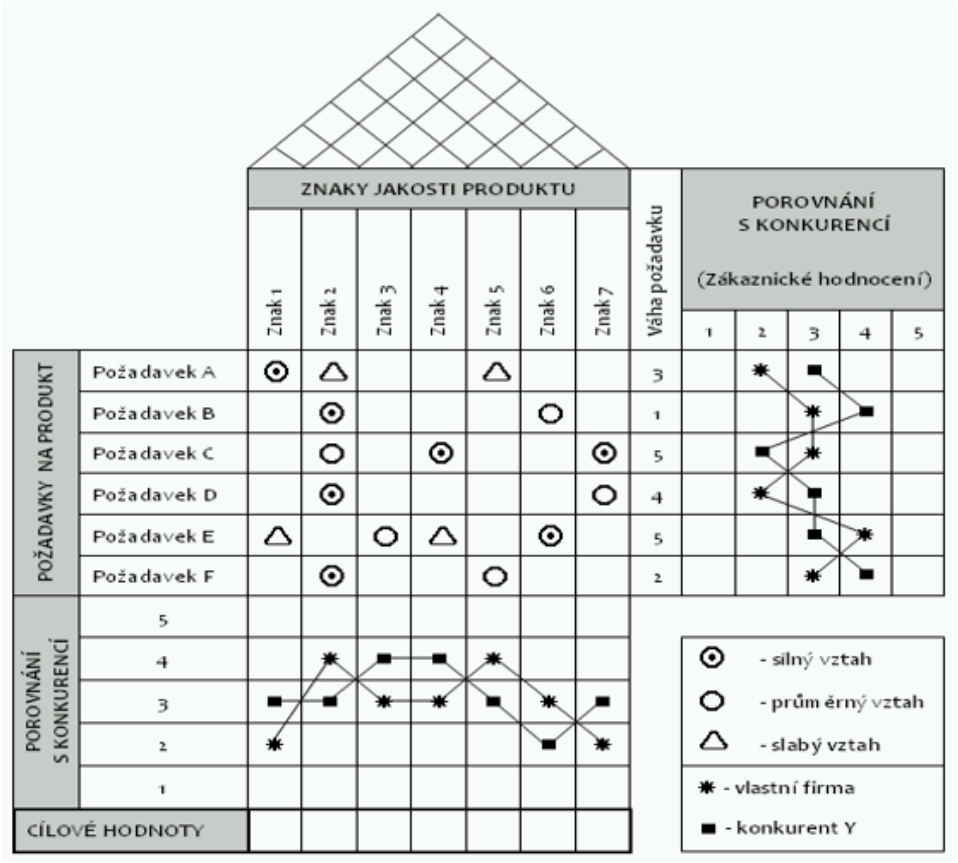
QFD metodika

Metodika implementace QFD zahrnuje čtyři základní fáze, které probíhají v průběhu procesu vývoje produktu. Během každé fáze je připravena jedna nebo více matic, které pomáhají plánovat a řešit důležité informace o výrobcích a procesech a návrhu [7].

Čtyři fáze QFD jsou [11]:

- Plánování produktového konceptu. Začíná se zákazníky a průzkumem trhu vedoucího k produktovým plánům, nápadům, náčrtkům, koncepčním modelům a marketingovým plánům.
- Vývoj a specifikace výrobku. Vede k vývoji prototypů a testům.
- Výrobní procesy a výrobní nástroje. Ty jsou navrženy na základě specifikace produktu a jeho součástí.
- Výroba výrobku. Začíná po vyřešení pilotního produktu.

V QFD je kvalita znázorněna mírou spokojenosti zákazníka s produktem nebo službou. Tvoříme matici, běžně nazývanou dům jakosti (obr. 10), kde se metodika QFD zaměřuje na nejdůležitější atributy nebo vlastnosti produktu nebo služby [27].



Obrázek 9: Dům jakosti

Zdroj: [42]

Postup zpracování „Domu jakosti“ [42]:

