

# Metody odhadování softwarového projektu

## Effort Estimation for Software Projects

Bc. Tereza Špalková



## **ZADÁNÍ DIPLOMOVÉ PRÁCE**

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Bc. Tereza Špalková**  
Osobní číslo: **A13451**  
Studijní program: **N3902 Inženýrská informatika**  
Studijní obor: **Počítačové a komunikační systémy**  
Forma studia: **prezenční**

Téma práce: **Metody odhadování softwarového projektu**  
Téma anglicky: **Methods for the Estimation of Software Projects**

Zásady pro vypracování:

1. Seznamte se možnostmi a způsoby odhadování úsilí při vývoji software.
2. Seznamte s principy dotazníkového šetření.
3. Vypracujte analýzu o využívání metodik odhadování v ČR.
4. Nalezená zjištění vyhodnoťte.
5. Navrhněte základní metodiku odhadování.

Rozsah diplomové práce:

Rozsah příloh:

Forma zpracování diplomové práce: tištěná/elektronická

Seznam odborné literatury:

1. Boehm, B. W., et al.: COCOMO II Model Definition Manual. USA, Los Angeles, University of Southern California 1999. International Function Points User Group: Function Point Counting Practices Manual: Release 4.1.1999.
2. Sedláčková, J.: Cenové odhady softwarových projektů. Masarykova univerzita v Brně, Fakulta informatiky, Brno, 2005. Diplomová práce.
3. Anda, B. Comparing Use Case based Estimates with Expert Estimates. Proc. of Empirical Assessment in Software Engineering (EASE), Keele, United Kingdom, April 8-10, 2002.
4. SMITH, John. The Estimation of Effort Based on Use Cases [online]. Somrs: IBM, 2003 [cit. 2011-01-24]. Dostupné z WWW: <ftp://ftp.software.ibm.com/software/rational/web/whitepapers/2003/finalTP171.pdf>.
5. SILHAVY, Radek; SILHAVY, Petr; PROKOPOVA, Zdenka. Requirements Based Estimation Approach for System Engineering Projects. In: Innovations and Advances in Computing, Informatics, Systems Sciences, Networking and Engineering. Springer International Publishing, 2015. p. 467-472.

Vedoucí diplomové práce:

Ing. Radek Šilhavý, Ph.D.

Ústav počítačových a komunikačních systémů

Datum zadání diplomové práce:

12. ledna 2015

Termín odevzdání diplomové práce:

15. května 2015

Ve Zlíně dne 6. února 2015



doc. Mgr. Milan Adámek, Ph.D.  
děkan



Ing. Miroslav Matýšek, Ph.D.  
ředitel ústavu

### **Prohlašuji, že**

- beru na vědomí, že odevzdáním diplomové/bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová/bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou/bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou/bakalářskou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování diplomové/bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové/bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové/bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

### **Prohlašuji,**

- § že jsem na diplomové/bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- § že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně

.....  
podpis diplomanta

## **ABSTRAKT**

Diplomová práce pojedná o odhadování projektů v softwarových firmách v České republice. Práce se zabývá popisem a členěním metod, které jsou a mohou být využívány při odhadování projektů v softwarových firmách. Pojednává i o kvalitě a použití odhadu, o volbě metody a dalších náležitostí s nimi spojenými.

Cílem práce je zjistit jak a podle čeho softwarové firmy v České republice kalkulují ceny projektů. Druhotným cílem je navrhnout metodiku odhadování, kde bude popsáno, jak postupovat při vytváření odhadů, co všechno bude firma potřebovat, aby odhad mohla provést, jakou metodu odhadu zvolit, apod.

K tomu, aby nedošlo k nějakým nejasnostem, jsou v první části popsány základní pojmy potřebné pro vytváření odhadů. V další části jsou kategoricky popsány jednotlivé metody odhadování včetně jejich omezení, kvality odhadu, parametrů použití, a jejich srovnání s dalšími metodami. Předposledním krokem je průzkum, kde se zjišťovalo, jak jednotlivé softwarové firmy v ČR odhadují. Dotazníky byly rozeslány mezi softwarové firmy po celé České republice, a na základě získaných odpovědí byli vyhodnoceny tři stanovené hypotézy.

Posledním krokem, je návrh metodiky pro odhadování projektů v softwarových firmách, která byla navržena z informací, které jsem získala po nastudování problematiky na internetu, v odborné literatuře a ze získaných informací od oslovených softwarových firem.

Očekávané přínosy práce spočívají především v získání informací o metodice odhadování v softwarových firmách. O tom jaké metody odhadování se dají používat, jaké jsou zásady a vlivy, které působí na odhady a zároveň jaké jsou správné způsoby odhadování, aby vyvíjený projekt nemusel být podhodnocován.

Klíčová slova: metoda, odhad, softwarová firma, projekt, fáze, pracnosti, úsilí

## **ABSTRACT**

This thesis will discuss the estimation of projects in software companies in the Czech Republic. Thesis deals with description segmentation methods which are and may be used in estimating projects in software companies. Also discusses the quality and the use of estimates, on the choice of methods and other matters associated with them.

The goal is to determine how and by what criteria are calculating software companies in Czech Republic. A secondary objective is to propose a methodology for estimating where it will describe how to proceed in making estimates of what the company will need in order to carry out an estimate, what estimation method to choose, etc.

In order to avoid any confusion, in the first part is describes the basic concepts needed for producing the estimates. In the next part are categorically describes various estimation methods and their limitations, quality of estimation, parameters for use, and their comparison with other methods. The penultimate step is a survey where it was examined how individual software companies in the Czech Republic estimated. Questionnaires were sent between software companies across the Czech Republic, and on the basis of the replies were evaluated three stated hypotheses.

The last step is to design a methodology for estimating projects in software companies that was designed from the information I acquired after studying the issue on the internet, in specialized literature and from information obtained from surveyed software companies.

Expected benefits from the work consist mainly in obtaining information about methodology of estimating in software companies. About what estimation methods can be used, what are the principles and influences that affect the estimates and also what are the right ways of estimating that the developed project might not be undervalued.

**Keywords:** Method, estimation, Software Company, project phase, elaborateness, effort

Děkuji vedoucímu práce Ing. Radkovi Šilhavému, Ph.D., za obětavé, odborné vedení, ochotu a pomoc při zpracování mé diplomové práce.

Současně bych ráda poděkovala všem softwarovým firmám, kteří se podíleli na mém výzkumu, za jejich ochotu při vyplňování dotazníků, které mě umožnili provést výzkum v oblasti metodiky využívané v softwarových firmách.

## OBSAH

<b>ÚVOD.....</b>	<b>9</b>
<b>I TEORETICKÁ ČÁST.....</b>	<b>11</b>
<b>1 ZÁKLADNÍ POJMY PRO ODHADOVÁNÍ SOFTWAREVÝCH PROJEKTŮ .....</b>	<b>12</b>
1.1 ODHADOVÁNÍ SOFTWAREVÝCH PROJEKTŮ .....	12
1.2 KVALITA ODHADOVÁNÍ.....	15
1.3 VLIVY OVLIVŇUJÍCÍ ODHADY .....	19
<b>2 METODY ODHADOVÁNÍ SOFTWAREVÝCH PROJEKTŮ .....</b>	<b>22</b>
2.1 INDIVIDUÁLNÍ ÚSUDKY EXPERTŮ.....	22
2.2 SKUPINOVÉ ÚSUDKY EXPERTŮ .....	28
2.3 ODHADY POMOCÍ ZÁSTUPCE .....	30
2.4 ODHAD POMOCÍ ANALOGIE.....	35
2.5 ALGORITMICKÉ ODHADY .....	36
2.6 FPA – FUNCTION POINTS ANALYSIS .....	44
2.7 UCP – USE CASE POINT .....	48
<b>3 SOUHRN METOD ODHADU CENY SOFTWARE.....</b>	<b>51</b>
<b>II PRAKTICKÁ ČÁST .....</b>	<b>55</b>
<b>4 ODHADOVÁNÍ SOFTWAREVÝCH PROJEKTŮ V ČESKÉ REPUBLICE.....</b>	<b>56</b>
4.1 METODIKA .....	56
4.2 METODIKA STATISTICKÉHO ZPRACOVÁNÍ VÝSLEDKŮ .....	60
4.3 STANOVENÉ HYPOTÉZY .....	61
<b>5 VÝSLEDKY VÝZKUMNÉ ČÁSTI A JEJICH ANALÝZA .....</b>	<b>62</b>
5.1 VYHODNOCENÍ DOTAZNÍKŮ .....	62
5.2 STANOVANÉ HYPOTÉZY A JEJICH OVĚŘENÍ .....	73
5.3 SHRUTÍ VÝZKUMNÉ ČÁSTI.....	77
<b>6 NÁVRH METODIKY PRO SOFTWAREVÉ FIRMY .....</b>	<b>80</b>
6.1 NÁVRH METODIKY ODHADOVÁNÍ PRO SOFTWAREVÉ FIRMY .....	80
6.1.1 Návrh metodiky odhadování .....	82
<b>ZÁVĚR .....</b>	<b>91</b>
<b>SEZNAM POUŽITÉ LITERATURY.....</b>	<b>97</b>
<b>SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK.....</b>	<b>99</b>
<b>SEZNAM OBRÁZKŮ .....</b>	<b>101</b>
<b>SEZNAM TABULEK.....</b>	<b>102</b>
<b>SEZNAM GRAFŮ .....</b>	<b>104</b>
<b>SEZNAM PŘÍLOH.....</b>	<b>105</b>

## ÚVOD

Správné odhadnutí softwarových projektů z pohledu jejich pracnosti a doby trvání je základní podmínkou pro jeho přesné naplánování. Podle odhadu, se pak rozhodujeme, zda projekt budeme provádět a pokud ano, v jakém rozsahu. Odhady prováděné v prvotní fázi vývoje nám pak slouží pro prioritizaci nejrozumnějších projektů v rámci dané firmy. Na základě odhadu taky určujeme termín, kdy bude projekt dokončen a předán zákazníkovi. Díky tomu pak firma může určit marketingové cíle nebo smluvní závazky.

I přesto, že odhadování je jednou ze základních činností každé firmy, tak pro většinu softwarových firem je provádění odhadů nedotčenou součástí projektové řízení a není mu věnována dostatečná pozornost.

Odhady jsou z velké části prováděny bez použití jakékoliv metodiky a vytváří ji většinou projektový architekti či vývojáři. Navíc většina firem nesleduje přesnost prováděných odhadů a neuchovávají si data z projektů vytvářených v minulosti, které by mohli dané odhady zpřesňovat.

## Cíl práce

Hlavním cílem této práce je zjistit, jak a podle čeho softwarové firmy v České republice kalkulují ceny projektů. Budu se zde zabývat, tím jaké metody odhadování využívají, co upřednostňují při vytváření odhadu, a podle čeho odhady provádí, a co je k tomu vede.

Dalším cílem této práce je navrhnout metodiku pro odhadování velikosti a pracnosti softwarových projektů. Metodika bude navržen tak, aby byla univerzální a mohla být použitelná pro libovolnou softwarovou firmu v České republice.

## Struktura práce

K tomu, aby nedošlo k nějakým nejasnostem, se budu v úvodní části práce zabývat základními pojmy, které jsou důležité znát ještě před tím, než začneme vytvářet samotný odhad. Ze škály používaných metod, vyberu metody, které jsou vhodné pro odhadování velikosti a pracnosti softwarových projektů. Vybrané metody pak rozdělím do jednotlivých kategorií, které následně popíšu, včetně jejich omezení, kvality odhadu a parametrů použití. Zároveň se u každé metody zaměřím na hlavní klady, zápory a použitelnost metody s ohledem na velikost projektu a fázi vývoje, ve které se nachází.

V další části práce provedu analýzu současného stavu odhadování projektů v softwarových firmách v ČR. Analýzu provedu pomocí dotazníkového šetření, které rozešlu elektronickou

formou firmám, které se zabývají vývojem softwarových projektů po celé ČR. Stanovím si zde tři hypotézy, které následně vyhodnotím ze získaných odpovědí od respondentů. Zároveň se pokusím zmapovat, jak firmy kalkulují, jakou metodiku odhadování využívají, podle čeho kalkulují a co při odhadu upřednostňují, a zda provádí správné odhady nebo ne.

V poslední části práce provedu návrh metodiky pro odhadování velikosti a pracnosti softwarových projektů. Metodika bude navržena tak, aby byla univerzální a mohla být použitelná pro libovolné softwarové firmy v České republice. Nejdříve provedu studii, jak v dnešní době softwarové firmy kalkulují a následně provedu návrh projektové metodiky, tak i návrh metodiky celého projektu. Metodika projektu bude rozdělena na jednotlivé fáze vývoje. V každé fázi určím jaké metody pro odhad využít, co je potřeba, aby odhad aktuální fáze mohl být proveden, a jaké bude jeho použití z hlediska vývoje projektu.

## **I. TEORETICKÁ ČÁST**

# 1 ZÁKLADNÍ POJMY PRO ODHADOVÁNÍ SOFTWAREVÝCH PROJEKTŮ

Odhadování je běžná lidská činnost, se kterou se každý z nás někdy setkal a každý z nás tvoří odhady, které se od sebe liší formou a účelem. [6]

Chápání a používání slova „odhad“ se poněkud liší od reality, před kterou stojí vývojáři při vytváření odhadu vyvíjeného projektu. Zatímco v běžném každodenním životě, nemusí být odhad přesný a spolehlivý. U softwarových projektů je spolehlivost a přesnost naopak předpokladem každého odhadu, protože ze softwarových odhadů vycházejí projektové plány a závazky. [6] Nelze tedy odhady softwarových projektů brát jako průběžné odhady, naopak je nutné, aby odhad co nejpřesněji předpovídal požadované parametry úkolu či projektu.

*Definice odhadu tedy je:* prozatímní vyhodnocení nebo hrubá kalkulace, která je použitelná i když vstupní data nejsou kompletní a ne zcela jistá. [6]

## 1.1 Odhadování softwarových projektů

U softwarových projektů se především odhadují náklady na vývoj softwarového projektu a doba trvání projektu.

Není ani překvapivé, že největší podíl nákladů na softwarové projekty tvoří výdaje na plat zaměstnanců, protože odhadování z velké části závisí právě na lidské činnosti. Dále se do nákladů zahrnují i potřebné licence na software či speciální hardware, pokud jej firma potřebuje pro vývoj daného projektu. Průměrně 80 % firem zahrnuje do nákladů i pronájem kanceláří, energii a další podobné výdaje. Často se celkové náklady projektu rovnají nákladům na jednoho zaměstnance. [10] Proto je někdy efektivnější počítat vynaložené úsilí (počet člověkoměsíců na projekt = počet hodin zaměstnance/měsíc) než náklady na vývoj softwarového projektu. Přesto, ale výpočet celkového odhadu není tak jednoduchý, protože často není jasné kdo, a jak dlouho se bude na vývoji daného projektu podílet. [10] Je to totiž velký rozdíl, když expert na programování, s dvojnásobným platem než obyčejný programátor, bude projekt vyvíjet měsíc nebo několik dní. V tomto případě se volí střední cesta, např. pro výpočet nákladů se používá průměrný plat. Naopak je-li k dispozici metoda, která umožňuje určit potřebný počet programátorů a dobu vývoje pro daný projekt, lze provést mnohem přesnější odhady.

Při odhadování celkové doby potřebné pro vývoj projektu, se často vychází z odhadu úsilí. Do celkové doby vývoje se započítávají i víkendy, svátky a čas kdy zaměstnanci z nějakého důvodu nepracují. Další co se musí do odhadu započítat je fakt, že firma pracuje na dalších projektech zároveň. [10] Pokud na projektu pracuje více lidí, je potřeba do odhadu zahrnout i posloupnost jednotlivých prací. Častá chyba, která při odhadování vzniká, je že není přesně definovaný začátek a konec projektu, což potom může mít značný vliv na celkový odhad doby. Je rozdíl pokud se jako začátek projektu bere prvotní sezení se zákazníkem nebo tvorba návrhu projektu. Stejný to je s koncem projektu. [10] Je rozdíl pokud se jako konec projektu bere finální verze programu, kdy projekt je testován nebo až okamžik, kdy dojde k předání projektu zákazníkovi.

### **Historická data**

V dnešní době jsou převážně všechny metody odhadování založeny na použití historických dat. [10] Historická data jsou data, která byla uchována již z realizovaných projektů v minulosti.

Pro odhadování pomocí historických dat je dobré si uchovávat všechna vlastní data, ale především velikost projektu a skutečné úsilí. Ostatní data vám pak pouze pomohou váš odhad zpřesnit. Obecně platí, pokud použijete historická data z vlastních projektů, bude odhad přesnější. Je to dosaženo tím, že data z minulých projektů vaší firmy nejlépe odpovídají fungování firmy, procesům a lidem vzhledem k aktuálnímu projektu. [10] I když je dobré si uchovávat všechny vlastní data nelze je vždy použít. Máte-li totiž například starý projekt, který je několikanásobně menší než aktuální nelze jej použít, vždy se totiž musí charakteristika firmy a projektů podobat.

V případě že vlastní historická data nemáte, lze využít cizí historická data z projektů jiné firmy. Jednou z množností je databáze cizích projektů. Tam se stačí přihlásit a vybrat si firmu s podobným projektem, který potřebujete. [10] Pokud ale přístup do databáze nemáte nebo neexistuje podobný projekt vašemu, tak z dalších možností je využití algoritmických modelů pro odhadování. Výhodou těchto modelů je to, že v sobě už mají zahrnuta historická data. Není, ale podmínkou že pokud máte historická data k dispozici, že tyto modely využít nemůžete.

### **Odhad, cíl, závazek**

Při odhadování softwarových projektů je potřeba umět rozlišit tři důležité pojmy: odhad, cíl a závazek. Odhad je předpověď, jak dlouho potrvá vývoj projektu nebo jaká bude jeho

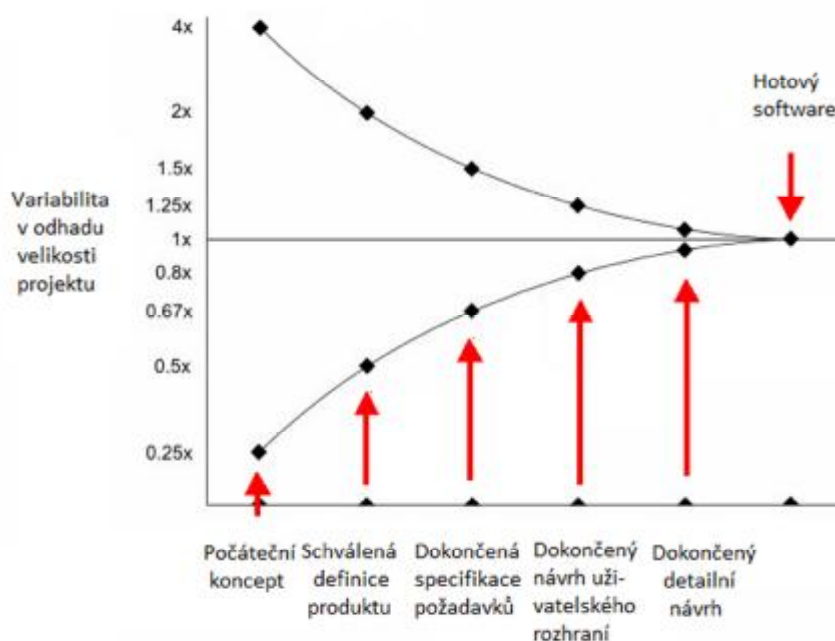
celková cena. [6] Cíl je popis procesu, jak bude vývoj projektu probíhat a závazek je slib, že projekt bude dodán v domluveném čase v požadované kvalitě a funkčnosti. [6] V praxi se, ale často stává, že jsou tyto pojmy zaměňovány nebo špatně pochopeny.

Navíc běžně se stává, že odhad je mnohem větší, než požadovaný cíl. Řešením však není ihned snižovat odhad, ale pokusit se udělat nejprve nějaký kompromis. [10] To znamená, například nepoužít v softwarovém projektu určité funkce, apod. Tím se odhad sníží a bude ve stejné úrovni, jako je obchodní cíl. V praxi to však často bývá opačně. Zákazník nebo vedení místo toho aby udělalo kompromis, tak vyvíjí velký tlak na snížení odhadu, to pak většinou vede k překročení stanoveného rozpočtu, prodloužení doby vývoje, apod. Závazek se od odhadu také může lišit, ale nemusí. [10] Většinou však hodnota závazku je pesimističtější, než je hodnota odhadu.

### **Kužel nejistoty**

Při vývoji softwaru se setkáváme s prováděním doslova tisíce rozhodnutí o vlastnostech daného projektu. Nejistota v odhadech softwaru vzniká z nejistoty, jak daná rozhodnutí dopadnou. [10] Čím více se těchto rozhodnutí provedete, tím více se sníží nejistota při odhadování. Výsledkem tohoto procesu je, že odhady projektu mají v průběhu vývoje, předpověditelný kvantum nejistot. [10]

Kužel nejistoty (*Obr. 1*) zobrazuje, jak se jednotlivé projekty v průběhu procesu zpřesňují. Horizontální osa na obrázku 1 zobrazuje průběh vývoje projektu v jednotlivých bodech, (počáteční koncept, schválená definice produktu, atd.). [6] Vertikální osa obsahuje stupeň chyby, která byla nalezena v odhadech, v různých fázích projektu.



Obr. 1: Kužel nejistoty [11]

Na kuželu nejistoty si můžete všimnout toho, že odhady prováděné v rané fázi projektu, mají velké chyby. [6] Tedy, odhady prováděné ve fázi počátečního konceptu mohou být nepřesné a to až s koeficientem 4x nahoru i dolů (u dolní hranice to znamená 0,25x tedy  $\frac{1}{4}$ ). Je tedy možný až 16násobný rozdíl v různých odhadech. To však, ale neznamená, že takto provedené odhady jsou špatné. Pouze to, že v projektu je obsaženo velké kvantum nejistoty, a není tedy možné garantovat přesnější odhady. Taky ale nehovoří nic o tom, že není možné vytvořit přesný odhad, lze vytvořit i dokonale přesný odhad, např. hned ve fázi počátečního konceptu, ale to že je pak tento odhad splněn, je pouze o štěstí.