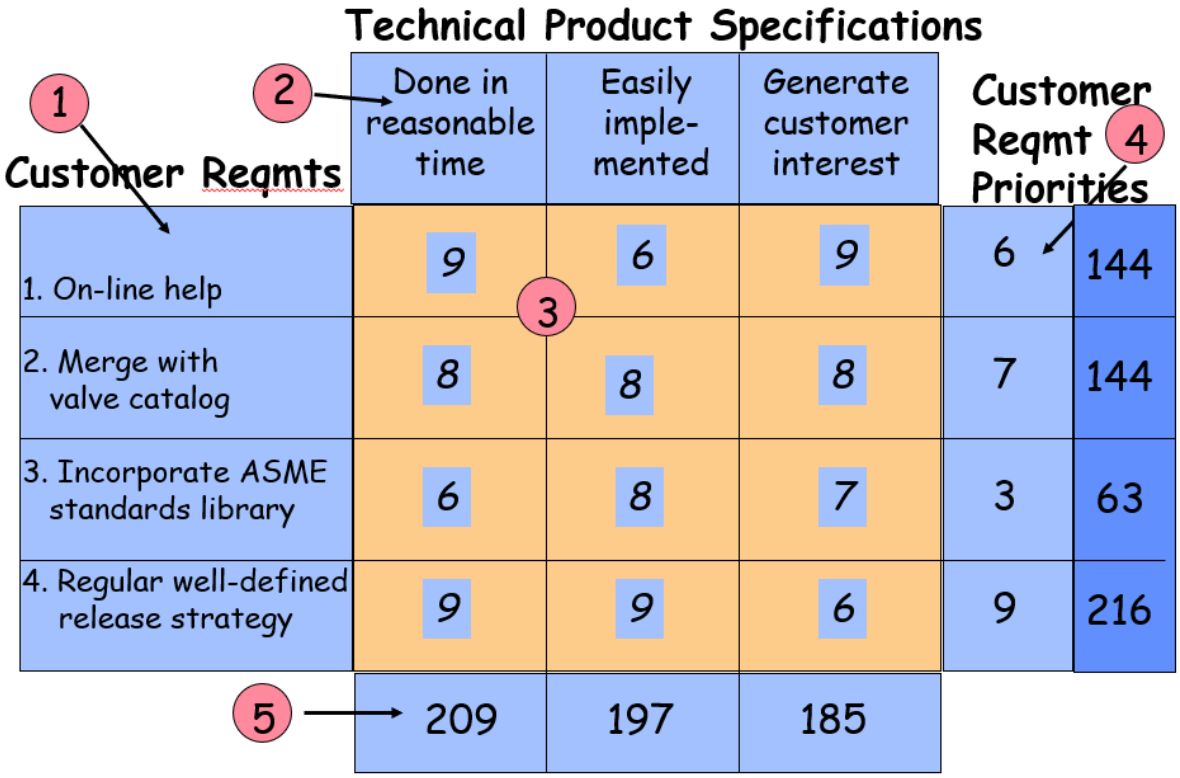
- 1) Určení požadavků na výrobek
- 2) Vymezení vah požadavků
- 3) Porovnání způsobilosti plnit požadavky
- 4) Identifikování měřitelných rysů kvality produktu
- 5) Rozbor vztahů mezi stanovenými požadavky a rysy
- 6) Porovnání schopnosti dosahovat rysů kvality
- 7) Analýza relací mezi rysy kvality
- 8) Koncept výsledných hodnot rysů kvality produktu

Použití QFD při vývoji SW (SQFD)

SQFD je technika požadavků, která je přizpůsobitelná jakékoli metodice softwarového inženýrství a která kvantifikuje požadavky a definuje kritické požadavky zákazníků. Jedná se o adaptaci matice domu jakosti (viz. výše), která je nejběžněji používanou v tradiční metodice QFD. Proces SQFD obsahuje následující kroky [32]:

- Krok 1 – Požadavky zákazníků jsou sbírány a zaznamenávány na levé straně. Mezi zákazníky mohou patřit koncoví uživatelé, manažeři, pracovníci v oblasti vývoje systému a všichni lidé, kteří by měli prospěch z používání navrhovaného softwarového produktu. Požadavky jsou obvykle krátké výkazy zaznamenané v terminologii zákazníka.
- Krok 2 – Ve spolupráci se zákazníky jsou pak požadavky převedeny na technické a měřitelné specifikace softwarového produktu a zaznamenány na horní straně. Je důležité poznamenat, že některé požadavky zákazníků mohou být převedeny na více technických specifikací produktu. Technické specifikace výrobku musí být navíc měřitelné. Použité metriky jsou obvykle numericky založeny, ale mohou být také typu Boolean (například odpověď Ano/Ne).
- Krok 3 – Zákazníci jsou požádáni o doplnění korelační matice tak, že identifikují vztahy mezi různými požadavky a technickými specifikacemi produktu. Protože je do tohoto procesu zapojeno mnoho zákazníků, je důležité získat hromadnou shodu.
- Krok 4 – Na základě údajů z průzkumů zákazníků jsou stanoveny priority pro uvedené požadavky zákazníka a jsou uvedeny na pravé straně. Další informace mohou být v tomto okamžiku shromážděny od zákazníků ohledně hodnocení softwarových produktů konkurence. Data týkající se indexu prodeje a zlepšení mohou být také získána z vývojového týmu.
- Krok 5 - Tento proces zahrnuje získání priorit technických specifikací produktu (spodní strana) sčítáním výsledků násobení priorit požadavků zákazníků s korelačními hodnotami mezi požadavky zákazníků a technickými specifikacemi produktu. Tyto váhy pro technické specifikace produktu jsou pak obvykle převedeny na procentní podíl.

Jednotlivé kroky jsou znázorněny na následujícím příkladu:



Obrázek 10: Příklad využití metody SQFD

Zdroj: [33]

Výhody použití QFD [5]:

- Zaměřuje se na to, co zákazník chce, a ne na to, co si společnost myslí, že zákazník chce, protože je řízen „Hlasem zákazník“.
- Jsou zkoumány další produkty na trhu a produkt společnosti je hodnocen proti konkurenci.
- Snížená doba vývoje: Pravděpodobnost změn návrhu je omezena, protože proces QFD se zaměřuje na zlepšení, která mají být provedena, aby splnila klíčové požadavky zákazníků. Pečlivá pozornost k požadavkům zákazníků snižuje riziko, že změny budou vyžadovány později v průběhu životního cyklu projektu.
- Snížené náklady na vývoj: K identifikaci požadovaných změn dochází na počátku životního cyklu projektu. Minimalizace změn po výrobě snižuje náklady na záruku a náklady na podporu produktu.
- Dokumentace: Při implementaci procesu QFD je vytvořena znalostní báze. Je vypracován historický záznam rozhodovacího procesu.

4.5 Porovnání a vyhodnocení vybraných metod

V této podkapitole jsem se zaměřil na subjektivní porovnání jednotlivých vybraných metod. Následující tabulka vyjadřuje srovnání všech výše popsaných technik. Samotné porovnání obsahuje několik vybraných faktorů.

Faktory:

- Časová náročnost metody (při shodném počtu požadavků)
- Nároky na tvůrce (předpoklady, které by měl mít pro úspěšné použití metody)
- Porozumění metodě
- Využití metody
- Kategorie priorit (zda se požadavky rozdělují dle priorit do kategorií)
- Důvěryhodnost výsledků
- Počet požadavků, pro které je daná metoda vhodná

Faktor/metoda	MoSCoW	KANO	Bublinkové řazení	QFD
Časová náročnost	Nízká	Střední	Střední	Střední
Nároky na tvůrce	Nízké	Střední	Střední	Střední
Porozumění metodě	Snadné	Střední	Snadné	Střední
Využití pro	Priorita	Priorita, relevantnost	Priorita	Priorita
Kategorie priorit	Ano	Ano	Ne	Ne
Důvěryhodnost výsledků	Střední	Střední	Vysoká	Střední
Počet požadavků	Středně	Středně	Málo	Středně

Obrázek 11: Porovnání metod vyhodnocení požadavků

Zdroj: vlastní zpracování

Co se týká časové náročnosti, z vybraných metod vychází nejlépe MoSCoW, což je dáno tím, že jejím principem je pouhé rozdělení požadavků do stanovených kategorií. Bublinkové řazení je sice dle svého principu také jednoduché, ale je nutné opakovaně porovnávat požadavky mezi sebou, což zvyšuje časovou náročnost. Kano model i QFD obsahují více kroků, které je nezbytné provést, a z toho důvodu jsem jim přiřadil střední hodnotu.

Nároky na tvůrce jsem stanovil nejnižší také u metody MoSCoW. U ostatních metod jsem zvolil střední nároky. U Kano modelu z důvodu nutné znalosti jednotlivých kroků a pochopení, jak přesně metoda funguje. Stejné důvody lze uvést i u metody QFD. V případě bublinkového řazení, je nutné pochopení daného algoritmu, který se v této technice využívá.

Porozumění všem metodám není nijak složité. Nicméně metoda MoSCoW a bublinkové řazení je oproti zbylým dvěma metodám výrazně jednodušší na pochopení z důvodu jejich

jednoduchého postupu, který v prvním případě spočívá v pouhém rozdělení a ve druhém v obyčejném porovnávání sousedních požadavků.

Všechny uvedené metody lze využít k určení priorit jednotlivých požadavků. Model Kano lze navíc využít pro nalezení nerelevantních požadavků, které lze odhalit ve druhém kroku této metody při pokládání pozitivně/negativně koncipovaných otázek. Pokud například zákazník odpoví na obě otázky, že to tak musí být, znamená to, že je daný požadavek nerelevantní.

Dalším faktorem byl fakt, zda technika umožňuje rozdělení požadavků do přesně stanovených kategorií. Tento fakt se potvrdil u metody MoSCoW a modelu KANO. Tyto dvě metody mají přesně definované kategorie, do kterých se následně požadavky rozdělují. Metoda MoSCoW navíc obsahuje i kategorii požadavků, které budou z aktuální implementace vyřazeny.

Důvěryhodnost výsledkům je dle mého názoru nejvyšší v případě bublinkového řazení, a to z důvodu, že se postupně porovnávají veškeré požadavky a postupně se znovu a znovu zařazují. Rovněž si myslím, že žádná z metod nemá vyloženě nedůvěryhodné výsledky, jelikož na vytváření všech metod se podílejí samotní zákazníci, kteří vědí, jak důležité jsou pro ně dané požadavky.