## 1.2 Kvalita odhadování

Na otázku co je dobrý odhad, neexistuje v dnešní době žádná přesná definice. Capres Jones, ale prohlásil, že pokud se jedná o dobře řízené projekty, je možné tyto projekty odhadnout s přesností  $\pm 10\%$ . Naopak chaotické projekty této přesnosti nikdy nedosáhnout, protože mají příliš velkou variabilitu. První kdo se pokusil o definici dobrého odhadu, byli v roce 1986 tři profesori (S. D. Conte, H. E. Dunsmore a Y. D. Shen) jejichž definice se dnes stala nejpoužívanějším standardem odhadování a zní, že „dobrý přístup k odhadování by měl dávat odhady, které jsou v rozmezí 25 % okolo skutečných výsledků v 75 % času“. [6].

Přesný odhad, jak už vyplývá z kužele nejistoty, lze získat až na konci projektu, tedy až po jeho ukončení. [6] Proto pokud provádíme odhad v jakékoliv fázi projektu, dostaneme odhad s určitou nejistotou, kterou je lepší vyjádřit, než ji opomenout. [6] V ideálním případě je vyjádřena v intervalu, jehož šířka odpovídá aktuálně používané metodě, fázi projektu a dostupným informacím z projektu.

Další velmi významnou vlastností při odhadování je aktuálnost, protože pokud chceme, aby odhad byl během projektu i po jeho ukončení stále dobrým odhadem, je potřeba sním po celou dobu vývoje pracovat. [6] To znamená, že pokud dojde k jakékoliv změně rozsahu během vývoje projektu, je potřeba odhad aktualizovat. To se zdá na první pohled logické a žádoucí, přesto v praxi často dochází k tomu, že s odhady během vývoje projektu nepracujeme. Přitom si však neuvědomujeme, že pouze průběžný odhad, může být na konci projektu vyhodnocen.

Avšak i samotné vyhodnocení odhadu v sobě ukrývá několik problému. Jeden z problémů může být právě samotný průběh projektu, který se může od odhadu lišit. Například, pokud budou jednotlivé činnosti projektu špatně řízeny, naplánovány nebo provedeny, bude konečný projekt ve zcela jiných rozměrech, než byl odhadnut. Proto vyhodnocení odhadu je potřeba uskutečňovat vždy nad průběžně aktualizovaným odhadem.

### **Výhody přesných odhadů**

Další velmi významným parametrem při vytváření odhadů je přesnost, protože právě přesné odhady přináší firmám nespočet výhod. [6]

Mezi nejčastější výhody přesných odhadů patří:

- *Zlepšená čitelnost stavu* – Jeli plánovaný postup postavený na přesných odhadech, je možné sledovat postup podle stanoveného plánu. [6] Pokud není, nemá smysl srovnávat skutečný postup s plánovaným.
- *Vyšší kvalita* – Jsou-li, k dispozici přesné odhady je možné předcházet problémům, které vznikají z nedodržení plánovaného rozvrhu.
- *Lepší koordinace s nesoftwarovými aktivitami*
- *Lepší rozpočtování* – Je-li přesný odhad, je přesnější rozpočet.
- *Zvýšený kredit vývojové týmu* – Nepřesné odhady způsobují, že práce není provedena v požadovaném termínu. A právě tým, který tyto odhady tvoří a zároveň i daný úkol provádí, se snaží tento problém snížit. [6]

- *Brzká informace o riziku* – nepřesné odhady mohou skrýt určitá rizika projektu, která by šla případnými opatřeními obejít.

### Příčiny nepřesných odhadů

Příčin nepřesných odhadů je celá řada, zde je však upozorněno jen na pár z nich. Pokud bychom, ale měli shrnout všechny tyto příčiny do jednoho celku, dá se říct, že se jedná o vlivy, které snižují přesnost rozsahu projektu a posunují projekt zpět na kuželu nejistoty.

Mezi nejčastější příčiny nepřesných odhadů patří:

- 1) *Chaotické vývojové procesy* – Jsou to situace, kdy nejsou přesně definovány metodiky vývoje, čas a práce vývojářů není dostatečně využita. Podstatné je, že oproti jiným nejistotám, lze tomuto chaotickému vývoji správným vedením předejít. [6] Můžeme sem zařadit např. používání novot, zavádění zbytečností, nezkušené zaměstnanci, špatný návrh, apod.
- 2) *Opomíjené aktivity* – Opomíjené aktivity, patří mezi nejčastější avšak mezi pochopitelné chyby, kterých se při vývoji dopouštíme. Řadíme je do tří základních kategorií, a to: chybějící požadavky, chybějící aktivity při vývoji softwaru a chybějící aktivity mimo vývoj. [6]
- 3) *Měnicí se požadavky* – Měnicí se požadavky jsou opět jednou z nejčastějších problémů, které ovlivňují přesnost odhadů. Pokud dojde ke změně požadavku od zákazníka, změní se celé řešení projektu a je velmi pravděpodobné, že pak bude odlišný i odhad. [6] Budou-li se požadavky během vývoje neustále měnit, a každá změna bude správně aktualizována, může dojít k takovým změnám v odhadu, že odhad, který byl stanoven na začátku vývoje, už nebude dostatečnou základnou pro aktuálně platné požadavky. Navíc dokud nebudou požadavky stabilizovány, není možné získat přesnější odhad.
- 4) *Nepodložený optimismus* - Optimismus útočí na odhady softwaru ze všech stran. Zhodnotíme-li optimismus jako univerzální lidskou činnost, pak odhady softwaru jsou často ovlivněny čímsi, co nazýváme jako spiknutí optimismů.  
Vývojářské odhady jsou optimistické. [6] Navíc projektovým manažérům se tyto odhady líbí, protože projekt je spuštěn a běží, bez toho aniž by se někdo pozdržel nad tím, jestli byly odhady postaveny na správných základech a stanovené cíle jsou dosažitelné.

### Důsledky nepřesných odhadů

Nepřesnost odhadů softwarových projektů je problémem už několika let. Například v 70. letech 20. století poprvé poukázal Fred Brooks na to, že „kvůli nedostatku času selhávalo více softwarových projektů než kvůli ostatním příčinám“. [6] O něco později Scoot Costello upozornil, že „tlak termínu je jednoduše největší nepřítel vývoje softwaru“.

Pokud tedy provádíme přesné odhady, může být práce mezi různými vývojáři efektivně koordinována. [6] Což nám potom umožňuje, že dodávky z jedné vývojové skupiny do druhé mohou být naplánovány přesně na den, hodinu a minutu. Protože, ale přesné odhady jsou vzácné, tudíž se nám tu otevírá otázka, a to „pokud máme chybovat, je lépe chybovat na stranu nadhodnocení nebo podhodnocení?“.

### Nadhodnocení odhadu

Často se manažeři obávají, že pokud projekt nadhodnotíme, začne platit Parkinsonův zákon. [6] Parkinsonův zákon znamená, že projektový tým si najde stále spoustu práce, aby byl zužitkován veškerý dostupný čas. Bude-li mít vývojář 5 dní na dokončení daného úkolu, který lze dokončit za 4 dny, vývojář si na zbylý den práci určitě najde. To stejné je, pokud projektový tým dostane 6 měsíců na projekt, který lze dokončit za 4 měsíce, tým si najde způsob, jak zbylé dva měsíce využít. Proto se stále častěji manažeři snaží odhady stlačit a tak Parkinsonovu zákonu předejít.

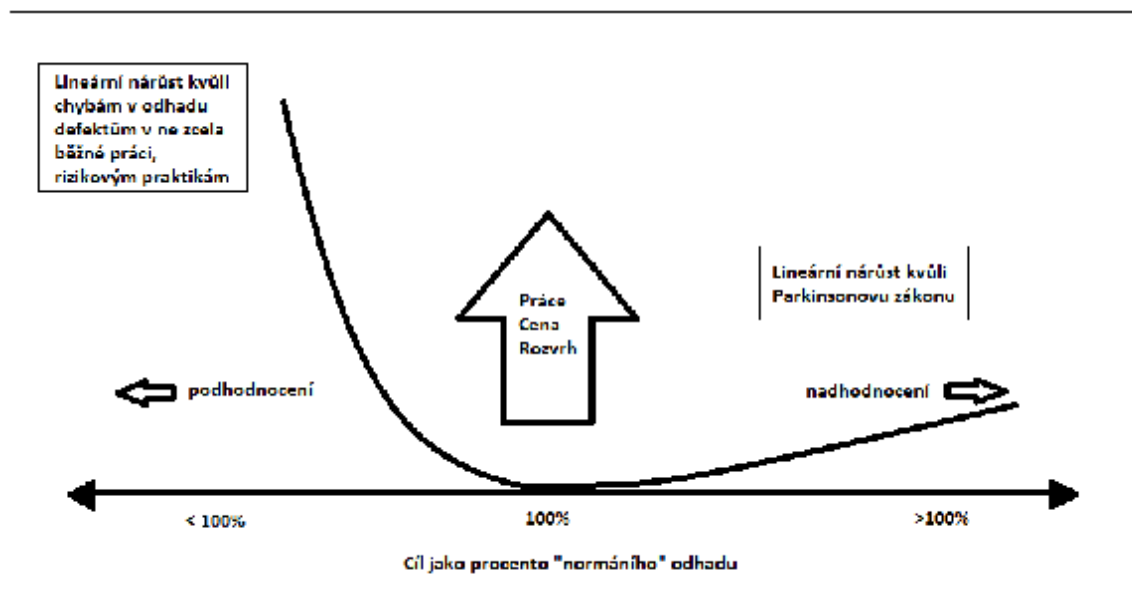
Může, ale nastat i opačný problém a to tzv. Goldrattův „studentský syndrom“. To znamená, že pokud vývojáři mají mnoho nadbytečného času, svoji práci neustále odkládají na později. Začnou pracovat až v určitém okamžiku, a i přes velkou pracovní intenzitu většinou svoji práci nedokončí včas. [6]

### Podhodnocení odhadu

Motivace pro podhodnocování je snaha vštípit vývojovému týmu pocit naléhavosti. I když podhodnocování přináší příliš problému, tak některé jsou zřejmé, jiné už tolik ne. [6] Studie uvádí, že „pokud by odhady způsobily, posunutí plánů pouze o 5 až 10 %, nezpůsobily by žádné vážnější problémy. Ale mnoho studií uvádí, že jsou odhady nepřesné o 100 a více procent. Což svědčí jen o tom, že průměrné plány projektu jsou založeny na předpokladech tak daleko od reality, že jsou tyto plány zpravidla k ničemu.

Přes všechny spekulace, ale nejlepší výsledky projektu vyházejí z přesných odhadů. [6] Protože, pokud jsou odhady velmi malé, jejich neúčinnost plánování zvýší jejich reálnou

cenu a rozvrh projektu. Naopak pokud jsou odhady velmi vysoké, začne se uplatňovat Parkinsonův zákon.



Obr. 2 Cena za nadhodnocování & podhodnocování [6]

### 1.3 Vlivy ovlivňující odhady

Každý odhad je v dnešní době ovlivňován určitými vlivy, které lze rozdělit několika způsoby. Pochopení pak těchto vlivů zlepšuje přesnost odhadů a podporuje chápání dynamiky softwarových projektů. [10] Mezi základní vlivy patří velikost projektu, lidé, druh vyvíjeného projektu a programovací jazyk.



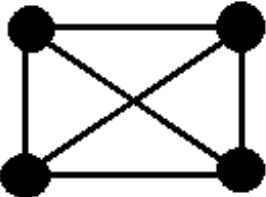
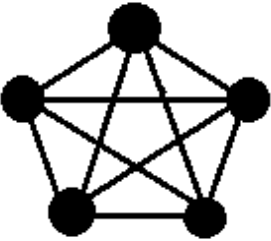
#### Velikost projektu

Velikost projektu patří mezi nejvýraznější vlivy ovlivňující odhad softwaru (celkové úsilí). To je způsobeno tím, že velikost projektu obsahuje mnohem více odchylek než jakýkoliv jiný faktor. [6]

Nevýhodou velikosti projektu je že, lidé obvykle předpokládají, že na vývoj systému, který je 10 krát větší než jiný systém je potřeba vynaložit přibližně 10 krát více práce, ale není tomu tak. [6] Protože práce na systému o 1 000 000 LOC je větší než desetinásobek práce na projektu o 100 000 LOC. Základní problém tedy spočívá v tom, že vývoj softwaru roste mnohem rychleji než velikost. U větších projektů je tedy potřeba větší komunikativnost. V případě, že počet lidí se zvětšuje, narůstá mezi nimi i kvadratický počet komunikačních kanálů.

Základní druhy komunikačních kanálů mezi 2 a více lidmi: [6]

**Tabulka 1:** Základní druhy komunikačních kanálů

	Komunikační kanál mezi 2 lidmi	1 komunikační cesta
	Komunikační kanál mezi 3 lidmi	3 komunikační cesty
	Komunikační kanál mezi 4 lidmi	6 komunikačních cest
	Komunikační kanál mezi 5 lidmi	10 komunikačních cest

### Druh vyvíjeného projektu

Další vliv ovlivňující celkovou velikost vynaloženého úsilí, je druh vyvíjeného projektu. I když velikost projektu zůstane po celou dobu stejná, produktivita práce se může jiným druhem projektu měnit. [10]

Například budete-li odhadovat produktivitu práce u kritického systému a interního intranetu a produktivitu práce vyjádříte v LOC/člověkoměsíc, pak u kritických systémů bude produktivita práce 10 krát větší než u interního intranetu. [10]

### Lidský faktor

Lidský faktor je dalším velmi významným faktorem ovlivňující, jak dobu trvání projektu, tak kvalitu softwaru. [10] Největší vliv na projekt v tomto případě nemají programátoři, ale

samotní analytici. Zkušený analytik může programátorům ušetřit čas a to až o 10-ti násobek jejich práce. Dalším důležitým vlivem působícím na projekt je míra frustrace zaměstnanců a jejich soudružnost v jednotlivých týmech. Z toho vyplývá, že pokud není jasné, kdo bude práci vykonávat, lze velmi těžko odhadnout náročnost projektu.

### **Programovací jazyk**

Použitý programovací jazyk ovlivňuje odhady celkového úsilí několika způsoby. [10] Programovací jazyky mají rozdílnou efektivitu, což znamená, že pro implementaci určité funkcionality je potřeba napsat různý počet řádků.

Jazyk jako takový může být efektivní, ale pokud s ním programátoři nemají zkušenosti, má to negativní vliv na celkové úsilí a také kvalitu softwaru. [10] Samozřejmě to neznamena, že by se neměly ve firmách začít používat nové jazyky, ale spíše nutnost u takového projektu počítat s dostatečným úsilím navíc ve formě školení a dočasné nižší efektivity programátorů. Tyto vyšší náklady je potřeba brát jako investice, které by se měly vrátit v dalších projektech.

Často se opomíjí vliv použitého jazyka na volbu softwarových nástrojů pro podporu vývoje. [10] Moderní a efektivnější programovací jazyk neznamena zrychlení vývoje, pokud k němu nejsou k dispozici vhodné podpůrné vývojové a testovací nástroje.

## 2 METODY ODHADOVÁNÍ SOFTWAREVÝCH PROJEKTŮ

### 2.1 INDIVIDUÁLNÍ ÚSUDKY EXPERTŮ

Individuální úsudky expertů jsou v praxi zdaleka nejčastějším přístupem k odhadům. Pomocí této metody můžete odhadovat jak úsilí, tak dobu trvání projektu. [10]

Použitelnost metod:

**Tabulka 2:** Použitelnost jednotlivých metod [6]

	<b>Použití strukturovaného procesu</b>	<b>Odhadování pracnosti úkolu v daném intervalu</b>	<b>Srovnávání odhadů úkolů a skutečnosti</b>
Co odhadujete	Práce, rozvrh, vlastnosti	Práce, rozvrh, vlastnosti	Velikost, práce, rozvrh, vlastnosti
Velikost projektu	Malá, Střední, Velká	Malá, Střední, Velká	Malá, Střední, Velká
Fáze vývoje	Počáteční - pozdní	Počáteční - pozdní	Střední - pozdní
Opakující nebo postupný vývoj	Oba	Oba	Oba
Dosažitelná přesnost	Vysoká	Vysoká	Není k dispozici

#### Strukturovaný úsudek experta

Výzkumy zjistili, že individuální úsudky expertů nemusí být nutně neformální nebo intuitivní. Intuitivní úsudky expertů, mají často tendence k nepřesnosti naopak strukturované úsudky expertů, poskytují mnohem přesnější odhady, a to téměř stejné jako odhady založené na modelech. [6]

Metoda strukturovaných úsudků se nejvíce hodí pro odhadování konkrétních úkolů, u kterých jsou známy veškeré požadavky, a jste schopni odhadovat jednotlivé aktivity samostatně. [10]

Často se stává, že vývojáři, testéři, manažeři, kteří vytváří odhady, rozumí pouze určitým úkolům, a úkoly kterým nerozumí, přehlíží. To vede k tendenci, že úkoly které by normálně trvali dva dny, trvají 2 měsíce. [8] Pokud tedy odhadujete na úrovni jednotlivých úkolů je dobré odhady rozložit na úkoly, které nebudou trvat déle než dva dny práce. Větší úkoly,

by již totiž obsahovali mnoho prostoru pro neočekávanou práci. Platí tedy pravidlo, čím přesněji definované a přesné aktivity máte, tím přesnější odhady dostaneme.

#### Kdo tvoří odhady?

Expertem provádějící odhadování u konkrétních úkolů by měla být osoba, která bude práci nakonec sama provádět. Odhady prováděné osobou, která sama práci neprovádí, vede k tomu, že odhady jsou méně přesné. [10] Navíc mnohem snáze odhadovatelé práci podhodnotí než vývojáři, kteří práci odhadují. Toto pravidlo avšak, platí pouze pro odhady prováděny na úrovni jednotlivých úkolů. V případě, že je projekt stále v začátcích vývoje (nemá vytvořené konkrétní úkoly), měl by být odhad proveden expertem nebo osobou kvalifikovanou v daném oboru. V lepším případě kombinace obou pravidel.

#### **Použití intervalů**

Z hlediska rychlosti a malé náročnosti, je nejjednodušší variantou provést jeden odhad pro každou aktivitu. [10] Jakmile, tedy požádáte vývojáře, aby vám vytvořil odhad pro danou sadu vlastností, dostanete něco podobného tabulce 3.

**Tabulka 3:** Ukázka jednoduchého odhadu vytvářeného vývojářem [6]

<b>Vlastnost</b>	<b>Odhad dnů potřebných pro dokončení</b>
Vlastnost 1	1.5
Vlastnost 2	1.5
Vlastnost 3	2.0
Vlastnost 4	0.5
Vlastnost 5	0.5
Vlastnost 6	0.25
Vlastnost 7	2.0
Vlastnost 8	1.0
Vlastnost 9	0.75
Vlastnost 10	1.25
<b>Celkem</b>	<b>11.25</b>

Tato varianta však není tolik přesná, proto je lepší stejného vývojáře poprosit, aby odhad opravil a doplnil každou vlastnost o nejlepší a nejhorší případ. [6] Výhodou této varianty

odhadu je, že zahrnuje nejistotu a navíc donutí vývojáře se nad daným odhadem zamyslet a tak promyslet všechny možné varianty, které by se v průběhu odhadování mohli pokazit a ovlivnit tak kvalitu odhadu. [10]

**Tabulka 4:** Ukázka odhadu s použitím nejlepšího a nejhoršího případu [6]

Vlastnost	Odhad dnů potřebných pro dokončení	
	Nejlepší případ	Nejhorší případ
Vlastnost 1	1.25	2.0
Vlastnost 2	1.5	2.5
Vlastnost 3	2.0	3.0
Vlastnost 4	0.75	2.0
Vlastnost 5	0.5	1.25
Vlastnost 6	0.25	0.5
Vlastnost 7	1.5	2.5
Vlastnost 8	1.0	1.5
Vlastnost 9	0.5	1.0
Vlastnost 10	1.25	2.0
<b>Celkem</b>	<b>10.5</b>	<b>18.25</b>

### Použití vzorce

Vytvoření odhadů nejlepších a nejhorších případů je pouze prvním krokem. Potom nastává otázka, který z těchto dvou případů použít, nebo zda je lepší použít matematický střed? Ani jedna možnost však není optimální.[6] Použijete-li nejhorší výsledek, v mnoha případech právě tento případ je horší než ten, který očekáváte. Pokud vezmete matematický střed intervalů, můžete naopak dostat zbytečně velký odhad.

Odpověď na otázku, „který z odhadů použít?“, můžete dostat nejsnadněji pomocí metody PERT. [8] Do metody PERT přidáte nejpravděpodobnější případ, který odhadnete pomocí posouzení experta. Dá se říct, že je to hodnota, kterou by odhadovatelé obvykle odhadovali při jednoduchém odhadu.

Pro výpočet očekávané pracnosti (výsledný odhad) platí tento vzorec: [6]

$$TE = (O + 4M + P)/6 \quad (1)$$

kde

O	Hodnota nejlepšího případu
P	Hodnota nejhoršího případu
M	Hodnota nejpravděpodobnějšího případu

Vypočítané odhady pracnosti pro jednotlivé případy zobrazuje tabulka 5:

**Tabulka 5:** Výpočet odhadu použitím nejlepšího, nejhoršího a nejpravděpodobnějšího případu [6]

Odhad dnů potřebný na dokončení				
Vlastnost	Nejlepší případ	Nejpravděpodobnější případ	Nejhorší případ	Očekávaný případ
Vlastnost 1	1.25	1.5	2.0	1.54
Vlastnost 2	1.5	1.75	2.5	1.83
Vlastnost 3	2.0	2.25	3.0	2.33
Vlastnost 4	0.75	1	2.0	1.13
Vlastnost 5	0.5	0.75	1.25	0.79
Vlastnost 6	0.25	0.5	0.5	0.46
Vlastnost 7	1.5	2	2.5	2.00
Vlastnost 8	1.0	1.25	1.5	1.25
Vlastnost 9	0.5	0.75	1.0	0.75
Vlastnost 10	1.25	1.5	2.0	1.54
<b>Celkem</b>	<b>10.5</b>	<b>13.25</b>	<b>18.25</b>	<b>13.62</b>

Celkový odhad dnů potřebný pro dokončení 10 vlastností byl původně 11,25 člověkodnů (Tabulka 3). [10] Tento odhad, se postupně změnil přes interval od 10,5 (nejlepší případ) do 18,25 člověkodnů (nejhorší případ) v tabulce 4 na 13,62 člověkodnů, který byl odhadnut pomocí metody PERT (Tabulka 5). Jak je z tabulky 5 vidět celkový odhad je blíže k dolnímu konci intervalu než jeho střed o hodnotě 14,4 člověkodnů, proto by pozitivní případy měli být podle expertů o něco pravděpodobnější.

Pro ještě lepší odhady, je dobré se vyvarovat přílišnému optimismu, což často bývá velkým problémem. [6] Abychom se tomu problému vyvarovali, upravíme vzorec PERT následovně:

$$TE = (O + 3M + 2P) / 6 \quad (2)$$

kde

O	Hodnota nejlepšího případu
P	Hodnota nejhoršího případu
M	Hodnota nejpravděpodobnějšího případu

Toto řešení problému je však pouze krátkodobé. [8]

Výhodou použití metody PERT je, že pro odhadnutí pracnosti můžete použít nejen výsledný odhad, který očekáváte, ale také nejhorší a nejlepší případ popřípadě jejich kombinaci pro následné tvoření projektů. [10] To vám umožní, vyhnout se velkým rozdílům mezi očekávaným a nejhorším případem, které často vznikají při detailnějším zpravování úkolů.

### Srovnání odhadů a skutečností

Oproštěním od vytváření jednočíslných odhadů s využitím nejlepších a nejhorších případů a metody PERT máte vyhráno teprve z poloviny. [8] Dalším nezbytným krokem je srovnání vašich skutečných výsledků s vašimi odhady, abyste si ujasnili, jak dobře umíte odhadovat.

Jedná se o dlouhodobou práci. Je potřeba si uchovávat veškeré své odhady, a tento seznam na konci pak doplnit o skutečné výsledky. [8] Jakmile takto vytvořený seznam máte, tak ze získané skutečně dokončené aktivity vypočítáte relativní chybu vašich odhadů (MRE - Magnitude of Relative Error).

MRE se vypočítá podle následujícího vzorce: [6]

$$MRE = |O_{real} - ORP_{(x)}| / O_{real} \quad (3)$$

kde

$O_{real}$	Hodnota skutečného výsledku
ORP	Hodnota odhadovaného výsledku

Tabulka 6 zobrazuje, jak by výpočet MRE fungoval pro výše uvedené odhady jednotlivých případů:

**Tabulka 6:** Ukázka – Tabulka pro sledování přesnosti odhadu [6]

Vlastnost	Nejlepší případ	Nejhorší případ	Očekávaný případ	Skutečné dokončení	MRE	Interval
Vlastnost 1	1.25	2	1.54	2	23 %	Ano
Vlastnost 2	1.5	2.5	1.83	2.5	27 %	Ano
Vlastnost 3	2	3	2.33	1.25	87 %	Ne
Vlastnost 4	0.75	2	1.13	1.5	25 %	Ano
Vlastnost 5	0.5	1.25	0.79	1	21 %	Ano
Vlastnost 6	0.25	0.5	0.46	0.5	8 %	Ano
Vlastnost 7	1.5	2.5	2	3	33 %	Ne
Vlastnost 8	1	1.5	1.25	1.5	17 %	Ano
Vlastnost 9	0.5	1	0.75	1	25 %	Ano
Vlastnost 10	1.25	2	1.54	2	23 %	Ano
<b>Celkem</b>	<b>10.5</b>	<b>18.25</b>	<b>13.62</b>			<b>80 % Ano</b>
<b>Průměr</b>					<b>29 %</b>	

V tabulce 6 je pro každý odhad vypočtena relativní chyba MRE, jejichž průměrná hodnota MRE je 29 %. Průměrnou hodnotu MRE, můžete využít pro kontrolu přesnosti vašich odhadů. [6] V případě, že se vaše odhady budou zlepšovat hodnota MRE se snižuje, v opačném případě narůstá. Poslední sloupec „Interval“ v tabulce 6, vám potom říká, u jaké vlastnosti jste se do intervalu „nejlepší a nejhorší případ“ trefili a kde naopak ne. Tuto informaci potom použijete pro zpětné hodnocení kvality vašich odhadů. [8]

Cílem celého tohoto procesu je získat zpětnou vazbu o odhadech, které jste prováděli, a na základě pak takto získané vazby vysledovat, v čem jste měli pravdu, v čem jste se mýlili, co jste přehlédli a jakým chybám předcházet v budoucnu. [6] Ať už ale zpětné vazby dosáhnete jakýmkoliv způsobem, klíčovým principem je nastolení zpětné vazby postavené na skutečných výsledcích, aby se Vaše odhady v průběhu času zlepšovali a Vámi používaná metoda byla úspěšná a co nejrychlejší.

## 2.2 SKUPINOVÉ ÚSUDKY EXPERTŮ

Metoda skupinových úsudků je vhodná především při odhadování v raném stádiu projektu nebo u odhadů s velkým počtem neznámých. [6]

Použitelnost metod:

**Tabulka 7:** Použitelnost metod v této kapitole [6]

	<b>Skupinové hodnocení</b>	<b>Wideband Delphi</b>
Co odhadujete	Velikost, práce, rozvrh, vlastnosti	Velikost, práce, rozvrh, vlastnosti
Velikost projektu	Střední, velká	Střední, velká
Fáze vývoje	Počáteční - střední	Počáteční - střední
Opakující nebo postupný vývoj	Oba	Postupný
Dosažitelná přesnost	Střední	Střední

### Skupinové hodnocení

Cílem metody skupinového odhadování je získat mnohem přesnější odhady než metodou odhadování jednotlivých expertů. [8] Nestrukturované úsudky expertů nemají žádné normy ani pravidla, ale doporučuje se při odhadování dodržovat 3 základní jednoduchá pravidla:

1. *Nechat všechny členy týmu individuálně odhadnout jednotlivé části projektu, a pak se setkat a diskutovat o nich.* – Je důležité diskuzi nepodceňovat a provádět ji důkladně, abyste objevili všechny zdroje rozdílů v odhadech. [6] Vždy diskutovat tak dlouho, dokud nedojdete k souladu v horním i dolním konci intervalu odhadu.
2. *Nespokojte se s prostým zprůměrováním odhadů* – Je totiž špatné brát automaticky vypočítanou průměrnou hodnotu, aniž byste nejdříve diskutovali. [8]
3. *Diskutujte, dokud nedojdete k souhlasu jednoho odhadu od všech členů týmu.* – V případě, že jeden člen nesouhlasí, dostáváte se do slepé uličky a není možné odhad schválit. [8] Musíte opět diskutovat o rozdílech a rozebrat je až dostanete souhlas od všech členů.

Tyto pravidla využívá i původní metoda Delphi. [10] Nevýhodou těchto pravidel je ale to, že více průbojní členové týmu mohou překřičet a přehlasovat členy tišší, i když svůj názor nejsou schopni nějak doložit.

Podle studie McConnella, který díky 24skupinám odhadovatelů, se kterými během studie pracoval, zjistil, že velikost relativní chyby jednotlivých individuálních odhadů je v průměru 55 %. [6] Naopak odhady zhodnocené skupinově mají průměrnou chybu pouze 30 %. Skupinové odhady jsou tedy skoro o polovinu přesnější než individuální odhady. [6]

### Wideband Delphi

Wideband Delphi je strukturovaná skupinová metoda odhadování. Vznikla z původní verze Delphi, která byla podle studií Barryho Boehma a jeho spolupracovníků pod velkým politickým tlakem. [6] Aby tento tlak eliminoval, vyvinul spolu se spolupracovníky dnes známou verzi Wideband Delphi.

Wideband Delphi se především hodí, pokud odhadujete projekt, který potřebuje mnoho různých specialit (např. algoritmické složitosti, apod.). [6] Často se, ale tato metoda používá i při odhadování práce v nové obchodní oblasti, nebo při odhadování práce na zcela novém druhu softwaru.

Dá se tedy říct, že tato metoda patří mezi nejužitečnější metody pro jednoduché odhady, přesně daných položek, které požadují vstupy z hodně oborů ve velmi široké části kužele nejistoty. [6]

Postup při odhadování pomocí metody Wideband Delphi je: [10]

1. Koordinátor Delphi seznámí všechny členy se specifikacemi a formou odhadů.
2. Všichni členové si vytvoří své vlastní odhady.
3. Koordinátor provede skupinové setkání, kde mezi sebou členové probírají problémy relevantní s projektem. Pokud už teď se skupina dohodne na odhadu, koordinátor zvolí jednoho člena, který bude zastávat opačnou pozici a snažit se ostatním tento odhad rozmluvit.
4. Koordinátor anonymně posbírání od všech členů odhady.
5. Koordinátor připraví iterační model, kde zobrazí všechny získané odhady. Členové potom na iteračním modelu vidí rozdíly v jednotlivých odhadech.
6. Na dalším setkání všichni členové rozebírají jednotlivé odhady a diskutují o nich.
7. Všichni členové hlasují, zda jsou pro navrhovaný odhad. V případě, že jeden odpoví „ne“, navrhovaný odhad se zamítá a proces se vrací do bodu 3.
8. Finální odhad je jednočíslný vycházející z procedury Delphi nebo je finální odhad interval vytvořený v průběhu diskuse Delphi a jednočíslný odhad Delphi je očekávaným odhadem.



### Fuzzy logika

Fuzzy logika se používá pro odhadnutí velikosti projektu v řádcích kódu. [6] Metoda spočívá v tom, že odhadovatelé rozdělí vlastnosti do základních kategorií podle velikosti (velmi malá, malá,...) a použitím historických dat zjistí, kolik řádků vyžaduje vlastnost velmi malá, malá, apod. Pak všechny hodnoty sečtete a získáte celkový počet řádků kódu.

**Tabulka 9:** Odhad velikosti programu pomocí Fuzzy logiky [6]

Velikost vlastnosti	Průměrný počet řádků kódu na vlastnost	Počet vlastností	Odhad počtu řádků kódu
Velmi malá	127	22	2 794
Malá	253	15	3 795
Střední	500	10	5 000
Velká	1 014	30	30 420
Velmi velká	1 998	27	53 946
<b>Součet</b>	—	<b>104</b>	<b>95 955</b>

Jako historická data můžete použít data z již dokončených projektů, a to nejlépe z projektů vaší firmy. [8] Předtím než začnete počítat počáteční průměr pro danou kategorii, je potřeba, nejdříve tyto kategorie určit. [10] Jen tak od oka se dá říct, že pro dosažení dobrých odhadů je vhodné zvolit poměry mezi velikostmi dvou sousedních kategorií v intervalu 2 až 4.

Počáteční průměry velikostí vytvoříte ohodnocením kompletní práce na jednom nebo více hotových projektech. [6] To se provádí tak, že každou vlastnost dříve vytvořeného projektu ohodnotíte, jako velmi malou, malou apod., a pak vypočítáte celkový počet řádků pro vlastnosti v dané kategorii. [6] Následně celkový počet řádků kódů podělíte, počtem vlastností a získáte průměrný počet řádků pro danou kategorii.

**Tabulka 10:** Výpočet průměrného počtu LOC pomocí Fuzzy logiky [6]

Velikost	Počet vlastností	Celkový počet LOC	Průměrný počet LOC
Velmi malá	117	14 859	127
Malá	71	17 963	253
Střední	56	28 000	500
Velká	169	171 366	1 014
Velmi velká	119	237 762	1 998

Před tím než začnete každou vlastnost zařazovat do jednotlivých kategorií, je nutné vědět, zda dispozice, o tom co je velmi malá, malá velikost apod. [8] Jsou totožné jako ty, které byly aplikovány pro výpočet průměrných vlastností.

Toho lze dosáhnout těmito způsoby:

- 1) Zajistěte, aby stejný tým, který vytvářel jednotlivé kategorie a zařazoval vlastnosti do jednotlivých kategorií, prováděl i odhad projektu.
- 2) Zajistěte odhadovatelům zaškolení pro přesnou klasifikaci vlastností.
- 3) Zajistěte zdokumentování specifických kritérií pro jednotlivé kategorie

Fuzzy logika je metodou, kterou každý odhadovatel ocení při odhadování větších celků a projektů v rané fázi. [8] Naopak nepoužitelná je pro odhady jednotlivých vlastností.

### Standartní komponenty

Tato metoda je vhodná tehdy, pokud firma vytváří programy, které jsou si architektonicky podobné. Metoda spočívá v tom, že nejprve najdete tzv. standartní komponenty. Pod standartním komponentem si můžete představit např. dynamické nebo statické webové stránky, soubory, databázové tabulky, apod. [6] Konkrétní typ, pak záleží vždy na druhu dané práce. Po nalezení komponenty se vypočítá z historických dat průměrný počet řádků kódů na komponenty nebo průměrná hodnota úsilí pro danou komponentu. [10]

Dále je potřeba odhadnout počet komponent v novém programu. [8] Tento odhad lze provést například pomocí vztahu z metody PERT (Program Evaluation and Review Technique).

Vzorec pro odhad počtu komponent: [6]

$$OPK = (K_{min} + 4N + K_{max}) / 6 \quad (4)$$

kde

OPK	Odhadovaný počet komponent
$K_{min}$	Minimální počet komponent
$K_{max}$	Maximální počet komponent
N	Nejpravděpodobnější počet komponent

Po vypočítání počtu komponent se vynásobí počet každé komponenty s průměrnou hodnotou z historických dat a sečtením všech získaných hodnot dostanete výslednou hodnotu odhadu. [8]

### Story Points

Story Points je metoda tzv. větších celků a jejich hodnocení, která se původně používala pro extrémní programování. [6] Dá se říct, že je to obnova fuzzy logiky. Od fuzzy logiky se liší tím, že v ní jsou určité zajímavé a užitečné odchylky, které dělají větší celky hodnotnými samotného pojednání.

Prvním krokem této metody je týmové sezení, kde tým společně prochází seznam celků (požadavků,...), které bude patrně vyvíjet. A každému takto vybranému celku přiřadí požadovanou velikost v bodech. [8] Toto je podobný postup jako u fuzzy logiky, ale s tím rozdílem, že celkům je přiřazena číselná hodnota z jedné ze škál, nikoliv z jednotlivých kategorií. Mezi dvě nejběžnější měřítka větších celků patří škála mocniny čísla 2 a Fibonacciho posloupnost.

**Tabulka 11:** Měřítka větším celků [6]

Škála větších celků	Konkrétní body na škále
Mocniny čísla 2	1, 2, 4, 8, 16
Fibonacciho posloupnost	1, 2, 3, 5, 8, 13

Výsledkem tohoto odhadování je seznam vlastností a jeho bodové ohodnocení. Ohodnocení je bezrozměrové, nelze tedy větší celky převést na něco srozumitelného, jako například na řádky kódu, počet dnů práce, apod. [8] Proto větší celky zde nejsou moc užitečné. Naopak důležité je, aby tým odhadnul všechny celky v jednom okamžiku, ohodnocení celků provedli podle stejného měřítka a stejným způsobem.

V dalším kroku, tým naplánuje iteraci, včetně plánování dodání určitého počtu funkčních celků. Po provedení iterace má tým určitou odhadovací schopnost, pomocí které může vyhodnotit, kolik bodů větších celků bylo dodáno, kolik práce bude potřeba a kolik uplynulo času.[6] Na základě takto získaných informací pak tým provede předběžnou kalibraci větších celků na práci a čas, a získá číslo, které bývá často nazýváno jako „rychlost“.

Bude-li v první iteraci dodáno například 27 bodů větších celků a využito 12 člověkotýdnů a celá iterace proběhne během 3 kalendářních týdnů, pak z toho plyne, že pracnost bude

2,25 bodů větších celků za člověkotýden a doba trvání projektu bude 9 bodů větších celků za kalendářní týden. [8] Takto provedená počáteční kalibrace dovoluje projektovému manažerovi vytvořit odhad zbytku projektu. Bude-li velikost projektu 180 bodů větších celků, pak odhad celého projektu se provede tak, že velikost projektu (180 bodů) se podělí 2,25 body (předběžná pracnost) a 9 body (předběžné trvání projektu). Z toho plyne, že pracnost celého projektu bude 80 člověkotýdnů a doba trvání bude 20 kalendářních týdnů.

Je zřejmé, že takto získaný odhad je ještě poměrně hrubý, avšak s každými přibývajícími daty z dalších iterací je odhad celého projektu zpřesňován. [8] Samozřejmě čím kratší iterace budou použity, tím dříve projekt získá data použitelná pro odhad zbytku projektu a umožní vytvořit dříve přesnější odhad celého projektu.

### T-shirt Sizing

T-shirt Sizing je tzv. metoda konfekčních velikostí, která se používá tam, kde investory nezajímá odhad v hodinách práce zaměstnanců, ale naopak chtějí vědět, jestli je daná vlastnost myš, králík, pes, nebo slon. [8] Vývojáři hodnotí velikost každé vlastnosti relativně k ostatním vlastnostem a to jako malou („S“), střední („M“), velkou („V“), a velmi velkou („XL“). Stejným způsobem hodnotí zákazník, pracovník marketingu, obchodník, nebo jiný netechnický pracovník každou vlastnost z pohledu přínosu, že daný požadavek bude v rámci projektu dodán. [6]

**Tabulka 12:** Použití konfekční velikosti pro hodnocení vlastností podle obchodní hodnoty a ceny vývoje [6]

Vlastnost	Obchodní hodnota	Cena vývoje
Vlastnost A	Velká	Malá
Vlastnost B	Malá	Velká
Vlastnost C	Velká	Velká
...	...	...
Vlastnost ZZ	Malá	Malá

Po skončení tohoto procesu vývojáři vidí seznam jednotlivých vlastností a již v rané fázi projektu se mohou rozhodnout, které vlastnosti vyřadit. Diskuze, o tom, co vzít a co vyhodit, je mnohem ale jednodušší, pokud se seznam vlastností utřídí podle poměru cena/výkon. [6] To se dá provést přiřazením kombinované obchodní hodnoty, která je postavená na kombinaci ceny vývoje a obchodní hodnoty. Získané hodnoty nakonec seřídíme.

**Tabulka 13:** Kombinované obchodní hodnoty založené na poměru ceny vývoje [6]

Cena vývoje				
Obchodní hodnota	Velmi velká (XL)	Velká (L)	Střední (M)	Malá (S)
Velmi velká (XL)	0	4	6	7
Velká (L)	-4	0	2	3
Střední (M)	-6	-2	0	1
Malá (S)	-7	-3	-1	0

Hodnota řazení podle přibližné obchodní hodnoty nám umožňuje vyhodit vlastnosti v horní části seznamu s typem odpovědi „rozhodně ano“ a vlastnosti v dolní části seznamu s typem odpovědi „rozhodně ne“. [8] Umožňuje tedy, aby se diskuze soustředila pouze na vlastnosti ve střední části vytvořeného seznamu.

**Tabulka 14:** Řazení odhadů podle přibližné kombinované hodnoty. [6]

Vlastnost	Obchodní hodnota	Cena vývoje	Přibližná kombinovaná hodnota
Vlastnost A	L	S	3
Vlastnost F	L	M	2
Vlastnost C	L	L	0
...			
Vlastnost H	S	M	-1
Vlastnost E	M	L	-2
...			
Vlastnost B	S	L	-3

## 2.4 ODHAD POMOCÍ ANALOGIE

Odhadování pomocí analogie spočívá v jednoduché myšlence založené na předpokladu nalezení jednoho nebo více podobných projektů odhadovanému projektu a jejich vzájemného porovnání a úpravě dat, podle zjištěných odlišností. [10]

Použitelnost metod:

**Tabulka 15:** Použitelnost metod v této kapitole [6]

	<b>Skupinové hodnocení</b>
Co odhadujete	Velikost, práce, rozvrh, vlastnosti
Velikost projektu	Malá, střední, velká
Fáze vývoje	Počáteční - pozdní
Opakující nebo postupný vývoj	Oba
Dosažitelná přesnost	Střední

**Zde je základní postup odhadu pomocí analogie: [6]**

Krok 1: *Získání detailních dat u podobného projektu vytvořeného v minulosti.*

Krok 2: *Srovnání velikostí starého a nového projektu po daných částech.*

Krok 3: *Sestavení odhadu nového projektu poměrem velikostí nového a starého projektu.*

Krok 4: *Vytvoření odhadu práce, srovnáním velikostí nového a předchozího projektu.*

Krok 5: *Ověřování souladu předpokladů ve starém a novém projektu.*

Pro kontrolu správnosti dat používaných pro odhad pomocí analogie, je potřeba se zaměřit na několik důležitých věcí, a to: na odlišné velikosti vyskytující se v jednotlivých částech odhadovaného a předchozího projektu, na odlišnost technologií používaných pro odhadování, na odlišnost ve složení jednotlivých týmu a na odlišnost softwaru. [10]

## 2.5 ALGORITMICKÉ ODHADY

Mezi algoritmické metody řadíme například COCOMO, Putmanův model nebo už ně tolik známý Halsteadův, Walston-felixův nebo Bailey-Baisilievův model. Základem algoritmických modelů jsou funkce, které mají za parametry vyčíslené faktory, které ovlivňují velikost odhadu. [10] Pokud, ale nástroj použijeme, urychlí nám to práci. Vždy ale přitom musíme dbát na to, abychom implementovali správný model, protože velmi často tyto softwary pracují s kombinací více modelů, nebo původní modely modifikují.

Algoritmické odhady nejsou příliš vhodné pro malé projekty. [10] Jedním z důvodů je zřejmá individualita mezi členy vývojového týmu, a že malé projekty jsou často jednoduché, aby se použila nebo určila přesnější metoda už hned na začátku vývoje projektu.