Jako poslední porovnávaný faktor jsem zvolil počet požadavků, pro které se dle mého názoru metoda hodí. Z tabulky lze snadno vyčíst, že nejmenší doporučený počet požadavků na zpracování je u metody bublinkového řazení. Není to z důvodu, že by při velkém počtu požadavků byla metoda nedůvěryhodná, ale jedná se o fakt, že při rostoucím rozsahu požadavků exponenciálně vzrůstá doba na provedení tohoto algoritmu. Pro větší rozsah požadavků je tedy dle mého názoru lepší využít jinou metodu.

Z vybraných metod bych metodu MoSCoW doporučil pro organizace, které mají méně zkušeností s prioritizačními metodami. Metoda neklade nijak velké nároky na tvorbu, požadavky přehledně dělí mezi stanovené skupiny, z nichž poslední kategorie dokonce stanovuje požadavky, které se zatím zavádět nebudou.

Model KANO a QFD jsou svou složitostí velmi podobné, jsou vhodné pro zkušenější pracovníky, kteří již mají nějaké zkušenosti. Obě metody shodně využívají spolupráce s koncovými zákazníky. Model KANO oproti QFD konečné požadavky zařazuje na základě mínění zákazníků do kategorií a lze ho také využít pro nalezení nerelevantních požadavků.

Bublinkové řazení je dle mého názoru vhodné pouze, pokud máme počet požadavků v řádu jednotek. Metoda je to velmi jednoduchá a dává nejlepší výsledky, protože mezi sebou porovnává všechny požadavky a tím dosáhneme přesné posloupnosti dle jejich priorit.

5 RIZIKA SPOJENÁ S POŽADAVKY

Rizika požadavků jsou rizika, která jsou přímo spojena se specifickými požadavky. Zahrnutí nebo přidání rizika může mít řadu dopadů na rizikový profil projektu. Některé požadavky mohou představovat rizika nedodržování předpisů, právních otázek nebo neočekávaných nákladů a tak dále [19].

Tato rizika jsou úzce spjata s kvalitou požadavků v tom, že požadavky na nízkou kvalitu představují pro projekt riziko. Projekty, které jsou založeny na chybných požadavcích, budou pravděpodobně čelit problémům a potenciálně selhávají [1].

V následujícím textu si blíže popíšeme některá rizika, která se mohou vyskytnout v souvislosti s požadavky.

Chybějící uživatelé

Proces řízení požadavků selhává při nedostatečné identifikaci všech zúčastněných. Příkladem toho může být zavádění nového produktu marketingovým oddělením bez zapojení oddělení prodeje [1].

Špatní uživatelé

Jedná se o zainteresované strany, které nemají potřebné znalosti, dovednosti nebo oprávnění k identifikování nebo ověřování požadavků. Například shromažďování požadavků může být přiděleno mladším zaměstnancům, kteří mají nedostatečný přístup k odborníkům v organizaci [1].

Nejednoznačné požadavky

Požadavky, které jsou definovány takovým způsobem, že jsou otevřeny nesprávnému výkladu. V některých případech mohou zainteresované strany záměrně definovat požadavky na dobu neurčitou, aby se vyhnuly rozhodnutí. V ostatních případech jsou požadavky jednoduše formulovány špatně [1].

Neúplné požadavky

Požadavky, které jsou neúplné a vedou k nestabilním, nepoužitelným nebo obecně nepřijatelným výsledkům. Například požadavky na systém, z nichž žádný nezmiňuje nároky na uživatelské rozhraní. Neúplné požadavky mohou také odkazovat na soubor požadavků, které jsou zaměřeny na funkční požadavky bez přiměřeného zohlednění požadavků nefunkčních [1].

Neuskutečnitelné požadavky

Požadavky, které přesahují možnosti organizace nebo systému, na které se vztahují. Tato rizika mohou být zmírněna rychlou proveditelností nebo hodnocením nákladů [1].

Neověřitelné požadavky

To jsou požadavky, které mají špatně definovaná kritéria pro jejich ověření [1].

Nedostatečné ověření

Zde mluvíme o požadavcích, které nebyly ověřeny příslušnými odborníky. Příkladem mohou být nefunkční požadavky na nový finanční systém, které nebyly přezkoumány analytikem z oblasti bezpečnosti [1]

5.1 Hledání chyb v požadavcích

Během zlepšování podnikových postupů pro vývoj a správu požadavků, je důležitá spolupráce všech zúčastněných stran. Prvním krokem takového zlepšení, které má vést k odstranění vzniklých problémů, je nalezení jejich příčin. Spoustu různých problémů může způsobit špatné řízení požadavků [28].

Při analýze příčin hledáme důvody problému, sledujeme jeho příznaky až k jeho příčinám, které je potřeba eliminovat. Pro grafické vyjádření analýzy příčin je vhodné využít diagram příčin a následků [28].