## Odhad COCOMO

COConstructive COst MOdel, je v dnešní době jeden z nejpoužívanějších algoritmů, který se používá pro odhad nákladů na vývoj softwarových projektů. [1] Původní model byl navržený v roce 1981 Barry Boehem jako model COCOMO 81. [10] Později v roce 1995 vznikl nástupce původního modelu, model COCOMO II, který plně nahrazuje původní model.

Nový model, byl vytvářen už tak, aby byl evoluční. To znamená, že tvůrci tuto metodu mohou aktualizovat a upravovat podle aktuálních změn v metodikách vývoje softwaru. Například COCOMO II, 2000 značí, že se jedná o metodu, která vznikla v roce 2000. [1] Proto je nezbytné ve smlouvách a při jednání s klienty uvádět vždy o jakou verzi se jedná. Pokud byste verzi neuváděli v průběhu delších a náročnějších projektu, mohli by vzniknout nesrovnalosti mezi vámi a klientem.

Základem každého modelu COCOMO jsou 2 modely:

- *Earlu Design model (EDM)* – ED model se používá v počáteční fázi vývoje projektu, kdy ještě není navrhnutá architektura a je k dispozici velmi málo informací o softwarovém projektu a jeho vývoji. [10]
- *Post-Architecture model (PAM)* - PA model už ve srovnání s ED modelem je mnohem detailnější. [10] Už se nepoužívá v úvodních fázích projektu, ale až je softwarový projekt připraven k samotnému vývoji.

I přesto, ale že každý model se používá v jiné fázi projektu, výpočet mají velmi podobný. Oba modely jsou rovněž navrženy pro odhad vývoje softwaru bez použití jakýkoliv moderních nástrojů pro rychlý vývoj softwaru (Např. RAD a GUI nástroje). [10]

Pokud vývoj bude probíhat pomocí moderních RAD technik, použije se rozšířený model APM – Application Point model.

## Základní parametry

Před každým vlastním výpočtem odhadu pracnosti a nákladnosti vývoje softwaru je potřeba stanovit základní parametry modelu. [1] Mezi základní parametry modelu řadíme: úroveň detailu výpočtu a vývojový typ modelu.

### **Základní úrovně detailu výpočtu**

Každý model COCOMO se skládá z hierarchie tří stále více detailnějších a přesnějších forem výpočtu. [2] Tyto formy stanovují, kdy a v jaké míře budou do výpočtu započítány i jiné faktory než je vlastní velikost kódu.

#### **1) Základní model – „Basic Model“**

Model Basic COCOMO je statický, jedno hodnotový model, který počítá odhady na základě velikosti programu, tedy počtu řádků zdrojového kódu. [9] Toto číslo je potom vyjádřeno, jako hrubý odhad KLOC a jeho výsledné hodnoty jsou víceméně orientační.

#### **2) Střední model – „Intermediate Model“**

Model Intermediate COCOMO už nepočítá odhady pouze na základě velikosti programu KLOC, ale do výpočtu zahrnuje i „Cost drivers“. [9] Cost drivers je množina faktorů, která zahrnuje subjektivní posouzení výrobku, hardwaru, personálu a projektových atributů. Cost drivers v těchto 4 skupinách zahrnuje celkem 15 atributů ovlivňujících výpočet odhadu nákladů softwaru. Dá se tedy říct, že odhad stanovený při této úrovni je mnohem přesnější než v základní úrovni.

#### **3) Detailnější model – „Detailed Model“**

Model Detailed COCOMO neaplikuje výpočet odhadu na celý projekt, jak tomu bylo u předchozích dvou úrovní. [9] Výpočet se aplikuje na každou etapu životního cyklu projektu zvlášť. To znamená, že model bere v úvahu i vlivy jednotlivých etap vývoje, ve kterých se právě daný okamžik nachází a ty pak přidává k výpočtu druhé úrovně.

### **Vývojové typy modelů**

Dalším základním parametrem pro odhadování pomocí Metody COCOMO je vývojový typ projektu. [2] Vývojové typy dělíme podle jejich složitosti vývoje do tří vývojových typů projektů.

#### **1) Organický typ – „Organic type“**

Organické projekty jsou relativně malé, jednoduché softwarové projekty realizovány malými týmy. Postup vývoje je známý, protože vychází ze zkušeností z podobných projektů, které byly již dříve vytvářeny. [2] Charakteristické pro tyto projekty jsou mírné požadavky na vývoj oproti jiným typům projektů a malá velikost projektu, která nepřesáhne 50 tis.



Přesné hodnoty parametrů stanovené pro daný projekt zobrazují následující tabulky. [2] Tabulka 16 pro základní úroveň projektu a tabulka 17 pro střední a detailnější úroveň projektu.

**Tabulka 16:** Hodnoty parametrů pro základní model. [2]

Hodnoty parametrů pro základní model				
Typ vývoje	a	b	c	d
Organický	2.4	1.05	2.5	0.38
Bezprostřední	3.0	1.12	2.5	0.35
Vázaný	3.6	1.20	2.5	0.32

**Tabulka 17:** Hodnoty parametrů pro střední a detailnější model. [2]

Hodnoty parametrů pro střední a detailnější model				
Typ vývoje	a	b	c	d
Organický	3.2 $F_c$	1.05	2.5	0.38
Bezprostřední	3.0 $F_c$	1.12	2.5	0.35
Vázaný	2.8 $F_c$	1.20	2.5	0.32

### Výpočet korelačního faktoru $F_c$ :

V tabulce 17 u parametru  $a$ , můžete vidět, že u konkrétních číselných hodnot se objevila další proměnná „ $F_c$ “ (korelační faktor), vyjadřující zohlednění již dříve vzpomenutých 15-ti atributů. (viz Kapitola *Základní úroveň detailu výpočtu*). [9] Dá se tedy říct, že faktor „ $F_c$ “ je součinem 15-ti hodnot popisujících softwarové, hardwarové, projektové a vývojové atributy.

Za normálních podmínek jsou zpravidla tyto atributy rovny 1. V případě, že atribut má vliv na zvýšení ceny projektu, jeho hodnota vzrůstá. [2] V opačném případě, kdy na daný parametr nemusíte při vývoji téměř přihlížet, jeho hodnota naopak klesá.

Pro výpočet korelačního faktoru platí následující vzorec: [9]

$$Fc = \prod_{k=1}^{15} Fk \quad (7)$$

Model COCOMO nabízí těchto 15 atributů, rozdělených do 4 základních skupin:

**Tabulka 18:** Patnáct základních atributů modelu COCOMO. [9]

Softwarové atributy		
Název	Popis	Rozsah „Fc“
RELY	Míra požadavků na spolehlivost	0.75-1.40
DATA	Míra rozsahu datové základny	0.94-1.16
CPLX	Složitost produktu	0.70-1.65
Hardwarové atributy		
TIME	Míra požadavků na dobu odezvy	1.00-1.66
STOR	Míra využitelnosti paměti	1.00-1.56
VIRT	Míra proměnlivosti OS počítače	0.87-1.30
TURN	Míra rychlosti oběhu úlohy počítačem	0.87-1.15
Projektové atributy		
MODP	Míra použití moderních metod vývoje softwaru	1.24-0.82
TOOL	Míra použití moder. prostředků vývoje softwaru	1.24-0.83
SCED	Přesnost požadavků na dobu realizace	1.23-1.10
Vývojové atributy		
ACAP	Míra schopnosti analytiků	1.46-0.71
AEXP	Míra zkušenosti programátorů s podobnými aplikacemi	1.42-0.70
PCAP	Míra kvality programátorů	1.29-0.82
VEXP	Míra zkušenosti s počítačem	1.21-0.90
LEXP	Míra zkušenosti s programovacím jazykem	1.14-0.95

### Stanovení odhadů metodou COCOMO II

COCOMO II je obnovou původního modelu COCOMO 81. I když model COCOMO 81 byl na svoji dobu dostačující, v 90. letech už přestávala splňovat požadavky potřebné pro vývoj softwaru. Dnes COCOMO II plně nahrazuje původní model. [9]

Model COCOMO II se skládá z 2 základních modelů a 1 rozšiřující modelu: [9]

Základní modely:

- 1) Early Design model (EDM)
- 2) Post Architecture model (PAM)

Rozšiřující model:

- 3) Application Point model (APM).

### EDM a PAM model

EDM model se používá v počáteční fázi vývoje projektu a PAM až tehdy, když je softwarový projekt připraven k samotnému vývoji. [1] I přesto, že se každý model používá v jiné fázi projektu, výpočet mají velmi podobný, liší se pouze v počtu vstupních parametrů.

Pro oba modely existují dva vzorce: [8]

- 1) Vzorec pro výpočet úsilí:

$$E = a KLOC^b \cdot EM \quad (8)$$

$$EM = \prod_{i=1}^7 EM_i \quad (9)$$

$$b = 0.91 + 0.01 \cdot \sum_{j=1}^5 W_j \quad (10)$$

kde

E - <i>Effort</i>	Úsilí (počet vynaložených člověkoměsíců)
a	Konstanta (obvykle okolo 2.94)
b	Vypočítaná konstanta dle vztahu (10)
KLOC – <i>Kilo Lines of Code</i>	Hlavní indikátor velikosti a složitosti softwaru (tisíc řádků zdrojového kódu)
EM - Effort Multipliers	Hodnota určená ze 7 atributů hodnocení driverů a jejich hodnot
$W_i$	Faktor 5-ti měřítek, určující komponenty projektů (viz Tabulka 19)

**Tabulka 19:** Měřítko zobecnění [9]

Název	Popis
PREC	Faktor, určující míru předchozích zkušeností s podobnými projekty
FLEX	Faktor, určující flexibilitu vývoje. Nelze ovlivnit.
RESL	Faktor rozhodující o rizikovosti a architektuře systému.
TEAM	Faktor soudružnosti týmu.
PMAT	Faktor, určující vyspělost procesu dle SEI CMM.

## 2) Odhad celkové doby vývoje

Kromě úsilí, lze pro oba modely vypočítat předběžnou dobu vývoje, která se vypočítá podle následujícího vztahu: [10]

$$T = \left[ 3.67 \cdot (\overline{E})^{(0.28+0.2(b-1.01))} \right] \cdot \frac{SCED\%}{100} \quad (11)$$

kde

T - <i>Time</i>	Celková doba vývoje (v měsících kalendářního času)
E - <i>Effort</i>	Úsilí (počet vynaložených člověkoměsíců), vypočítané dle vztahu (8)
b	Vypočítaná konstanta dle vztahu (10)
SCED%	je procento potřebné komprese/expanze plánu kódu

Podle takto vypočítaných hodnot lze určit potřebný počet programátorů pro daný projekt. Ten se vypočítá dle následujícího vztahu: [10]

$$AS = \frac{E}{T} \text{ [člověk]} \quad (12)$$

kde

AS	průměrný počet programátorů pro daný projekt
T - <i>Time</i>	Celková doba vývoje (v měsících kalendářního času), vypočítané dle vztahu (11)
E - <i>Effort</i>	Úsilí (počet vynaložených člověkoměsíců), vypočítané dle vztahu (8)

**Model APM**

Model APM se používá, pokud vývoj bude probíhat pomocí moderních RAD technik. Základem tohoto modelu jsou objektové body, nikoliv LOC metrika používána u předchozích dvou modelů. [2]

Počet nových objektových bodů se vypočítá, dle následujícího vztahu: [8]

$$NOP = OP \cdot (100 - r) / 100 \quad (13)$$

kde

NOP	počet nových objektových bodů
OP	počet objektových bodů
r	procento opakovaně použitých objektů z projektů v minulosti

Jakmile máte vypočítaný počet nových objektových bodů, můžete určit úsilí dle následujícího vztahu: [8]

$$E = NOP / PROD \quad (14)$$

kde

NOP	počet nových objektových bodů
PROD	je míra produktivity založená na zkušenosti vývojářů

## 2.6 FPA – Function Points Analysis

FPA je metoda pracující s funkčními body, která byla vyvinuta v roce 1983 Allanem Albrechtem. Je to metoda, která nám umožňuje provádět odhady na základě analytické dokumentace, ještě před samotným návrhem projektu. [2] Tím obchází problematiku, spojenou se stanovením velikosti kódu, kterou trpěla především metoda COCOMO. U COCOMO pokud není k dispozici odhad předpokládané velikosti kódu, je metoda COCOMO nepoužitelná.

Každý výpočet pomocí metody funkčních celků probíhá v několika krocích, které jsou při provádění odhadu potřeba provést: [10]

- 1) *Vypočítat tzv. neupravené funkční body (NFP).* Pod neupravenými funkčními body si můžete představit základní charakteristiky vyvíjeného softwaru. [9] Výpočet pak těchto bodů spočívá v tom, že identifikujete základní funkční typy a jejich příslušné složitosti. [2] Součtem pak všech získaných váhových hodnot, získáte počet neupravených funkčních bodů (NFP).
- 2) *Vypočítat tzv. upravené funkční body, vztahující se k danému projektu (FP).* Neupravené funkční body vynásobíte tzv. faktorem přizpůsobení. [9] Tento faktor potom na základě 14 atributů definující systém o něco přesněji, změní výpočet NFP na upravené funkční body.
- 3) *Určit pracnost a cenu 1 funkčního bodu, a z toho vypočítat celkové náklady vyvíjeného softwarového projektu.* [2] Ze stanoveného počtu upravených funkčních celků stanovíte pracnost a cenu jednoho funkčního bodu a z toho pak vypočítáte celkové náklady potřebné na vyvíjený softwarový projekt.

## **Výpočet neupravených funkčních bodů**

Prvním krokem výpočtu pomocí metody funkčních bodů je vypočítat tzv. neupravené funkční body. Pro jejich výpočet, je potřeba rozdělit a definovat požadovanou funkcionality systému do 5 základních tříd funkcí, které jsou rozděleny do 2 kategorií. [9] První kategorií je kategorie datových funkcí, kam patří vnitřní logické soubory (FILE) a soubory vnějších rozhraní (FILEE). [2] Druhou kategorií je kategorie transakčních funkcí, kam patří externí vstupy (IN), externí výstupy (OUT) a externí dotazy (ENG).

### **Kategorie transakčních funkcí**

Transakční funkce jsou definované třemi třídami funkcí: [9]

#### **1) Externí vstupy – IN**

Funkce IN, definuje počet externích vstupů. Pod externími vstupy, si můžete představit funkci, proces, transakci, která umožňuje uživateli aplikace provádět změny ve vnitřní struktuře dat (aplikaci) a to buď formou přidávání, mazání nebo jejich editací. [9]

#### **2) Externí výstupy – OUT**

Funkce OUT, definuje počet externích výstupů. Pod externími výstupy, si můžete představit všechna unikátní uživatelská nebo řídicí data, která opouští hranice aplikace. [9] Často se jedná o různá hlášení, zprávy, apod., které dále slouží, jako podklad pro další rozhodování a zpracování u jiných aplikací, kam byli zaslány.

#### **3) Externí dotazy – ENG**

Funkce ENG, definuje počet externích dotazů. [2] Dá se říct, že ENG je obnova funkcí IN a OUT, s tím rozdílem, že v tomto případě se nejedná o manipulaci s daty, ale o dotazování se po konkrétních informacích v souborech, které byli zprostředkovány naší aplikací.

### **Kategorie datových funkcí**

Datové funkce jsou definované 2 třídami funkcí:

#### **1) Vnitřní logické soubory – FILE**

Soubor FILE, definuje počet vnitřních logických souborů. Jako vnitřní logický soubor, si můžete představit soubor dat. Data jsou generována, používána a udržována systémem, který právě vyvíjíte. [2] V podstatě sem řadíme pracovní datové soubory, na kterých bude vyvíjený systém postavený a ze kterých bude čerpat. Vždy se jedná pouze o logické soubor

ry, nikoliv o fyzické. Pro představu to mohou být např. datové soubory bez definovaných vzájemných relací. [9]

## 2) Soubory vnějších rozhraní – FILEE

Soubor FILEE definuje počet souborů vnějších rozhraní. Jako vnější logický soubor si můžete představit soubor, který tvoří stejné logické části jako soubor FILE. Rozdíl, oproti prvnímu případu je, že soubor je sdílen více programy a data nejsou systémem udržována ani zpracovávána. [9] Data systém pouze využívá odkazováním na ně. Pro představu se může například jednat o posloupnost zpráv, které jsou předávány z jednoho programu do druhého.

### Postup při výpočtu neupravených funkčních bodů: [2]

- 1) Naleznete všechny používané aplikace IN, OUT, ENQ, FILE, FILEE
- 2) Nalezené funkce rozdělíte do jednotlivých skupin podle třídy a složitosti
- 3) Počet prvků každé skupiny vynásobíte příslušnou váhou
- 4) Na závěr všechny získané hodnoty sečtete a získáte počet neupravených bodů.

**Tabulka 20:** Ukázka - Výpočet neupravených funkčních bodů. [2]

Funkční typy	Složitost			
	Jednoduché	Průměrné	Složitě	Celkem
IN	___ x 3 +	___ x 4 +	___ x 6 =	___
OUT	___ x 4 +	___ x 5 +	___ x 7 =	___
ENQ	___ x 3 +	___ x 4 +	___ x 6 =	___
FILE	___ x 7 +	___ x 10 +	___ x 15 =	___
FILEE	___ x 5 +	___ x 7 +	___ x 10 =	___
<b>Počet neupravených funkčních bodů</b>				___

### Výpočet upravených funkčních bodů

Druhým krokem výpočtu je, že z neupravených funkčních bodů se vypočítají upravené funkční body pro daný systém. [2] Při výpočtu jsou vždy zohledněny i vlivy 14-ti faktorů obecných charakteristik systému.

Seznam 14-ti uvažovaných faktorů technické složitosti je:

**Tabulka 21:** Seznam 14-ti uvažovaných faktorů. [9]

1.	Distribuované zpracování dat	8.	Přímá oprava
2.	Datová komunikace	9.	Složitost zpracování
3.	Výkonost	10.	Znovupoužitelnost
4.	Plné využití konfigurace	11.	Jednoduchá instalace
5.	Rychlost zpracování transakcí	12.	Jednoduché použití
6.	Přímé vstupy dat	13.	Možnosti nasazení
7.	Efektivita koncového uživatele	14.	Možnost změny

Vztah pro odhad upravených funkčních bodů (FP) je: [9]

$$FP = NFP \cdot TCA \quad (15)$$

$$TCA = 0.65 + 0.01 \cdot DI \quad (16)$$

$$DI = \sum_{i=1}^{14} DI_i \quad (17)$$

kde

TCA	faktor přizpůsobení
DI	faktor technické složitosti

### Výpočet odhadu celkových nákladů pomocí funkčních bodů

Ze stanoveného počtu upravených funkčních celků stanovíte pracnost a cenu jednoho funkčního bodu a z toho pak vypočítáte celkové náklady potřebné na vyvíjený softwarový projekt, vynásobením této hodnoty celkovým počtem funkčních bodů. [2]

Kromě stanovení celkových nákladů na vyvíjený software je možné stanovit i odhad velikosti kódu, který se stanovuje v závislosti na programovacím jazyce, ve kterém daný software vyvíjíte. [2] Velikost kódu pak odpovídá jednomu funkčnímu bodu, který odpovídá počtu řádků kódu v LOC.

Vztah pro odhad velikosti kódu LOC je:

$$LOC = FP \cdot LOC/FP \quad (18)$$

Vztah pro odhad vynaloženého úsilí E:

$$E \cong c \cdot FP + b \quad (19)$$

nebo

$$E \cong c \cdot FP^b \quad (20)$$

kde

c, b - lze odhadnout regresní analýzou pro logaritmy metrik E a FP. [2]

Vztah pro odhad doby vývoje projektu je:

$$T = FP^{0.4} \text{ [měsíce]} \quad (21)$$

Vztah pro odhad počtu pracovníků potřebných pro vývoj softwaru je:

$$AS = FP/150 \quad (22)$$

## 2.7 UCP – Use Case Point

Metoda Use Case Point, byla původně vyvinuta Gustavem Karnerem. [5] Je založena na metodě funkčních bodů (Function point) a cílem její vzniku bylo, poskytnout jednoduchou metodu odhadu přizpůsobenou k objektově orientovaným projektům.

V dnešní době se Metoda Use Case Point, používá pro odhad úsilí na základě modelu případu užití. [4] Jakmile máte tedy vytvořený model případu užití, můžete tuto metodu použít.

Základní vzorec pro výpočet odhadu pomocí metody UCP je: [7]

$$\text{CelkovýOdhad} = UUCP \cdot TCF \cdot ECF \cdot PF \quad (23)$$

nebo

$$\text{CelkovýOdhad} = UCP \cdot PF \quad (24)$$

Kde

$$UCP = UUCP \cdot TCF \cdot ECF \quad (25)$$

UUCP	Nevyrovnaný počet bodů
TCF	Technický faktor
ECF	Faktor prostředí
PF	Faktor produktivity

### Výpočet UUCP

Po vytvoření modelu případu užití rozčleníme jednotlivé body případu užití do 3 základních kategorií podle jejich složitosti, [7] kde každé kategorii je přiřazena odpovídající váha.

**Tabulka 22:** Základní kategorie složitosti – případy užití. [7]

Kategorie	Popis	Počet transakcí	Váha
Jednoduchý	Jednoduché uživatelské rozhraní s vazbou na jednu databázovou entitu.	< 4	5
Střední	Víceuživatelské rozhraní s vazbou na několik databázových entit.	4-7	10
Složitý	Složitě uživatelské rozhraní s vazbou na více jak 3 databázové entity.	>7	15

**Tabulka 23:** Základní kategorie složitosti – aktéři. [7]

Kategorie	Popis	Váha
Jednoduchý	Jiný systém komunikující přes aplikační programové rozhraní (API).	1
Střední	Uživatel se znakovým terminálem nebo jiný systém komunikující přes TCP/IP.	2
Složitý	Osoba komunikující přes uživatelské rozhraní nebo webové stránky.	3

Jakmile jsou rozčleněny jednotlivé body, počet případů užití pro danou kategorii vynásobíme odpovídající váhou dané kategorii (získáme hodnotu UUCW) a celý tento proces zopakujeme pro všechny aktéry. [7] To znamená, že počet případů užití tentokrát vynásobíme počtem aktérů (získáme hodnotu UAW). Sečteme-li tyto dvě hodnoty, získáme výpočet UUCP.

Vzorec pro výpočet UUCP: [7]

$$UUCP = UUCW + UAW \quad (26)$$

### Výpočet TCF a ECF

Dalším krokem je výpočet hodnoty faktoru technické složitosti (TCF) a faktoru složitosti prostředí (ECF). [7] Abychom, ale tyto faktory mohli vypočítat, musíme nejdříve určit hodnotu technického faktoru (TF) a faktoru prostředí (EF).

Výpočet hodnoty TF

Každý technický faktor je složen z dílčích vlastností (bezpečnost, jednoduchost užití, přenositelnost,...), Každá z jeho vlastností má svoji definovanou váhu a odhadovatelem přiřazenou individuální váhu (faktor vlivu), kterou určuje podle jejího vlivu na odhadovaný systém. [3] Stupnice faktoru vlivu na odhadovaný systém je 0 (žádný vliv) až 5 (silný vliv). Pro výpočet TF se tyto dílčí váhy mezi sebou vynásobí a vypočítané hodnoty sečtou.

Vzorec pro výpočet celkového technického faktoru je:

$$TF = \sum_{i=1}^n VáhaVlastnosti_i \cdot FaktorVlastnosti_i \quad (27)$$

Výpočet hodnoty EF

Výpočet faktoru prostředí se téměř neliší od výpočtu technické faktoru. Opět je složen z dílčích vlastností. [3] Každá vlastnost má svoji definovanou a odhadovatelem přiřazenou váhu. Váhy jsou opět sečteny a vynásobeny.

Jediný rozdíl je v tom, že některé faktory prostředí mohou nabývat záporných hodnot a jiné naopak kladných. [4] To znamená, že některé faktory zvyšují a jiné naopak snižují celkový odhad.

Vzorec pro výpočet celkového technického faktoru je:

$$EF = \sum_{i=1}^n VáhaVlastnosti_i \cdot FaktorVlastnosti_i \quad (28)$$

Jakmile máte vypočítané hodnoty TF a EF můžeme je dosadit do následujících vzorců (Vzorec X a vzorec X) a vypočítat hodnoty TCF a ECF. [5]

Vzorec pro výpočet TCF:

$$TCF = 0.6 + 0.01 \cdot TF \quad (29)$$

kde

TCF	hodnota faktoru technické složitosti
TF	hodnota technického faktoru

Vzorec pro výpočet ECF: [5]

$$ECF = 1.4 - 0.03 \cdot EF \quad (30)$$

kde

ECF	hodnota faktoru složitosti prostředí.
EF	hodnota faktoru prostředí

### 3 SOURHN METOD ODHADU CENY SOFTWARE

Při navrhování nového projektu je pro firmy důležité stanovit takovou cenu projektu, která bude odpovídat odhadu nákladů. Velmi podstatnou složku nákladů tvoří výdaje na zaměstnance a proto je důležité při odhadování velikosti a ceny projektu zjistit, o jak velký objem práce jde. [12] To však v prvních fázích projektu je velmi obtížné zjistit, protože nemáme kompletní a detailnější informace o projektu. Řešením jsou však metody odhadování, které firmám poskytnou jasnou představu o rozsahu nákladů. [12]

Existují tři základní kategorie metod odhadu velikosti a ceny projektu: [11]

- Expertní odhady
- Odhady na základě analogie
- Odhady založené na modelu

Každá z uvedených metod, může provádět odhad postupem shora/dolů, kde získáme celek sečtením odhadů pro jednotlivé části projektu a postupem zdola/nahoru, kde získáme pracnost jednotlivých částí projektu podílem z celku. Při odhadu je dobré vždy použít několik různých technik odhadování a ty mezi sebou kombinovat. [11] V případě, že se odhady od sebe výrazně liší, je potřeba zjistit více informací a celý odhad provést znovu.

Pomocí studií, bylo zjištěno, že nejpoužívanější způsobem odhadu je expertní odhad, který se využívá v 70-80 % případech odhadování. Jedním z důvodů je komplikovanost formálních metod odhadu založených na modelu a dalším důvodem je, že neexistuje ani jeden pádný důkaz, že by formální metody směřovali ke kvalitnějším výsledkům odhadu než expertní metody. [11] I když podle studie B. Anda bylo zjištěno, že odhad provedený metodou Use Case Point, poskytl bližší výsledky skutečnosti, než odhad provedený 37 experty.

#### Expertní odhady

Technika expertních odhadů vychází z odhadu vytvořeného odborníkem na základě jeho předešlých zkušeností. [11] Tento způsob odhadování je prospěšný především tam, kde nemáme k dispozici měřitelná data potřebná pro daný odhad.

Při vytváření expertního odhadu můžeme například vycházet z postupu, který navrhli konzultanti společnosti Software metrics. [11] Postup je založen na myšlence, že fáze analýzy představuje 20 % z celkové pracnosti. Z toho pak může expert na základě znalosti požadavků a problémové oblasti odhadnout pracnost analýzy. Odhad celkové pracnosti pak

získáme vynásobením pracnosti analýzy pěti ( $5 \times 20 \% = 100 \%$ ). Pěti proto, že životní cyklus projektu má 5 fází. [11] Pro představu je rozložení pracnosti mezi etapy znázorněno na obrázku 3.



Obr. 3: Rozložení pracnosti mezi etapy [11]

Dá se tedy říci, že toto pojetí odhadování má velmi blízko k reálnému navrhování projektu. Pro správné řízení projektu, je ale potřeba mít dostatečné informace o tom jaké úkoly, v jakém pořadí a po jak dlouho dobu budou prováděny. Zhodnotíme-li tuto metodu, můžeme říci že, je to jedna z nejlepších metod pro navrhování projektového plánu a metoda pomocí, které je možné včas odhalit zpoždění projektu. [11] Avšak i tato metoda není bez chyb. Nevýhodou je, že při odhadování pracujeme pouze s všeobecnými úkoly. Pokud tedy odhadce nemá dostatečné znalosti a zkušenosti není moc realistické získat relevantní odhad. [12] Naopak poměrně přesné odhady lze získat u týmu odhadovatelů, kteří pracují na stejných nebo podobných projektech. Obecně platí, čím větší počet znalců provádí odhad, tím jsou kvalitnější a objektivnější výsledky odhadu.

### Analogie

Tato metoda odhadování spočívá v jednoduché myšlence založené na předpokladu nalezení jednoho nebo více podobných projektů odhadovanému projektu a jejich vzájemného porovnání a úpravě dat, podle zjištěných odlišností. [11] Nevýhodou je, že pokud nemáme k dispozici data o podobných projektech, není možné tuto metodu použít. Ani to, že jsme někdy v minulosti vytvářeli podobný analogický projekt, ještě neznamená, že máme k dispozici data o tomto projektu.

### Odhady založené na modelu

Na základě toho, že v dnešní době existuje spousta informací o skutečných projektech a jejich charakteristikách, byly vytvořeny modely závislosti velikosti projektu na pracnosti. Dá se říci, že se jedná o algoritmy, které na bázi vstupní hodnoty (velikost projektu) vypočtou výstupní hodnoty projektu (pracnost a délku trvání projektu). [1] Výpočet je však ovlivňován mnoha faktory, které jsou zadávány subjektivně ve vymezeném měřítku. (např. znalosti vývojářů velmi dobré, dobré, apod.).

Mezi nejznámější algoritmy, které provádí odhady založené na modelu, řadíme COCOMO a FPA (Function Point Analysis). [1] Avšak ani jeden z těchto algoritmů není vhodný pro odhady, které jsou realizovány ve fázi požadavků a pro systémy, které jsou navrženy objektově orientovaným přístupem.

Metoda COCOMO provádí odhady na bázi LOC (počet zdrojových řádků) vyvíjeného systému. FPA má oproti COCOMO tu výhodu, že umožňuje provádět odhady na základě analytické dokumentace, ještě před samotným návrhem projektu. [11] Tedy, jako rozsah složitosti využívá „funkční body“, nikoliv LOC. Tím obchází problematiku, spojenou se stanovením velikosti kódu, kterou trpěla především metoda COCOMO. [2] U metody COCOMO pokud není k dispozici odhad předpokládané velikosti kódu, je metoda COCOMO nepoužitelná. [11] Získat ale počet funkčních bodů není tak banální jak se zdá, protože je to možné až ve fázi detailního návrhu projektu. To je však už ale pozdě, protože cena a smlouva je už obvykle dohodnuta a podepsána.

V praxi existuje samozřejmě mnoho jiných metod odhadu ceny a pracnosti, avšak už nejsou tolik objektivní.

Shrneme-li to, tak přes veškerá možná zkreslení, zůstává stále fakt, že odhad pracnosti obzvlášť v prvotních fázích vývoje je velmi náročný. [11] Proto je dobré provádět průběžné odhady pracnosti, které slouží ke kontrole, že vývoj probíhá tak jak by měl. Pokud totiž projekt není naplánován realisticky, a provede se špatný odhad, projeví se to ve zvýšené pracnosti projektu. Bude-li projekt nadhodnocen, budou dle Parkinsonova zákona, využity veškeré zdroje. V opačném případě, kdy bude projekt podhodnocen, náklady na projekt se zvýší díky nerealistickému plánu a velkému počtu chyb.

**Tabulka 24:** Srovnání vybraných metod odhadu velikosti a ceny softwaru  
[6],[11],[12],[13]

<b>Souhrn vybraných metod odhadu velikosti a ceny softwaru</b>				
<b>Zkratka</b>	<b>Název</b>	<b>Metrika</b>	<b>Výhody</b>	<b>Nevýhody</b>
COCOMO	Constructive Cost Model 1.1	řádky kódu	Snadná aktualizace odhadu. [6]	Není zohledněna znovupoužitelnost kódu a jsou časově náročné a pracné. [12]
COCOMO II	Constructive Cost Model 2.0	objektové body	Je zohledněna znovupoužitelnost kódu. [12]	Jsou časově náročné a pracné. [12]
Delphi	-----	člověkodny	Menší náročnost na spotřebu zdrojů zohlednění specifik posuzovaného systému. [13]	Vysoké nároky na organizaci, zpracování, čas potřebný k získání výsledného názoru. [13]
FPA	Function Point Analysis	funkční body	Standardizováno a umožňuje přesné odhady. [12]	Přesné odhady získáme až v pozdějších fázích při detailnějším návrhu a není zohledněna kvalita pracovníků. [12]
PERT	Program Evaluation and Review Technique	člověkodny	Počítá s nejistotou a je vhodná pro velké projekty. [12]	Časově náročná a pracná metoda a nejsou zde zohledněny disponibilní zdroje. [12]
UCP	Use Case Points	use case body	Provádění odhadů hned v počátečních fázích projektů [12]	Není standardizována [11]

## **II. PRAKTICKÁ ČÁST**

## 4 ODHADOVÁNÍ SOFTWAREVÝCH PROJEKTŮ V ČESKÉ REPUBLICE

V praktické části se zaměřuji na metody určování ceny softwarových projektů. Cílem práce je vypracovat průzkum jak se oceňují (kalkulují) softwarové projekty v České republice. Práce přinese přehled, jak pracují české softwarové firmy při kalkulování ceny díla. Byly stanoveny tři hypotézy, které jsem se snažila ponechat v platnosti nebo vyvrátit. Zároveň jsem se snažila přispět firmám doporučením, jak by měli při kalkulaci postupovat a z čeho vycházet.

Pro analýzu a jednotlivá šetření jsem si vybrala firmy po celé České Republice zabývající se vývojem a realizací softwarových projektů. Celkem se na mém projektu podílelo 92 firem z 250 rozeslaných dotazníků.

### 4.1 Metodika

Pro získávání a shromažďování informací k danému tématu jsem zvolila dotazník, který jsem sestavila po prostudování dané problematiky v odborné literatuře a na internetu. K vytvoření dotazníku mě vedla snaha se dozvědět více o současném stavu v softwarových firmách po celé ČR, co se týče používání a aplikace metod pro odhadování softwarových projektů.

Otázky v dotazníku jsem formulovala tak, aby odpovědi byly srozumitelné a jednoznačné. Zvýšila jsem počet otázek, kde firmy mohli uvést více možností odpovědí. Použila jsem i otevřené otázky, abych se dozvěděla více o dané problematice, ne jen z konkrétních odpovědí, které jsem si sama zvolila.

Celý dotazník, byl formulován tak, aby nezabral více jak 10 minut. Dotazníky jsem rozeslala elektronickou formou firmám po celé České republice, zabývajících se vývojem a realizací softwarových projektů. Celková doba výzkumu trvala 5 týdnů, z toho 1 týden zabralo vytvoření a rozesílání dotazníků firmám, 3 týdny pak probíhal samotný výzkum.

Vzhledem k počtu otázek, kde mohli firmy uvést více odpovědí, jsem se nesetkala s tím, že by neporozuměli některé z otázek. Naopak vzhledem k velkému počtu rozeslaných dotazníků jednotlivým firmám, byl pro mě velkým problémem to, že firmy na dotazník buď nereagovali, nebo nemohli, poskytnout informace z důvodu mlčenlivosti k zákazníkovi.

Dotazník se skládá z 15 položek a obsahuje otázky:

- a) Uzavřené      9 otázek
- b) Alternativní    4 otázky
- c) Otevřené       2 otázky

Samotné znění dotazníku vypadá následovně:

Metody určování ceny softwarového projektu

1) Kolik zaměstnanců má Vaše firma?

- a) 1 pracovník
- b) 2 až 5 pracovníků
- c) 6 až 10 pracovníků
- d) 11 až 25 pracovníků
- e) 26 až 50 pracovníků
- f) 51 až 250 pracovníků
- g) více jak 250 pracovníků

*Záměr otázky:* Zjistit počet pracovníků ve firmě.

2) Jakým projektům se Vaše firma věnuje?

- a) www stránky (návrh, realizace)
- b) webové aplikace
- c) mobilní aplikace
- d) desktopové aplikace
- e) jiné (*uved'te jaké*)

---

*Záměr otázky:* Zjistit jakým softwarovým projektům se firma věnuje. V případě, že se firma zabývá i jinými projekty než sem uvedla v nabídce, má možnost svůj projekt dopsat.

3) Jakým způsobem Vaše firma určuje dobu (pracnost) realizace projektu?

- a) odhadem na základě zkušeností
  - b) porovnáním s již realizovanými projekty
  - c) metodou Use Case Point
  - d) metodou Funkčních bodů
  - e) pomocí počtu řádků zdrojového kódu
  - f) jinak (*uved'te jak*)
-

*Záměr otázky:* Zjistit jaké konkrétní techniky, metody firma používá při odhadování úsilí vyvíjeného softwarového projektu.

4) Používá Vaše firma softwarové nástroje pro kalkulaci?

a) Ano (*uved'te jaké*)

b) Ne

*Záměr otázky:* Zjistit, zda firma používá pro kalkulaci nějaké softwarové nástroje. Pokud používá, zjistit jaké.

5) Popište, co při kalkulaci ceny za projekt posuzujete a co upřednostňujete?

*Záměr otázky:* Zjistit co je zdrojem pro tvorbu odhadů a na základě čeho firma odhaduje úsilí, tedy stanovuje cenu vyvíjeného projektu.

6) Na základě čeho odhadujete pracnost a tím pádem stanovujete cenu?

a) dle funkčních bloků

b) dle člověkohodin

*Záměr otázky:* Zjistit podle čeho firma stanovuje cenu vyvíjeného projektu.

7) Co všechno zahrnuje Vaše firma při kalkulaci do projektu?

a) pouze cenu

b) pouze dobu realizace

c) cenu i dobu realizace

*Záměr otázky:* Zjistit co firma zahrnuje do kalkulace projektu.

8) Co všechno používá vaše firma, jako vstupní zdroje při odhadování projektů?

a) neúplný soubor požadavků

b) detailní soubor požadavků

c) detailní uživatelské rozhraní

d) kompletní architekturu systému

*Záměr otázky:* Zjistit co firma využívá jako vstup při stanovení odhadu vyvíjeného projektu.

9) Kolik zaměstnanců firmy se podílí na zpracování odhadu vyvíjeného projektu?

- a) 1 až 2 zaměstnanci
- b) 3 až 5 zaměstnanců
- c) 6 až 10 zaměstnanců
- d) více jak 10 zaměstnanců

*Záměr otázky:* Zjistit počet zaměstnanců, kteří se podílí na tvorbě odhadů.

10) Jsou osoby podílející se na odhadování ovlivněny nějakými vedlejšími zdroji?

- a) Ano
- b) Spíše ano
- c) Spíše ne
- d) Ne

*Záměr otázky:* Zjistit zda je firma hned na začátku odhadování omezena nějakými vedlejšími vlivy (cena, čas, počet zaměstnanců, apod.).

11) Provádíte před každý nový projekt úvodní sezení?

- a) Ano
- b) Spíše ano
- c) Spíše ne
- d) Ne

*Záměr otázky:* Zjistit zda firma před každým projektem provádí úvodní sezení se všemi zaměstnanci, kde projednávají jak odhad vyvíjeného projektu, tak celkový vývoj.

12) Kolik procent projektů není předáno klientovi v domluvený, termín?

- a) 0 až 25 %
- b) 26 až 45 %
- c) 46 až 75 %
- d) 75 až 100 %

*Záměr otázky:* Zjistit kolik procent projektů se dostane do takové fáze vývoje, že firma není schopna je předat zákazníkovi v domluvený, termín.

13) Kolik procent projektům bývá podhodnoceno?

- a) 0 až 25 %
- b) 26 až 45 %

- c) 46 až 75 %
- d) 75 až 100 %

*Záměr otázky:* Zjistit kolik procent projektů musí firma podhodnotit, tak aby nepřekročila stanovený rozpočet na začátku.

14) Co je nejčastějším důvodem podhodnocení projektů?

---

*Záměr otázky:* Zjistit, co vede firmu k tomu, že musí projekt podhodnotit.

15) Je Vaše firma ochotna se dále na projektu spolupodílet a poskytnou mi doplňující informace o Vašich projektech (*např. poskytnout „Use Case Model“, cenovou kalkulaci projektu, skutečnou dobu realizace, ...*)?

- a) Ano
- b) Ne

*Záměr otázky:* Cílem bylo požádat firmu o další materiály k výzkumu.

## 4.2 Metodika statistického zpracování výsledků

Výsledky mého dotazníkového šetření jsou uváděny v absolutních číslech a v procentech. Pro lepší chápání jsou některé tabulky doplněny i grafy. Všechna data byla zpracována elektronicky ve statistickém programu Statistica, cz7.

K vyhodnocení četnostních tabulek jsem využila počítačové vyhodnocení Chí-kvadrátu ( $\chi^2$ ) pro kvalitativní proměnné s hladinou pravděpodobnosti 0,01 % ( $p < 0.01$ ). [11]

Vzorec pro výpočet testovacího kritéria Chí-kvadrát je:

$$\chi^2 = \sum \frac{(P-O)^2}{O} \quad (31)$$

kde

P	Pozorovaná četnost
O	Očekávaná četnost

Výsledná hodnota Chí-kvadrátu vyjadřuje velikost rozdílu mezi skutečnou a stanovenou nulovou hypotézou. [11] Výsledná hodnota Chí-kvadrátu byla porovnána s kritickou hodnotou testovacího kritéria Chí-kvadrát pro aktuální hladinu významnosti a počet stupňů volnosti. Počet stupňů volnosti se vypočítá podle vztahu:

$$f = (r - 1) \cdot (s - 1) \quad (32)$$

kde

r	Počet řádků v kontingenční tabulce
s	Počet sloupců v kontingenční tabulce

### 4.3 Stanovené hypotézy

Byly stanoveny 3 hypotézy, které jsem ponechala v platnosti nebo naopak jsem se je snažila zvrátit.

Stanovené hypotézy pro výzkum jsou:

**Hypotéza č. 1:** *Firmy používají standardní algoritmy pro stanovení ceny projektu.*

U této hypotézy jsem vycházela z předpokladu, že firmy při stanovení ceny vyvíjeného projektu používají standardní algoritmy pro kalkulaci (metoda Use Case Point, apod.).

Otázky k hypotéze: 4,5,6,7,9

**Hypotéza č. 2:** *Softwarové firmy v ČR jsou schopni stanovit cenu aplikace (projektu) správně.*

U této hypotézy jsem vycházela z předpokladu, že firmy si dokáží vypracovat kvalitní plán vývoje, tak že hned na začátku projektu dokáží stanovit kvalitní cenu projektu, aniž by na konci museli cenu měnit nebo v horším případě projekt podhodnocovat.

Otázky k hypotéze: 4, 9,10,11,12,14,15

**Hypotéza č. 3:** *Velké softwarové firmy provádí odhadování pomocí standardních algoritmů častěji než menší softwarové firmy.*

Tato hypotéza mi měla pomoci odhalit, zda velké softwarové firmy provádí odhady pomocí standardních algoritmů mnohem častěji než firmy s menším počtem zaměstnanců. Jako velké firmy jsem zde uvažovala firmy, které zaměstnávají 25 a více zaměstnanců.