Ishikawův diagram

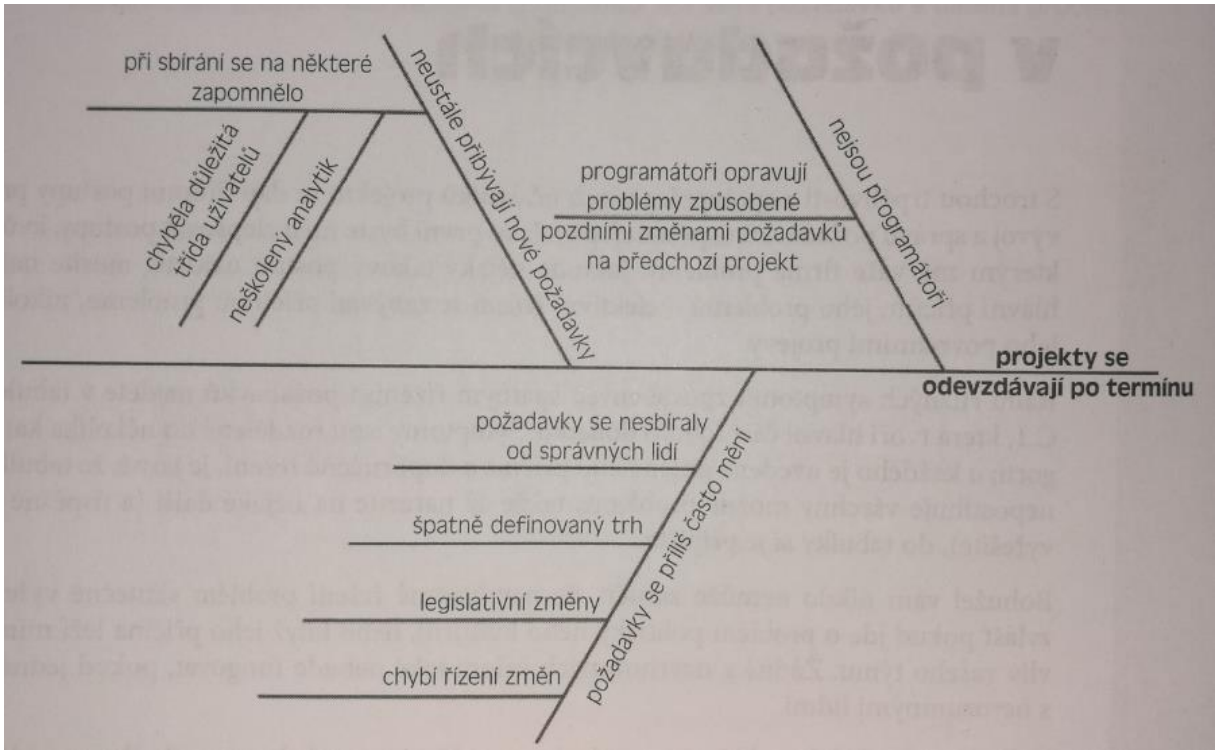
Tento diagram, který se jinak také nazývá diagram příčin a následků nebo rybí kost, vytvořil chemický inženýr Kaoru Ishikawa v roce 1943. Názvy jsou odvozeny z cíle, tvaru diagramu nebo jeho kategorií [52].

Schéma je vizuální uspořádání možných příčin problému. Schéma začíná jedním řádkem, na jehož konci je uveden problém, který má být vyřešen. Pak se přidá řada větví, které označují obecné oblasti, ve kterých lze nalézt příčiny problémů [50]. Obecnými názvy, které se používají pro tyto oblasti, jsou materiál, procesy, metody, lidé, stroje a technologie [51].

Postup použití tohoto diagramu se skládá z následujících kroků [52]:

- Definice problému, který se má analyzovat
- Nakreslení vodorovné šipky směřující doprava, na jejíž konec se umístí definovaný problém
- Nakreslení diagonálních čar, které budou kategoriemi nalezených příčin

- Nalezení možných příčin s celým týmem pomocí brainstormingu
- Rozdělení zjištěných příčin do stanovených kategorií, lze je uvádět dle jejich závažnosti nebo relevance.



Obrázek 12: Příklad diagramu příčin a následků

Zdroj: [28]

Jedná se o velmi užitečný diagram, protože informace zobrazuje přehledným a srozumitelným způsobem. Diagram má 5 hlavních přínosů [40]:

- Jasně a logicky zobrazuje vazby a vztahy mezi potenciálními příčinami a následky zobrazenými v diagramu. Vztah lze logicky pochopit na první pohled,
- Schéma ilustruje každý možný důvod v jediném diagramu, což z něj činí užitečný nástroj pro prezentaci problému a řešení všem zúčastněným stranám,
- Jedná se o skvělý způsob, jak posílit brainstorming důvody pro určité následky, protože zachycuje všechny příčiny,
- Prezentování důvodů v grafické podobě může stimulovat celý tým ke zjištění možných řešení daného problému,
- Diagram může pomoci udržet celý tým soustředěný ve chvíli, kdy diskutujeme o tom, co je důležité udělat pro vyřešení problému nebo dosažení společného cíle. Tím pádem zajišťuje, že nikdo nevěnuje pozornost řešení neexistujících problémů.