Otázky k hypotéze: 1,2,4

## 5 VÝSLEDKY VÝZKUMNÉ ČÁSTI A JEJICH ANALÝZA

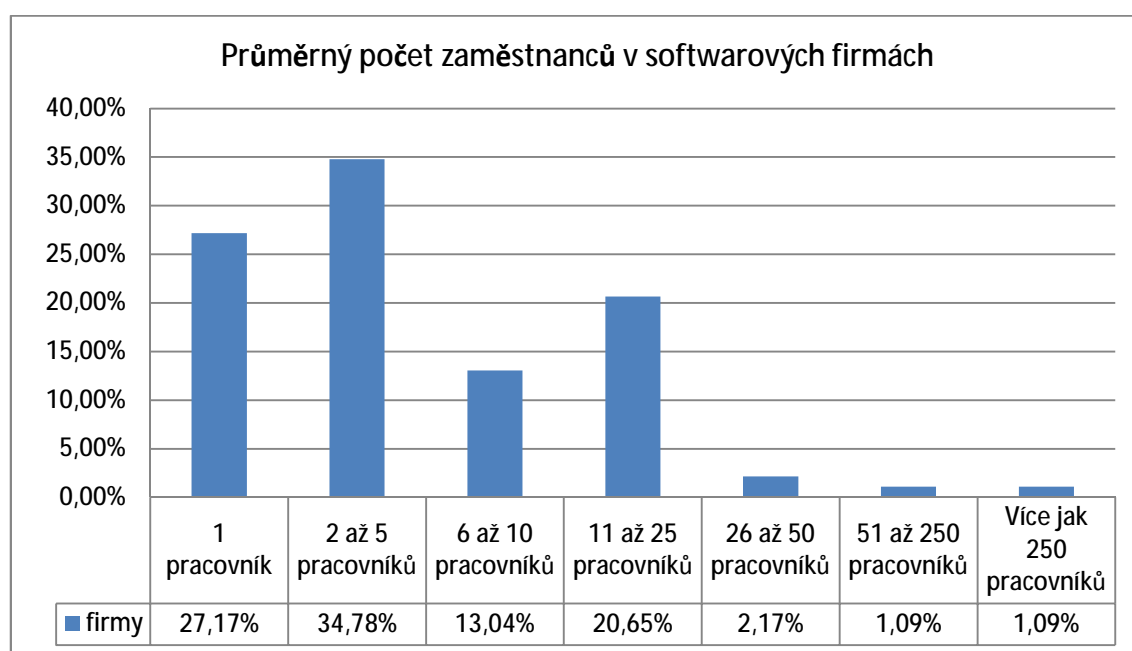
### 5.1 Vyhodnocení dotazníků

Výsledky dotazníkové šetření získané od respondentů budou uváděny v následujících v tabulkách a grafech. Hodnoty v tabulkách budou uváděny v číslech absolutní četnosti a relativní četnosti (%).

#### Průměrný počet zaměstnanců v softwarových firmách ČR

**Tabulka 25:** Počet zaměstnanců [vlastní zdroj]

Průměrný počet zaměstnanců ve firmě		
Kategorie	Absolutní četnost [počet]	Relativní četnost [%]
1 pracovník	25	27,17
2 až 5 pracovníků	32	34,78
6 až 10 pracovníků	12	13,04
11 až 25 pracovníků	19	20,65
26 až 50 pracovníků	2	2,17
51 až 250 pracovníků	1	1,09
více jak 250 pracovníků	1	1,09



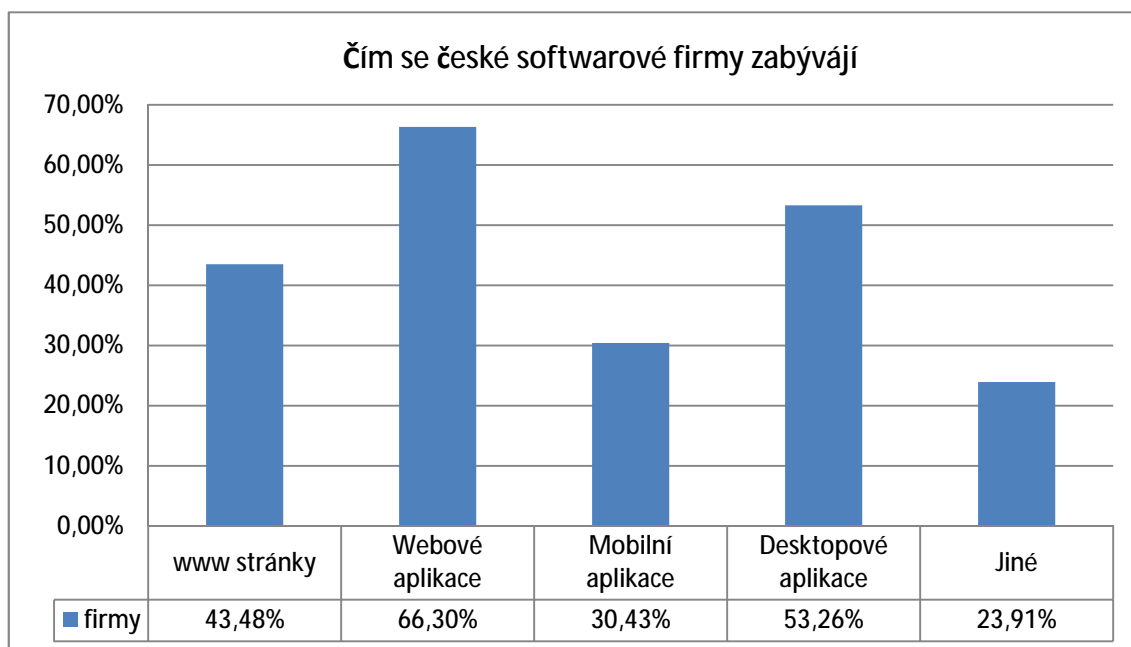
*Graf 1: Průměrný počet zaměstnanců v softwarových firmách [vlastní zdroj]*

Z celkového počtu 92 firem, zaměstnává 2 až 5 pracovníků 32 firem. Což je více než  $\frac{1}{4}$  z celkového počtu. Další čtvrtinu tvoří firmy, kde je zaměstnán pouze 1 pracovník. V absolutní hodnotě je to 25 firem. Na třetím místě se umístily firmy, které zaměstnávají 11 až 25 pracovníků. Celkem je to 19 firem, tedy necelá  $\frac{1}{4}$ . Poslední čtvrtinu tvoří firmy, které zaměstnávají 6 až 10 a 26 a více pracovníků.

### Čím se softwarové firmy v ČR zabývají

**Tabulka 26:** Oblasti vývoje jednotlivých firem [vlastní zdroj]

Projekty softwarových firem v ČR		
Kategorie	Absolutní četnost [počet]	Relativní četnost [%]
www stránky	40	43,48
Webové aplikace	61	66,30
Mobilní aplikace	28	30,43
Desktopové aplikace	49	53,26
Jiné	22	23,91



*Graf 2: Oblasti vývoje v jednotlivých firmách [vlastní zdroj]*

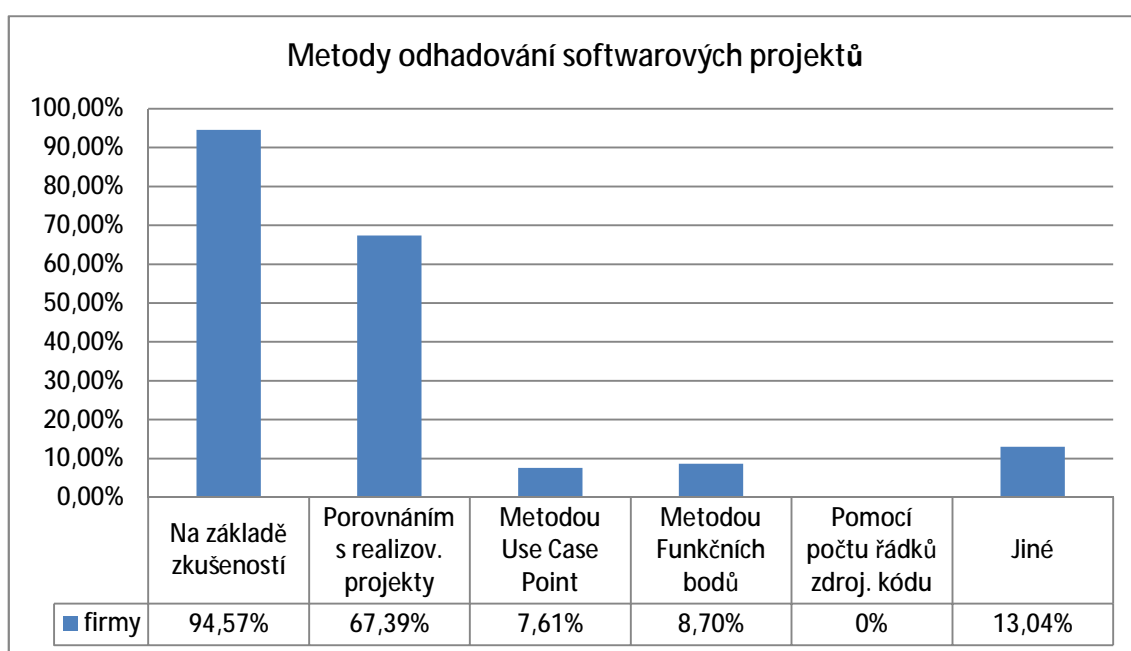
Zde jsme zjišťovali, čím se české softwarové firmy zabývají. Firmy mohli k odpovědi využít více možností. Více jak  $\frac{1}{2}$  oslovených firem odpověděla, že působí v oblasti vývoje

webových aplikací, celkem to bylo 61 firem. Další velmi oblíbenou oblastí firem je vývoj desktopových aplikací. Této oblasti se věnuje 49 firem (53,26 %). O něco méně je firem, které se zabývají tvorbou webových stránek, jejíž tvorbou se zabývá okolo 40 firem (43,48 %). Ostatní firmy se zabývají tvorbou speciálního softwaru a mobilními aplikacemi, které se dají ještě dnes považovat za méně rozšířené softwary.

### Způsoby odhadování softwarových projektů jednotlivých firem v ČR

**Tabulka 27:** Metody odhadování [vlastní zdroj]

Způsoby odhadování softwarových projektů v ČR		
Kategorie	Absolutní četnost [počet]	Relativní četnost [%]
Odhadem na základě zkušeností	87	94,57
Porovnáním s již realizovanými projekty	62	67,39
Metodou Use Case Point	7	7,61
Metodou Funkčních bodů	8	8,70
Pomocí počtu řádků zdrojového kódu	0	0
Jiné	12	13,04



*Graf 3: Metody odhadování softwarových projektů [vlastní zdroj]*

Zde jsme zjišťovali, podle čeho firmy provádí odhad pracnosti vyvíjeného projektu. Firmy mohli vybrat více z nabízených možností. Celkem překvapivé bylo, že 87 firem (97,57 %), nepoužívá v první řadě žádné metody pro odhad pracnosti. Obvykle nepoužívají žádné složité metodiky, ale vychází z předchozích zkušeností. Je tedy vidět, že hodně firem si neuvědomuje, že se jedná o neformální metody pro odhad pracnosti, která často vede k nepřesným odhadům. Na další pozici se umístili firmy, které kromě odhadů na základě zkušeností používají i formální metody pro odhad pracnosti. Z toho většinu odhadů provádí pomocí analogie prováděné porovnáním s již realizovanými projekty (62 firem) a menší počet odhadů provádí, pomocí metody Use Case Point a Funkčních bodů. A žádný z respondentů neodpověděl, že by používal metodu odhadu na základě počtu řádků zdrojového kódu.

### Počet firem využívající softwarové nástroje při kalkulaci

**Tabulka 28:** Počet firem, které používají softwarové nástroje pro stanovení odhadu [vlastní zdroj]

Počet firem využívající softwarové nástroje pro kalkulaci		
Kategorie	Absolutní četnost [počet]	Relativní četnost [%]
Ano	7	7,61
Ne	85	92,39

Je zřejmé, že v tomto pohledu mají firmy jasno. Více jak 90 % (85 firem) z nich odpovědělo, že nevyužívají žádné softwarové nástroje při kalkulaci vyvíjeného projektu. U sedmi firem, které odpověděli ano (softwarové nástroje používají), jsem zjišťovala, jaké nástroje skutečně používají. Dostala jsem pouze dvě odpovědi, a to program „MS Excel“ a „tabulkový kalkulátor“. Zde je vidět, že firmy pokud už nějaký softwarový nástroj používají, nepoužívají žádné speciální nebo vlastní kalkulátory, ale pouze klasické programy, které jsou dostupné všem, a každý se s nimi někdy setkal.

### Co firmy posuzují a co upřednostňují při kalkulaci projektů

Firmy můžeme rozdělit na 4 typy:

**1. typ:** Firma si nevede žádné statistiky skutečně strávené času nad jednotlivými projekty. Při tvorbě nabídek vychází z následujících předpokladů, že zákazník to chce levně a co

možno nejdříve. Snaží se nabídnout cenu, která je pro zákazníka přijatelná a cenu více méně odhadují na základě dosavadních zkušeností a reakcí na jeho cenové nabídky.

**2. typ:** Firma provádí hierarchický rozpad projektu na menší části a pro ty následně určuje kvalifikovaným odhadem cenu dle zkušeností např. ze zkušeností z dosud realizovaných projektů. K těmto odhadovaným cenám si přičte odhadované náklady za kompletaci projektu a vývoj.

**3. typ:** Firma odhadne pracnost v hodinách a nastaví nějakou hodinovou sazbu, kterou pak snižuje s velikostí projektu.

**4. typ:** Firma posuzuje rozsah uživatelského rozhraní, množství funkcí, jejich složitost a speciální přání klienta. Zohledňuje také úplnost konečných výstupů, které zákazník požaduje. Dále vychází s předchozí zkušenosti se zákazníkem a znalosti problematiky projektu na straně zákazníka. Na základě toho pak stanoví vstupy a z nich určí výslednou cenu.

### Jaké parametry používají firmy pro odhad pracnosti

**Tabulka 29:** Parametry podle kterých firmy v ČR kalkulují [vlastní zdroj]

Podle jakých parametrů dělají softwarové firmy odhady		
Kategorie	Absolutní četnost [počet]	Relativní četnost [%]
Funkčních bodů	20	21,74
Člověkohodin	72	78,26

Z celkového počtu 92 firem, odpovědělo 72 firem (78,26 %), že provádí odhady pracnosti pomocí člověkohodin a 20 firem (21,74 %) firem pomocí funkčních bodů. To vypovídá o tom, že provádět odhady pomocí člověkohodin je mnohem rychlejší a jednodušší forma než pomocí funkčních bodů.

### Co všechno firmy zahrnují do kalkulace

**Tabulka 30:** Co všechno firmy zahrnují do odhadu vyvíjeného projektu [vlastní zdroj]

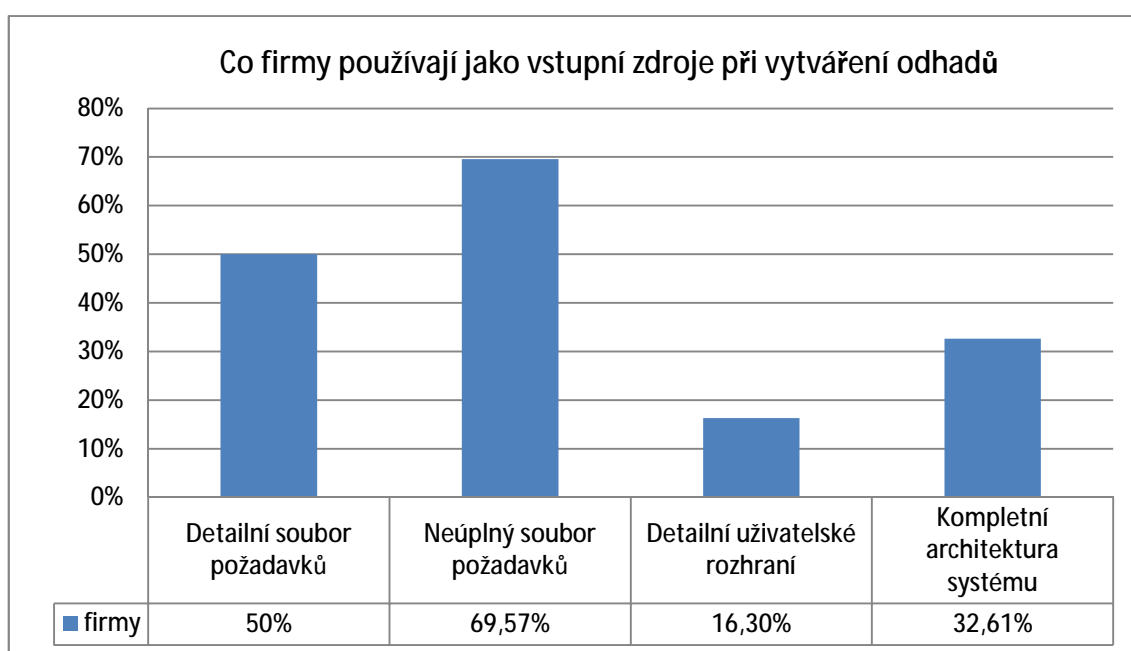
Co všechno zahrnuje softwarová firma při kalkulaci do projektu		
Kategorie	Absolutní četnost [počet]	Relativní četnost [%]
Pouze dobu realizace	6	6,52
Pouze cenu	10	10,87
Cenu i dobu realizace	76	82,61

Bylo zjištěno, že většina dotazovaných firem (82,61 %) zahrnuje do odhadu vyvíjeného projektu jak cenu, tak i dobu realizace.

### Co firmy používají jako vstupní zdroje při vytváření odhadů

**Tabulka 31:** Jednotlivé vstupy používané pro odhadování nákladů [vlastní zdroj]

Jaké vstupy jsou používány pro odhadování nákladů		
Kategorie	Absolutní četnost [počet]	Relativní četnost [%]
Detailní soubor požadavků	46	50,00
Neúplný soubor požadavků	64	69,57
Detailní uživatelské rozhraní	15	16,30
Kompletní architektura systému	30	32,61



**Graf 4:** Jednotlivé vstupy používané pro odhadování nákladů [vlastní zdroj]

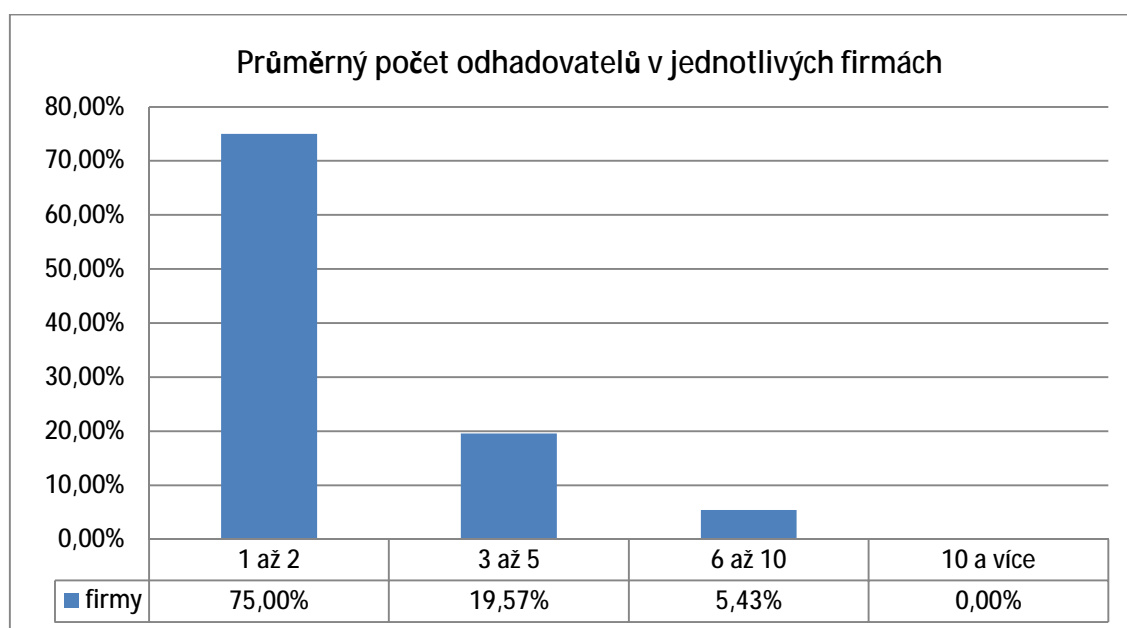
Zde jsem zjišťovala, co firmy používají jako vstup při vytváření odhadu vyvíjeného projektu. Firmy si mohli vybrat z více možností. Nadpoloviční většina dotazovaných firem (69,57 %) odpověděla, že při vytváření odhadů má nejčastěji k dispozici neúplný soubor požadavků od zákazníka. To je většinou způsobeno tím, že zákazník sám dobře neví, co chce nebo samotné problematice projektu dostatečně nerozumí. Na druhé pozici se umístilo kupodivu hodně firem (50 %), které mají k dispozici detailní soubor požadavků.

V tomto případě je pravděpodobné, že se jedná o projekty menšího rozsahu s nižší složitostí. Naopak minimum firem má k dispozici detailní uživatelské rozhraní (16,30 %) a kompletní architekturu systému (32 %).

### Průměrný počet odhadovatelů v jednotlivých firmách

**Tabulka 32:** Počet zaměstnanců podílejících se na vytváření odhadů v jednotlivých firmách [vlastní zdroj]

Počet odhadovatelů v softwarových firmách		
Kategorie	Absolutní četnost [počet]	Relativní četnost [%]
1 až 2	69	75,00
3 až 5	18	19,57
6 až 10	5	5,43
10 a více	0	0,00



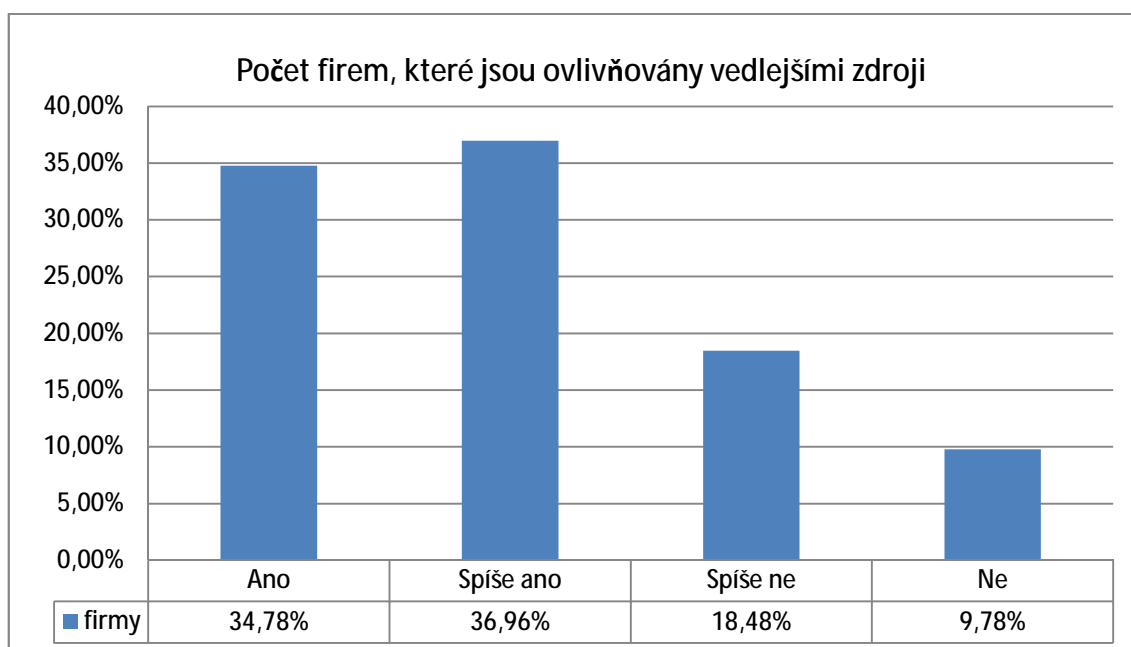
*Graf 5: průměrný počet odhadovatelů ve firmách [vlastní zdroj]*

Bylo zjištěno, že většina dotazovaných firem využívá pro stanovení odhadu pracnosti minimální počet odhadovatelů. Většinou se tento počet pohybuje okolo 1 až 2 odhadovatelů (75 %). Dá se tedy předpokládat, že odhady tvoří buď projektový manažer sám, nebo spolu s dalšími vedoucími pozic.

**Počet firem, které jsou ovlivňovány vedlejšími zdroji**

**Tabulka 33:** Počet firem, které jsou omezovány vedlejšími zdroji při stanovení odhadů.  
[vlastní zdroj]

<b>Počet firem, které jsou ovlivňovány vedlejšími zdroji</b>		
<b>Kategorie</b>	<b>Absolutní četnost [počet]</b>	<b>Relativní četnost [%]</b>
Ano	32	34,78
Spíše ano	34	36,96
Spíše ne	17	18,48
Ne	9	9,78

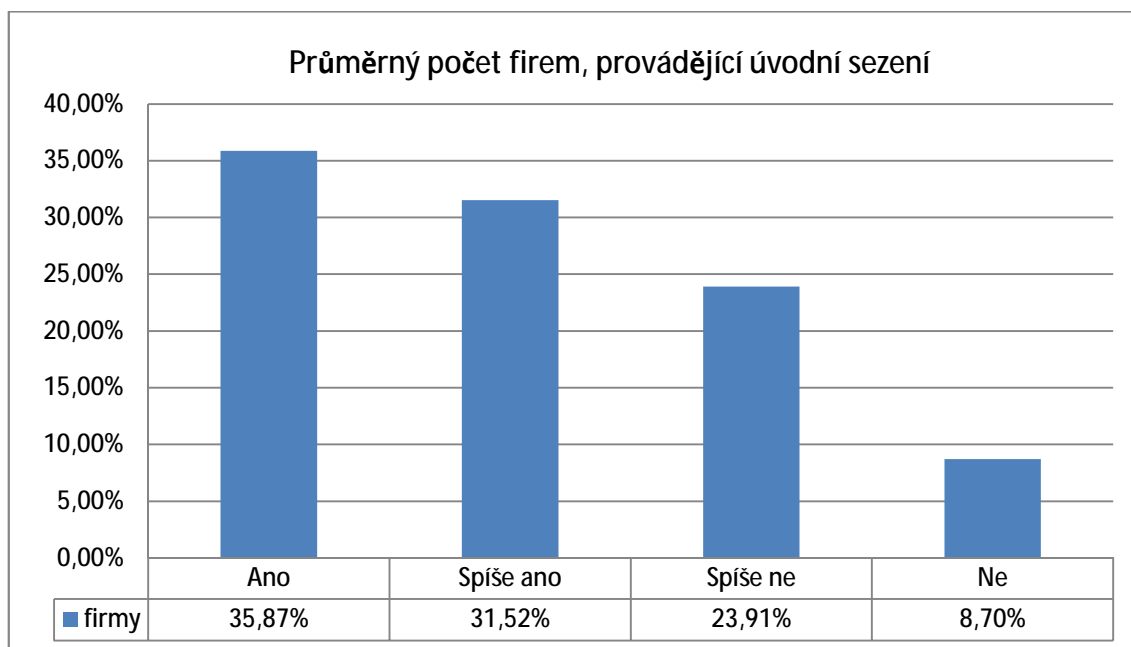


*Graf 6: počet firem, které jsou ovlivňovány vedlejšími vlivy [vlastní zdroj]*

Na otázku, zda jsou ihned na začátku odhadování firmy omezeny vedlejšími vlivy (čas, cena, počet zaměstnanců) odpovědělo více jak 2/3 firem, „ano“ nebo „spíše ano“. Tento výsledek, ale není nijak překvapivý. Protože jak je známo i z jiných projektů, zákazník to chce levně a co nejrychleji.

**Průměrný počet firem, provádějící úvodní sezení u nového projektu****Tabulka 34:** Počet firem provádějící vstupní sezení [vlastní zdroj]

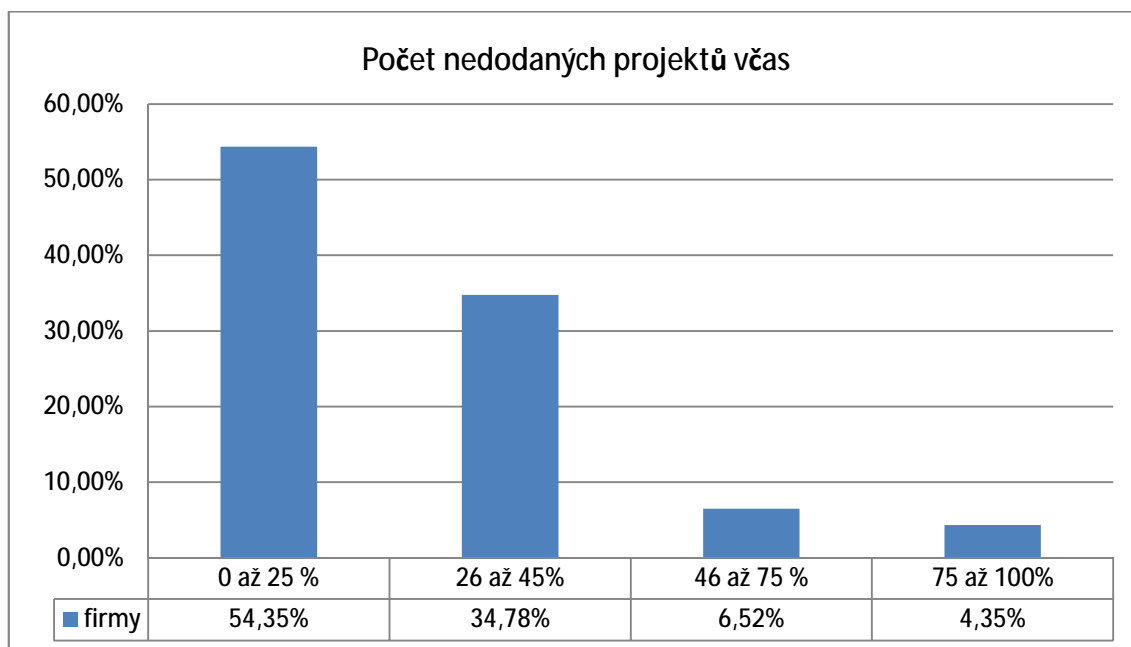
Průměrný počet firem, provádějící úvodní sezení před každým projektem		
Kategorie	Absolutní četnost [počet]	Relativní četnost [%]
Ano	33	35,87
Spíše ano	29	31,52
Spíše ne	22	23,91
Ne	8	8,70

**Graf 7:** Průměrný počet firem provádějící úvodní sezení [vlastní zdroj]

Zde jsme zjišťovali, zda firmy provádí před každým novým projektem úvodní sezení, kde všichni členové týmu projednávají realizaci celého projektu. Nadpoloviční většina dotazovaných firem odpověděla „ano“ a „spíše ano“. Nicméně, celkem překvapivé bylo, že 22 firem odpovědělo „spíše ne“ a 8 firem „ne“. Jde tedy konstatovat, že tyto firmy si neuvědomují, že úvodní sezení je právě jedním z nejdůležitějších faktorů, pokud chtějí dosáhnout požadovaných výstupů.

**Počet projektů, které nejsou firmy schopni předat zákazníkovi včas****Tabulka 35:** Počet nedodaných projektů včas [vlastní zdroj]

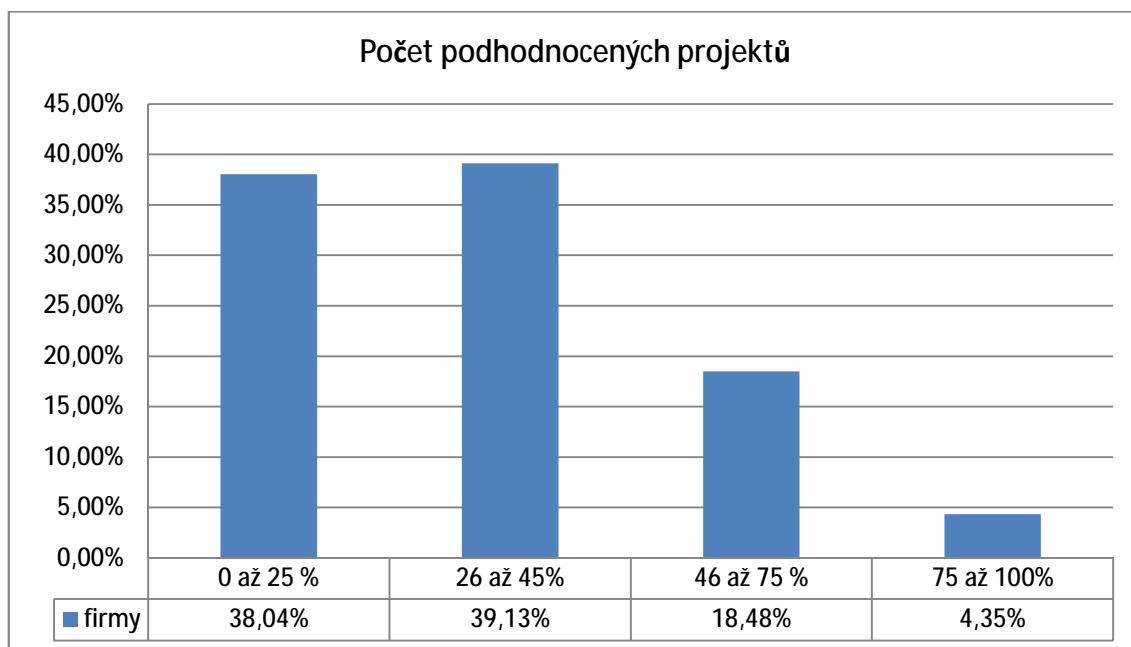
<b>Kolik procent projektů nebývá dodáno včas?</b>		
<b>Kategorie</b>	<b>Absolutní četnost [počet]</b>	<b>Relativní četnost [%]</b>
0 až 25 %	50	54,35
26 až 45 %	32	34,78
46 až 75 %	6	6,52
75 až 100 %	4	4,35

*Graf 8: počet nedodaných projektů včas [vlastní zdroj]*

Zde jsem zjišťovala, zda jsou firmy schopny dodat projekt zákazníkovi včas. Celkem překvapivé bylo, že více jak polovina firem nemá nebo má minimální problémy s dodáním projektů včas i přesto že v prvotní fázi projektu nepoužívá žádné metody pro odhad pracnosti a provádí odhady na základě zkušenosti. Kupodivu jde tedy říci, že i firmy, které nepoužívají formální metody odhadování, jsou schopné projekt klientovi předat ve stanoveném termínu.

**Průměrný počet projektů, které firma podhodnotí****Tabulka 36:** Počet podhodnocených projektů [vlastní zdroj]

<b>Kolik procent projektů bývá podhodnoceno?</b>		
<b>Kategorie</b>	<b>Absolutní četnost [počet]</b>	<b>Relativní četnost [%]</b>
0 až 25 %	35	38,04
26 až 45 %	36	39,13
46 až 75 %	17	18,48
75 až 100 %	4	4,35

*Graf 9: Počet podhodnocených projektů [vlastní zdroj]*

Zde jsem zjišťovala, kolik procent projektů musí jednotlivé firmy podhodnotit, aby nepřekročili původní odhad stanovený na začátku projektu. Zde už, ale výsledek byl horší. Sice 35 firem odpovědělo, že nemá nebo má pouze minimální problémy v tomhle směru. Což by byl skvělý výsledek v případě, kdyby problémy s podhodnocením nemělo dalších 36 firem u 26 až 45 % projektů, 17 firem u 46 až 75 % projektů a 4 firmy u 75 až 100 % projektů. Z toho plyne, že více jak polovina firem není schopna se držet stanoveného plánu na začátku projektu. Firmy si však neuvědomují, že jakákoliv aktivita navíc ubírá čas potřebný na produktivní práci a prodlužují projekt více, než by trval při přesném odhadu a plánu.

### Nejčastější důvod podhodnocení

U firem, které odpověděli, že projekty podhodnocují, mě zajímalo, jaký je jejich nejčastější důvod, který je k tomu vede. Na velké množství odpovídajících firem bylo překvapivé, že se mi vrátilo pouze pět odpovědí, a to špatný odhad původního vývoje, nejasné požadavky na začátku projektu, změny požadavků od zákazníka, dlouhodobá spolupráce se zákazníkem a velikost projektu. Zpravidla firmy uvádějí, že čím větší projekt, tím větší podhodnocení.

## 5.2 Stanované hypotézy a jejich ověření

**Hypotéza č. 1:** *Firmy používají standardní algoritmy pro stanovení ceny projektu.*

Otázky k hypotéze: 4,5,6,7,9,13

Tato hypotéza byla vyvrácena. Výsledky z dotazníkových šetření poukazují na fakt, že většina softwarových firem nepoužívá pro odhad pracnosti projektu žádné standardizované metody. Veškeré informace k tomuto problému byly vyhodnoceny, na základě dotazníkových otázek 4, 5, 6, 7, 9, 13, které se vztahovaly ke zjištění, zda softwarové firmy provádí odhady pomocí standardních metod odhadování.

Bylo zjištěno, že více jak 90 % nepoužívá v první řadě žádné metody pro odhad pracnosti, ale vychází z předchozích zkušeností.

Svědčí o tom, i to že více jak 80 % firem, provádí odhady na základě MD (člověkohodin), které jsou snadno spočítatelné oproti FP, které jsou hůře spočítatelné, ale naopak z hlediska funkcionality mnohem více korespondují s realitou.

O tom, že firmy nepoužívají, standardní metody odhadování svědčí i další hodnoty získané z dotazníkového šetření. Bylo zjištěno, že nadpoloviční většina dotazovaných firem při vytváření odhadů má k dispozici neúplný soubor požadavků od zákazníka. Většinou to však není problém firmy, ale tím, že zákazník sám dobře neví, co chce nebo samotné problematice projektu dostatečně nerozumí. Pokud, ale nemáme dostatečné informace, nejsme schopni v žádném případě stanovit správně odhad nebo provádět odhad podle složitějších metod odhadování.

Potvrzuje to i další otázka, zda jsou firmy schopny předat projekt zákazníkovi včas. Bylo ale zjištěno, že více jak 90 % firem provádějící odhady na základě vlastních zkušeností, nejsou schopny projekt dodat zákazníkovi včas. Je to způsobeno často, tím že firmy si neu-

vědomují, že odhady prováděné na základě zkušeností vedou k nepřesným odhadům, které pak pro firmu znamenají další práci navíc, která ubírá čas potřebný na vývoj projektu, a tím se projekt prodlužuje a trvá mnohem déle, než kdyby firma provedla přesný odhad.

Navíc více jak 30 % dotazovaných firem odpovědělo, že při odhadování upřednostňují to, že pokud by brali skutečnou dobu strávenou nad jedním projektem (včetně datové analýzy, návrhu grafického rozhraní) a průměrnou hodinovou mzdu v ČR, tak se dostanou na cenu řádově v desítkách tisíců korun a více. Zákazník, to chce ale levně a proto více méně odhadují na základě dosavadních zkušeností a reakcí na jejich cenové nabídky, aby zákazník byl spokojen.

Teoreticky se tento způsob odhadu dá přirovnat k tzv. spirálovému přístupu vývoje softwaru. Tím, ale začíná kolečko, kdy zákazník říká „a ještě bych potřeboval...“, „to by mělo fungovat jinak...“ a doba realizace se může několikanásobně prodloužit. Z tohoto důvodu, si firmy často nepřipravují žádné, Use case modely, atd., ale tvoří přímo software. I toto svědčí o tom, že většina softwarových firem provádí odhady na základě dosavadních zkušeností nebo na základě porovnání s již realizovanými projekty.

Z výzkumu je tedy zřejmé, že pokud si firmy nepovedou žádné statistiky skutečně stráveného času nad jednotlivými fázemi projektu a budou při tvorbě nabídek vycházet jen z toho, že zákazník to chce levně a co možno nejdříve. A budou tedy nabízet cenu, která je pro zákazníka přijatelná a odhadovat ji pouze na základě dosavadních zkušeností, nikdy nedosáhnou správných odhadů a budou nuceni projekt podhodnocovat.

Taky to, že firmy provádějí odhady na základě dosavadních zkušeností a provádí převážně odhady pomocí MD, než pomocí FP, vypovídá o tom, že provádět odhady tímto způsobem je mnohem rychlejší a jednodušší forma, než pomocí funkčních bodů a standartních metod odhadování.

**Hypotéza č. 2:** *Softwarové firmy v ČR jsou schopni stanovit cenu aplikace (projektu) správně.*

Tato hypotéza byla vyvrácena. Na základě analýzy odpovědí z dotazníkových šetření vyplynulo, že softwarové firmy nejsou schopni provést odhady správně. Informace o tomto problému byly vyhodnoceny na základě dotazníkových otázek č. 4, 9, 10 a 12, 14, 15, které se vztahovaly ke zjištění, zda softwarové firmy provádí odhady správně. Ve všech těch-

to otázkách bylo viditelné, že softwarové firmy mají hodně daleko k tomu, aby byli schopni stanovit správný odhad.

Bylo zjištěno, že většina softwarových firem, v první řadě nepoužívá žádné metody pro odhad pracnosti, ale vychází z předchozích zkušeností. Z toho více jak 60 % firem (podhodnocení více jak 25 %) provádí podhodnocování projektů, jen proto, aby nepřekročili původně stanovený odhad. Už toto, ale svědčí o tom, že firmy nejsou schopni se držet stanoveného plánu na začátku projektu. Většinou to však není vinou firmy, ale zákazníka, který není schopen na začátku projektu poskytnout dostačující informace pro správný odhad projektu.

I to, že firmy nejsou schopni provést odhad správně, dokládají otázky týkající se zpracování odhadu. Přesto, že většina firem, na začátku každého projektu provádí úvodní sezení, podílí se na něm celkem málo odhadců. Většinou se na odhadu podílí dva zaměstnanci, což firmám určitě pomůže při stanovení kvalitnějšího odhadu, ale je to názor proti názoru, proto mnohem účinnější, by bylo, kdyby firmy prováděli odhady minimálně pomocí tří odhadců, kteří mohou mezi sebou odhady porovnat, a na základě odlišností prodiskutovat a odhad zopakovat.

V případě, že bychom uvažovali rozdělení firem, na malé (méně jak 25 zaměstnanců) a velké firmy (více jak 25 zaměstnanců) tak malé firmy projekty podhodnocují, ale na druhou stranu, jsou schopni je dodat zákazníkovi včas. Naopak velké firmy projekty většinou nepodhodnocují, ale nejsou schopni je včas dokončit.

Navíc nadpoloviční většina firem má při odhadování projektu k dispozici neúplný soubor požadavků od zákazníka. Ani sebelepší firma, pokud nebude mít úplné nebo skoro úplné požadavky od zákazníka, není schopna provést odhad správně.

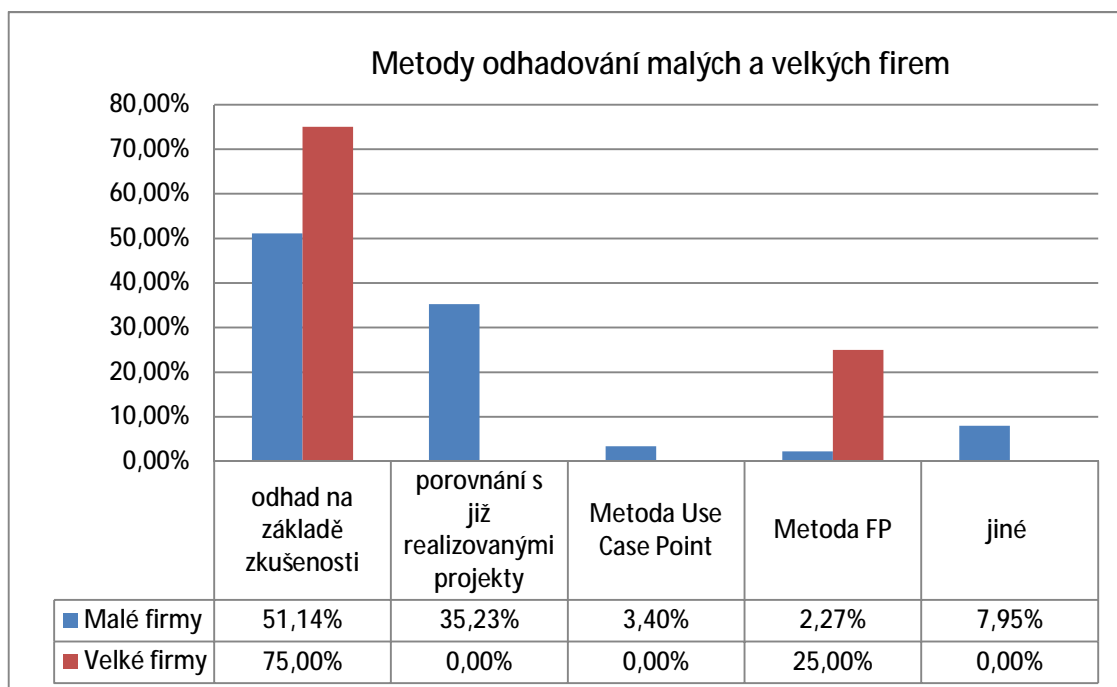
Z výzkumu je tedy zřejmé, že firmy si často neuvědomují, že špatně stanovený odhad, znamená další aktivitu navíc, která ubírá čas potřebný na produktivní práci, a prodlužuje projekt více, než by trval při přesném odhadu a plánu.