ZÁVĚR

Podstatou této práce bylo charakterizovat požadavky a jejich druhy, zdroje a možnosti vyhodnocení. Z obsahu práce vyplývá, že bez požadavků se dnes neobejdeme při tvorbě jakéhokoliv projektu, je potřeba tyto požadavky sbírat ze správných zdrojů, najít veškeré požadavky a mít možnost je vyhodnotit, abychom věděli, které požadavky jsou pro náš projekt důležité. V neposlední řadě je také důležité uvědomit si, že při práci s požadavky se mohou vyskytnout jistá rizika, se kterými je nutno počítat a umět si s nimi poradit.

První kapitola práce byla věnována definici požadavku jako takového, dále identifikace jeho druhů, které se liší dle toho, čeho se dané požadavky týkají. Dále byla část kapitoly věnována definování dat, informací a znalostí, bez kterých se při tvorbě správného a úplného produktu rovněž neobejdeme.

Druhá kapitola byla zaměřena na jednotlivé zdroje požadavků, které lze v organizaci využít. Jednotlivé zdroje se od sebe liší jak rychlostí, kterou z nich lze požadavky získat, tak i svou pouhou existencí. Například chybová hlášení a požadavky na zlepšení stávajícího informačního systému mohou využít pouze organizace, které již nějaký IS zavedený mají. V případě použití Hlasu zákazníka jej můžeme využít i při některých vyhodnocovacích metodách.

Třetí kapitola pojednává a samotných technikách sběru požadavků, což je základní činnost pří práci s požadavky, ve které se hledají uživatelské potřeby a omezení kladená na SW. Čtenář zde byl rovněž seznámen se základními právy a povinnostmi softwarových zákazníků. Bez dodržování těchto práv a povinností nemusí vůbec ke správnému sběru dojít. Jednou z popisovaných technik je i prototyping, který lze využít nejen jako nástroj pro sběr požadavků, ale také jako metodu jejich vyhodnocení. V další části této kapitoly byly představeny metody, jejichž pomocí lze vyhledat chybějící požadavky. Konkrétně se jednalo o využití diagramu datových toků, použití případů užití a matice CRUD.

Čtvrtá kapitola se zabývala konečným vyhodnocením již sesbíraných požadavků. V této práci jsem se zaměřil na metodu MoSCoW, která přesně definuje skupiny, do kterých by měly být požadavky rozděleny dle jejich důležitosti pro konečný produkt. KANO model, který umožňuje klasifikovat funkce produktu v závislosti na hodnotě, kterou poskytují svým uživatelům. Dále bublinkové řazení, což je algoritmus, jehož pomocí lze třídit požadavky dle jejich priority do uceleného seznamu. A jako poslední metodu jsem zvolil QFD, ve které se jedná o přeměnu uživatelských požadavků do návrhu produktu. Na konci této kapitoly jsem poté vytvořil porovnávací tabulky včetně popisu, dle které jsem jednotlivé vybrané metody porovnal dle vybraných faktorů a navrhl, pro koho je daná metoda vhodná.

V poslední kapitole byla představena některá rizika, která se při práci s požadavky mohou vyskytnout. Kvůli těmto rizikům může docházet k nesprávnému uchopení požadavků, a tím i k vytvoření produktu, který nebude přesně odpovídat požadavkům zákazníků. Dále byl v této kapitole představen Ishikawův diagram, který může být užitečným nástrojem v případě, že v projektu dojde k nějakému problému.

Cíle, který byl pro práci stanoven v úvodní části, bylo dosaženo. Tato bakalářská práce může sloužit jako podpůrný text při studiu problematiky zdrojů, sběru a vyhodnocení uživatelských požadavků.

POUŽITÁ LITERATURA

[1] 13 Examples of Requirements Risk. *Simplicable* [online]. [cit. 2019-04-11]. Dostupné z: https://simplicable.com/new/requirements-risk

[2] An Overview of the Kano Model. *Storm Software Solutions* [online]. [cit. 2019-04-06]. Dostupné z: http://www.stormsoftwaredevelopment.com/an-overview-of-the-kano-model/

[3] ARLOW, Jim a Ila NEUSTADT. *UML 2 a unifikovaný proces vývoje aplikací: objektově orientovaná analýza a návrh prakticky*. 2., aktualiz. a dopl. vyd. Brno: Computer Press, 2007. ISBN 978-80-251-1503-9.

[4] BHATTACHARJEE, Abhideep. The MoSCoW Method of prioritizing requirements and the final 'Oops' factor. *LinkedIN* [online]. [cit. 2019-04-02]. Dostupné z: https://www.linkedin.com/pulse/20140824071224-112433871-the-moscow-method-of-prioritizing-requirements-and-the-final-oops-factor

[5] BORYSOWICH, Craig. Benefits of Using QFD. *ToolBox* [online]. [cit. 2019-04-10]. Dostupné z: https://it.toolbox.com/blogs/craigborysowich/benefits-of-using-qfd-120406

[6] BRUCKNER, Tomáš. *Tvorba informačních systémů: principy, metodiky, architektury*. Praha: Grada, 2012. Management v informační společnosti. ISBN 978-80-247-4153-6.

[7] Customer-Focused Development with QFD. *NPD Solutions* [online]. [cit. 2019-04-06]. Dostupné z: http://www.npd-solutions.com/qfd.html

[8] How to Prioritize with the MoSCoW Technique. *Project Manager* [online]. [cit. 2019-04-02]. Dostupné z: https://www.projectmanager.com/training/prioritize-moscow-technique

[9] How to Use a CTQ Tree. *Villanova University* [online]. [cit. 2019-04-06]. Dostupné z: https://www.villanovau.com/resources/six-sigma/how-to-use-ctq-tree/

[10] How to Use Kano Model for Requirement Assessment. *SupplyhainOpz* [online]. [cit. 2019-04-06]. Dostupné z: https://www.supplychainopz.com/2013/02/kano-model.html

[11] CHEN, Chi-Ming a Victor SUSANTO. Quality Function Deployment. *Iowa State University*[online]. [cit. 2019-04-06]. Dostupné z: https://vardeman.public.iastate.edu/IE361/s00mini/chen.htm

[12] CHLEBOVSKÝ, Vít. *CRM: řízení vztahů se zákazníky*. Brno: Computer Press, 2005. Praxe manažera (Computer Press). ISBN 80-251-0798-1.

[13] KANISOVÁ, Hana a Miroslav MÜLLER. *UML srozumitelně*. 2., aktualiz. vyd. Brno: Computer Press, 2006. ISBN 80-251-1083-4.

[14] Kano Model Analysis: Delivering Products That Will Delight. *MindTools* [online]. [cit. 2019-04-02]. Dostupné z: https://www.mindtools.com/pages/article/newCT_97.htm

[15] Katalog uživatelských požadavků. *SystemOnline* [online]. [cit. 2019-02-28]. Dostupné z: https://www.systemonline.cz/sprava-it/katalog-uzivatelskych-pozadavku.htm

[16] Product Management: MoSCoW Prioritization. *Product Plan* [online]. [cit. 2019-04-02]. Dostupné z: https://www.productplan.com/glossary/moscow-prioritization/

[17] Quality function deployment. *TechTarget* [online]. [cit. 2019-04-06]. Dostupné z: https://searcherp.techtarget.com/definition/quality-function-deployment

[18] Requirements Gathering Techniques. *Axia Consulting* [online]. Chatham [cit. 2019-03-22]. Dostupné z: https://www.axia-consulting.co.uk/html/requirements_gathering.html

[19] Requirements Risk Management. *Modern Analyst* [online]. [cit. 2019-04-11]. Dostupné z: https://www.modernanalyst.com/Resources/Articles/tabid/115/ID/212/Requirements-Risk-Management.aspx

[20] Rules For Creating Data Flow Diagrams. *Eternal Sunshine of The is Mind* [online]. [cit. 2019-03-28]. Dostupné z: https://eternalsunshineoftheismind.wordpress.com/2013/02/25/rules-for-creating-data-flow-diagrams/

[21] SVOZILOVÁ, Alena. *Zlepšování podnikových procesů*. Praha: Grada, 2011. Expert (Grada). ISBN 978-80-247-3938-0.

[22] ŠIMONOVÁ, Stanislava. *Databázové systémy I*. Pardubice: Univerzita Pardubice, 2013. ISBN 978-80-7395-702-5.

[23] Techniques to Prioritize Requirements. *Modern Analyst* [online]. [cit. 2019-04-02]. Dostupné z: https://www.modernanalyst.com/Resources/Articles/tabid/115/ID/3332/Techniques-to-Prioritize-Requirements.aspx

[24] The Complete Guide to the Kano Model. *Folding Burritos* [online]. [cit. 2019-04-02]. Dostupné z: https://foldingburritos.com/kano-model/

[25] The Kano Model Explained: Analysis and Examples. *Fieldboom* [online]. [cit. 2019-04-02]. Dostupné z: https://www.fieldboom.com/kano-model

[26] VOC (Voice of Customers) – Hlas zákazníka. *Lean six sigma* [online]. [cit. 2019-03-19]. Dostupné z: http://lean6sigma.cz/voc-hlas-zakaznika/

[27] WHAT IS QUALITY FUNCTION DEPLOYMENT (QFD)?. *ASQ* [online]. [cit. 2019-04-06]. Dostupné z: https://asq.org/quality-resources/qfd-quality-function-deployment

[28] WIEGERS, Karl Eugene. Požadavky na software. Brno: Computer Press, 2008. ISBN 978-80-251-1877-1.

[29] ACHIMUGU, Philip, Ali SELAMAT, Roliana IBRAHIM a Mohd Naz´ri MAHRIN. A systematic literature review of software requirements prioritization research. *Information and Software Technology*. 2014, **56**(6), 568-585. ISSN 0950-5849.

[30] SOFTWARE REQUIREMENTS PRIORITIZATION TECHNIQUES YOU SHOULD KNOW. *Apiumhub* [online]. [cit. 2019-04-25]. Dostupné z: https://apiumhub.com/tech-blog-barcelona/software-requirements-prioritization-techniques/

[31] Bubble sort | Sorting Algorithm. *Code pumpkin* [online]. [cit. 2019-04-25]. Dostupné z: https://codepumpkin.com/bubble-sort/

[32] HAAG, Stephen, M. K. RAJA a L. L. SCHKADE. Quality function deployment usage in software development. *Communications of the ACM* [online]. **39**(1), 41-49 [cit. 2019-04-25]. DOI: 10.1145/234173.234178. ISSN 00010782. Dostupné z: https://www.researchgate.net/publication/220420894_Quality_Function_Deployment_Usage_in_Software_Development

[33] SQFD Software Quality Function Deployment - Examples. *SlideServe* [online]. [cit. 2019-04-25]. Dostupné z: https://www.slideserve.com/hampton/sqfd-software-quality-function-deployment-examples

[34] Nefunkční požadavky. PM Consulting [online]. [cit. 2019-02-27]. Dostupné z: https://www.pmconsulting.cz/slovnikovy-pojem/nefunkcni-pozadavky/

[35] Data. *Management mania* [online]. [cit. 2019-02-28]. Dostupné z: https://managementmania.com/cs/data

[36] Voice of the customer (VOC). *ISIXSIGMA* [online]. [cit. 2019-03-19]. Dostupné z: https://www.isixsigma.com/dictionary/voice-of-the-customer-voc/

[37] CRUD Matrix. *SyBase* [online]. [cit. 2019-03-28]. Dostupné z: http://infocenter-archive.sybase.com/help/index.jsp?topic=/com.sybase.stf.powerdesigner.docs_12.1.0/html/bpug/bpugp100.htm

[38] CRUD. *TMap* [online]. [cit. 2019-03-28]. Dostupné z: http://www.tmap.net/wiki/crud

[39] Use Case. *Technopedia* [online]. [cit. 2019-03-28]. Dostupné z: https://www.techopedia.com/definition/25813/use-case

[40] Benefits of Fishbone Diagrams. *Edraw soft* [online]. [cit. 2019-04-11]. Dostupné z: https://www.edrawsoft.com/fishbone-diagram-benefits.php

[41] Metoda MoSCoW a model KANO. *SystemOnline* [online]. [cit. 2019-04-02]. Dostupné z: https://www.systemonline.cz/rizeni-projektu/metoda-moscow-a-model-kano.htm

[42] Metoda QFD. *Technická univerzita Ostrava* [online]. [cit. 2019-04-10]. Dostupné z: www.elearn.vsb.cz/archivcd/FMMI/MJ/Animace/Animace%2009%20-%20QFD.pps

[43] Requirements Prioritization. *Requirements* [online]. [cit. 2019-04-02]. Dostupné z: http://www.requirements.com/Glossary/RequirementsPrioritization/tabid/121/Default.aspx

[44] About the Kano model. *Kanomodel* [online]. [cit. 2019-04-02]. Dostupné z: https://www.kanomodel.com/about-the-kano-model/

[45] MOSCOW METHOD. *Project Smart* [online]. [cit. 2019-04-02]. Dostupné z: https://www.projectsmart.co.uk/moscow-method.php

[46] Kano Model. *Open School of Management* [online]. [cit. 2019-04-06]. Dostupné z: https://www.openschoolofmanagement.com/resources/management-encyclopedia/kano-model

[47] QFD - Quality Function Deployment. *IPA* [online]. [cit. 2019-04-06]. Dostupné z: https://www.ipaczech.cz/cz/ipa-slovnik/qfd-quality-function-deployment

[48] QFD. *Quality one* [online]. [cit. 2019-04-06]. Dostupné z: https://quality-one.com/qfd/

[49] What is QFD?. *QFD Institute* [online]. [cit. 2019-04-06]. Dostupné z: http://www.qfdi.org/what_is_qfd/what_is_qfd.htm

[50] Fishbone Diagram. *AccountingTools* [online]. [cit. 2019-04-11]. Dostupné z: https://www.accountingtools.com/articles/fishbone-diagram.html

[51] Ishikawa diagram. *Vlastní cesta* [online]. [cit. 2019-04-11]. Dostupné z: https://www.vlastnicesta.cz/metody/ishikawa-diagram-1/

[52] What is an Ishikawa diagram?. *Siteware* [online]. [cit. 2019-04-11]. Dostupné z: https://www.siteware.com.br/en/strategic-management/what-is-an-ishikawa-diagram/

[53] What is bubble sort?. *Answer* [online]. [cit. 2019-04-25]. Dostupné z: https://www.answers.com/Q/What_is_bubble_sort