**Hypotéza č. 3:** *Velké softwarové firmy provádí odhadování pomocí standartních algoritmů častěji než menší softwarové firmy.*

Otázky k hypotéze: 1,2,4,13,14

Tabulka 37: Způsob odhadování malých a velkých sw firem [vlastní zdroj]

Způsob odhadování malých a velkých firem				
	Malá firma		Velká firma	
Profese	Absol. četnost [počet]	Relat. četnost [%]	Absol. četnost [počet]	Relat. četnost [%]
Využívají metody odhadování	43	48,86	1	25
Provádí odhady na základě zkušeností	45	51,14	3	75
<b>Celkem</b>	<b>88</b>	<b>100</b>	<b>4</b>	<b>100</b>



Graf 10: metody odhadování malých a velkých sw firem v ČR [vlastní zdroj]

Tato hypotéza byla opět vyvrácena. Na základě analýzy odpovědí z dotazníkových šetření vyplynulo, že velké softwarové firmy neprovádí odhady pomocí standardních algoritmů častěji než malé softwarové firmy. Informace o tomto problému byly vyhodnoceny na základě dotazníkových otázek č. 1,2,4, 6,7,9,13, které se vztahovaly ke zjištění, zda softwarové firmy provádí odhady správně. Ve všech těchto otázkách bylo viditelné, že velké softwarové firmy používají mnohem méně standardní metody pro odhad než malé softwarové firmy.

Bylo zjištěno, že okolo 50 % malých firem využívá standartní metody odhadování, na rozdíl od velkých firem, kde využívá standartní metody pouze 25 % firem. Z toho téměř 36 % malých firem využívá metodu odhadování na základě porovnání s již realizovanými projekty a 4 % firem používají metodu, Use Case Point a metodu Funkčních bodů. Velké softwarové firmy pokud už nějakou metodu využívají tak používají většinou metodu Funkčních bodů.

O tom, že malé softwarové firmy využívají častěji standartní metody odhadování, svědčí i to, že více jak 30 % malých firem provádí odhady pomocí FP, zatímco u velkých firem jeto pouze 10 %.

Častější využívání metod odhadování dokládá i otázka týkající se dodání vyvíjeného projektu zákazníkovi včas. Kde bylo zjištěno, že žádná velká firma, v rámci dotazníkového šetření, není schopna předat projekt zákazníkovi včas. Zatím co více jak 50 % malých firem ano. I to svědčí o tom, že firmy, které provádí odhady na základě standartních metod, mají mnohem větší šanci provést odhad správně než firmy, které provádí odhady na základě zkušeností.

Navíc většina velkých softwarových neprovádí odhady podle složitých parametrů, ale většinou při odhadu vycházejí z toho, že zákazník to chce levně a co nejrychleji a nevedou si ani žádné statistiky o tom, kolik času nad daným projektem strávili, apod. Zatímco malé softwarové firmy, se snaží neprovádět odhady pouze na základě toho, že zákazník to chce levně, ale snaží se do návrhu zahrnout i další parametry, jako složitost funkcí, přibližnou dobu strávenou nad projektem, úplnost požadavků od zákazníka, apod. I to svědčí o tom, že malé firmy mnohem častěji využívají standartní metody odhadování.

Z výzkumu, tedy vyplývá, že velké softwarové firmy si především zakládají na tom, aby odhad byl co nejrychlejší a nejjednodušší, zatímco malé softwarové firmy si zakládají na tom, aby odhad byl co možno nejpřesnější. Velké softwarové firmy si, ale při tom neuvědomují to, že tento způsob odhadu, který používají, mnohem častěji vede, k nepřesným odhadům, které pro firmu znamenají další práci navíc.

### 5.3 Shrnutí výzkumné části

Výsledky výzkumu poskytly řadu zajímavých informací týkajících se metodiky odhadování softwarových firem v České republice. Na výzkumu se více méně podílely malé softwarové firmy, které zaměstnávají 1 až 25 pracovníků. Zbytek pak, tvořily velké softwarové

firmy, které zaměstnávají 25 a více zaměstnanců. Z toho více jak jedna polovina firem se zabývá vývoje webových aplikací a desktopových aplikací. Jen malé množství firem se zabývá speciálními mobilními aplikacemi a tvorbou webových stránek.

Většina těchto společností obvykle nepoužívají žádné složité metodiky odhadování, ale vychází z předchozích zkušeností. Kromě odhadů na základě zkušeností používají i formální metody pro odhad pracnosti. Z toho většinu odhadů provádí buď pomocí analogie prováděné porovnáním s již realizovanými projekty anebo pomocí metody Use Case Point a Funkčních bodů.

V České republice se setkáváme s nejrůznějšími typy softwarových firem, z nichž každá využívá jinou metodiku odhadování. Máme firmy, které si nevedou žádné statistiky skutečně stráveného času nad jednotlivými projekty. Při tvorbě nabídek vychází z předpokladů, že zákazník to chce levně a co možno nejdříve a snaží se, nabídnou zákazníkovi cenu, která je pro něj přijatelná. Na druhé straně máme ale firmy, které neposuzují jen to, že zákazník to chce levně, ale posuzují jak rozsah uživatelského rozhraní, množství funkcí tak i složitost a speciální přání klienta. Do posudku zahrnují také úplnost konečných vstupů, které zákazník požaduje a předchozí zkušenosti se zákazníkem a jeho znalost problematiky projektu. Na základě toho pak stanovují vstupy a z nich výslednou cenu. Často jsou to právě firmy, které používají k odhadu matematické modely, díky kterým mají možnost získat mnohem kvalitnější odhady než předchozí skupina firem, která většinu odhadů provádí na základě zkušeností.

Bylo také zjištěno, že nadpoloviční většina firem, při vytváření odhadu má k dispozici neúplný soubor požadavků. Většinou to není chybou firmy, ale zákazníka, že neví přesně, co chce nebo dané problematice nerozumí.

Dalším zajímavým zjištěním bylo, že většina firem provádí úvodní sezení, do něhož je zapojeno jen několik málo osob (1-2 lidi), které jsou většinou při vytváření odhadu ovlivněny vnějšími vlivy. Dá se tedy říct, že na odhadu se podílí buď sám projektový manažer, nebo spolu s dalšími vedoucími pozicemi.

I přesto, že většina společností neaplikuje žádné formální metody pro odhad pracnosti, a odhady většinou provádí na základě vlastních zkušeností odhadované v MD (člověkohodin), nemají značné problémy s tím, že by vyvíjený software nebyli schopni předat zákazníkovi včas. Naopak mnohem častěji firmy mají problémy s tím, že projekty musí podhod-

notit. Většinou je to způsobeno tím, že na začátku provedli špatný odhad, nebo neměli od zákazníka dostatečné úvodní informace.

Na závěr jsem vyhodnotila tři stanovené hypotézy, které jsem na základě provedeného výzkumu zamítla.

## 6 NÁVRH METODIKY PRO SOFTWARE FIRMY

### Současný stav v softwarových firmách v ČR

Software firmy je možné rozdělit na dvě skupiny, a to malé a velké software firmy. V současné době většina softwarových firem zabývající se vývojem softwarových projektů nepoužívá pro odhad ceny projektu žádné standardizované metody, ale provádí odhady přímo pomocí expertního úsudku na základě dosavadních zkušeností nebo pomocí obecné analogie, založené na porovnání vyvíjeného projektu s podobnými projekty vytvořenými v minulosti. Firmy si neuchovávají nic, kromě zdrojových kódů, takže odhad čistě závisí na dobré paměti a zkušenosti odhadců. [6] Toto chování, ale není nijak překvapující, protože zpravidla se jedná o aplikace menší velikosti, kdy firmy využívají agilní metody vývoje, a každé nasazení složitější metody by způsobilo, že by zvolený způsob vývoje aplikace nebyl flexibilní k zákazníkovi.

S modely větších softwarových firem, se v dnešní době moc nesetkáváme. Je to dáno tím, že i společnost s několika desítkami zaměstnanců, je schopna fungovat, bez jakýchkoliv velkých problémů. [6] Je to způsobeno tím, že zaměstnanci firmy na sebe tzv. vidí, navíc většinu projektů řeší, plánují a odhadují plynule podle potřeby. Problém nastává až tehdy, kdy se vedoucí firmy domnívá, že pokud přijme dvojnásobek až trojnásobek vývojářů, tak se výdělek firmy zdvojnásobí, až ztrojnásobí. [15] Pokud, ale nebudou provedeny důležité kroky pro řízení takového počtu osob, tak opak bude pravdou. Prakticky to vede k tomu, že vše co do teď fungovalo, postupně přestává fungovat. Dá se konstatovat, že firmy se většinou rozrůstají za účelem toho, aby dostali zakázky větší velikosti. Tyto zakázky jsou, ale zpravidla mnohem složitější, než zakázky, se kterými se rozrůstající firmy doposud setkaly. [14] V takto rostoucích firmách, je velmi důležité, aby kladly důraz na dobré rozpracování plánu, řízení vývoje a na odhad pracnosti a další náležitosti důležité pro správný vývoj aplikace.

### 6.1 Návrh metodiky odhadování pro software firmy

V této kapitole se budu zabývat návrhem metodiky pro odhadování softwarových projektů, který jsem navrhla po nastudování této problematiky na internetu, z materiálů firem a odborných knih. Metodika je navrhována tak, aby ji bylo možné používat v jakékoliv software firmě.

Nejdříve se budu zabývat hodnocením odhadovacích metod a dalších atributů potřebných pro tvorbu jednotné metodiky pro odhadování projektů a poté se budu zabývat samotnou metodikou odhadování, kterou jsem zde navrhla.

### **Výběr metodiky pro měření velikosti projektu**

Základem každé metodiky je způsob, jakým se měří velikost projektu. [15] Zde jsem se rozhodovala mezi měřením pomocí řádků kódů a měřením pomocí funkčních bodů. Řádky kódů jsou snadno spočítatelné, ale za to v prvotních fázích vývoje mohou být nejasné a matoucí.

Protože v dnešní době většina softwarových firem zabývajících se vývojem softwarových aplikací nepoužívá pro odhad ceny projektu žádné standardizované metody, ale provádí odhady přímo pomocí expertního úsudku. Proto návrh metodiky pro měření velikosti projektu nebyl proveden podle žádného základního pravidla. [14] Já jsem zvolila měření pomocí funkčních bodů. Hlavním důvodem byl funkční pohled na velikost projektu, který mnohem více koresponduje s realitou než LOC. Jedním z dalších důvodů výběru byla možnost, že funkční body lze spočítat před implementační fází, což u počítání LOC nelze provést.

### **Výběr metodiky pro odhadování**

Při výběru metod odhadování jsem vycházela z toho, že jsem zvolila jako způsob měření, měření pomocí funkčních bodů. Proto jsem vybírala mezi metodami, které provádí odhady pomocí funkčních bodů. Vzhledem odlišnosti jednotlivých fází a nároků na přesnost odhadu jsem pro každou fázi vývoje zvolila jiné metody. [14]

V první fázi vývoje, kdy nemáme skoro žádné informace, a tedy nároky na přesnost jsou minimální, jsem vybrala metodu odhadování pomocí zástupce, kterou nazýváme fuzzy logika.

V další fázi máme už o trochu více informací a již celistvý názor na to, co se bude v rámci vývoje řešit a provádět. [15] Proto bych vybrala metody uvedené v kapitole 2.3, 2.5, 2.7. Tyto metody jsem zvolila z toho důvodu, že požadují mnohem více informací a mají větší nároky na přesnost než fuzzy logika, kterou jsem zvolila pro první fázi vývoje. Z hlediska toho, že metody mají při odhadování rozdílné požadavky na vstupní data, jsem nechala volbu metody na tvůrci odhadu.

V další fázi, před samotnou implementací dokážeme vypočítat již samotné funkční body, proto jsem pro tuto fázi vývoje vybrala metodu FPA. [14]

Samotný převod odhadu je potom dobré realizovat prostřednictvím historických dat.

### **Historická data**

Další velmi významnou část každého odhadování tvoří historická data z minulých projektů. V rámci návrhu metodiky odhadování jsem se rozhodla ustálit sběr historických dat z projektů. [14] To bude provedeno tak, že bude vytvořena databáze, kam budou data z projektů ukládána.

U projektů budou ukládána tyto historická data:

- Všeobecná data o projektu
- Časové intervaly jednotlivých etap projektu
- Rozsah projektu ve FP
- Odhad práce v jednotlivých etapách projektu a celého projektu

Takto získána data budou využívána pro výpočet koeficientů pro převádění rozsahu projektu na práci, pro výpočet nákladů a pro plánování vývoje projektů. [14] Dále mohou být využívány i pro přezkoumání správnosti odhadování a pro případné změny nastavení metod.

#### **6.1.1 Návrh metodiky odhadování**

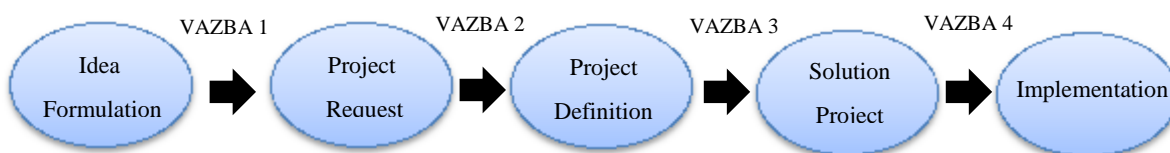
V této kapitole se budu zabývat návrhem metodiky odhadování pro softwarové firmy v ČR. Základní prvek, který bude metodika využívat, je měření velikosti projektu uceleným postupem. Díky tomu jsme schopni porovnávat velikosti a měřit účinnost vývoje. Navíc je podkladem pro metody odhadování využívané v jednotlivých etapách vývoje projektu. [15]

Velikost projektu bude měřena pomocí FP a následně upravena na náklady a dobu nezbytnou pro vývoj projektu prostřednictvím koeficientů získaných z dat předešlých projektů.

Další důležitou částí metodiky je i databáze, kam jsou ukládána data z předešlých projektů a projektů, které právě běží. [15] Data z databáze jsou potom využívána při odhadování velikosti projektu, tak při převádění velikosti projektu na práci a časovou náročnost. Součástí databáze je i seznam předpřipravených východisek, který zahrnuje zpracovaná řešení s detailním popisem řešení, nákladů a doby důležité pro vývoj projektu.

## Metodika projektování

Návrhem byla i metodika odhadování a jejich vzájemné propojení. Jednotlivé fáze vývoje a jejich propojení, které jsem pro návrh této metodiky použila, můžete vidět na obr. 4. [14]



Obr. 4: Životní cyklus projektu [14]

### Idea Formulation

Činnost: Vytvoření vstupního dokumentu, v němž jsou popsány jednotlivé části projektu a definován cíl vývoje.  
Vytvoření odhadu celého projektu, který bude sloužit, jako základ pro prioritizaci projektů.

Vazba: 1

Přesnost: Projekt:  $\pm 100$  %

Použití: Pro prioritizaci projektů

### Project Request

Činnost: Aktualizace vstupního dokumentu a úprava odhadu celého projektu.

Vazba: 2

Přesnost: Projekt:  $\pm 100$  %

Fáze:  $\pm 30$  %

Použití: Pro detailnější odhad celého projektu.  
Pro odhad fáze PD.

### Project Definition

Činnost: Vytvoření dokumentu Definition, který v sobě zahrnuje detailnější popisy požadavků.  
Rozčlenění projektu na jednotlivé úlohy a jejich následné sloučení do slotů a iterací.  
Vytvoření odhadu, který bude za konkrétních okolností a rozpracovanosti projektu co nejpřesnější.

Vazba: 3

Přesnost: Projekt:  $\pm 50$  %

Fáze:  $\pm 20$  %

Použití: Pro odhad celého projektu, který nám poslouží jako základna pro schvalování projektu.  
Pro odhad fáze PS.

### Project Solution

- Činnost: Vytvoření dokumentu Solution, který nám poslouží jako IT návrh vývoje projektu.  
Rozčlenění projektu na detailnější seznam úloh.  
Popis řešení jednotlivých úloh a stanovení jejich nákladů.
- Vazba: 4
- Přesnost: Projekt:  $\pm 10\%$   
Fáze:  $\pm 10\%$
- Použití: Pro odhad rozpočtu fáze Implementation.

### Implementation

- Činnost: Vývoj, testování, a předání softwaru zákazníkovi

Zde jsou stručně popsány jednotlivé etapy vývoje. U každé etapy vývoje je uvedena její předpokládaná činnost, její propojení s další etapou vývoje a předpokládaná přesnost odhadů. Přesnosti odhadů, které jsou zde uváděny, je potřeba brát s určitou rezervou, protože jsou to hodnoty pouze předpokládány z hlediska úrovně odhadu, nikoliv podloženy zkušenostmi s navrženou metodikou odhadu. [15] Konkrétnější hodnoty, lze získat až po delší době využívání, této navržené metodiky odhadu.

V dalších kapitolkách, se zaměřuji na detailnější popis jednotlivých etap vývoje softwaru. Vždy u každé etapy uvádím, co je definováno, co o dané aplikaci v dané etapě vývoje víme, a co z toho lze použít pro odhadování. [14] Dále taky uvádím, souhrn vstupů a výstupů, a seznam všech odhadů, které jsou v dané fázi vývoje odhadnuty.

### **Fáze Idea Formulation**

V této fázi se stanoví cíl projektu a provede se odhad nákladů na celý projekt, který bude použit, jako základ pro prioritizaci projektů. [14] Vzhledem k tomu, že v této fázi vývoje nemáme skoro žádné informace a naopak máme poměrně dost neznámých, je dobré brát provedený odhad jen jako informativní s poměrně minimální přesností.

**Tabulka 38:** Metodika fáze Idea Formulation [14]

Metodika odhadu na celý software:	Metoda založená na expertním úsudku a využívající historická data.
Vstupy:	Vstupní dokument
Je vymezen:	Výchozí cíl projektu
Výstupy:	Počáteční odhad nákladů na vyvíjený software

Odhad celého softwaru

Pro odhad vyvíjeného softwaru použijeme metodu odhadování pomocí funkčních bodů, která je blíže specifikována v kapitole 2.6.

**Fáze Project Request**

Tato fáze je využívána ke kontrole změn mezi fázemi Idea Formulation a Idea Request. Kde posuzování dopadu a změn je prováděno stejným způsobem, který byl použit ve fázi Idea Formulation.

**Tabulka 39:** Metodika fáze Project Request [14]

Metodika odhadu na celý software:	Metoda založená na expertním úsudku a využívající historická data shodná s metodou odhadu provedeného ve fázi IF.
Metodika odhadu na fázi PD:	Odhad bude určen z odhadu na celý software a kontrolován prostřednictvím expertního odhadu.
Vstupy:	Aktualizovaný vstupní dokument
Je vymezen:	Výchozí cíl projektu
Výstupy:	Aktualizovaný odhad nákladů na vyvíjený software a odhad na fázi PD.

Odhad celého softwaru

Pokud bude provedena nějaká změna ve vstupním dokumentu, potom odhad celého projektu se musí provést znovu a to stejnou metodou jako ve fázi IF.

V případě že žádná změna v dokumentu se neprovede, není potřeba znovu odhad celého projektu dělat, a odhad provedený ve fázi IF bude platit. [15]

Stejně jako u fáze IF je dobré brát odhad jako informativní s poměrně malou přesností.

Odhad fáze Project Definition

Z odhadu celého projektu, který byl vytvořen v této fázi nebo ve fázi IF (případně, že nedošlo ke změně požadavků), dostaneme odhad rozsahu projektu ve FP. [14] Nezbytné náklady na další vývoj projektu vypočítáme prostřednictvím koeficientů historických dat. Nakonec získaný odhad rozpočítáme mezi jednotlivé role a rozčleníme je na interní a externí. [16]

Výsledný odhad nám potom poslouží, jako základna pro schvalování projektu.

**Fáze Project Definition**

V této fázi se pomocí intenzivní spolupráce menší skupiny vývojářů a odhadců, [16] vytvoří dokument Definition, který v sobě zahrnuje detailnější popisy požadavků a objasňuje téměř všechny problémy v projektu (např. dobu vývoje, velikost projektu, přínosy, výdaje, apod.)

Dále je potřeba rozčlenit projekt na jednotlivé úlohy a ty následně sloučit do jednotlivých slotů a iterací. [15] V případě, že je to možné, je zásadní pro získání kvalitnějšího odhadu, vybrat, analyzovat a posoudit alternativní řešení nebo navrhnout jedno z nich.

Z takto získaných výstupů se provede odhad, který nám potom slouží, jako základna pro schvalování projektu. [14] Snahou je vytvořit odhad, který bude za konkrétních okolností a rozpracovanosti projektu co nejpřesnější.

Z odhadu celého projektu je pak vytvořen odhad pro další fázi vývoje projektu (SD).

**Tabulka 40: Metodika fáze Project Definition [14]**

Metodika odhadu na celý software:	Pro odhad použijeme metodu odhadování pomocí funkčních bodů.
Metodika odhadu na fázi SD:	Odhad je tvořen metodou založenou na expertním úsudku a kontrolován s odhadem celého projektu
Vstupy:	Dokument Definition Odhad celého projektu z fáze PR
Je vymezen:	Procesy (jejich četnost, doba trvání, návaznosti mezi sebou, apod.) Počet vývojářů Počet dat Požadavky (dostupnosti, podpory, historie dat, archivace, apod.)
Výstupy:	Přibližný odhad celého projektu s přesností +/- 50 % Přesný odhad další etapy (SD) s přesností +/- 20 %

### Odhad celého projektu

Pro odhad vyvíjeného softwaru použijeme metody odhadování pomocí funkčních bodů. Odhad můžeme provádět na celý scope projekt. [15] Je-li, ale projekt rozdělen na jednotlivé úlohy, je přívětivější odhadnout každou úlohu jednotlivě. V případě, že zvolíme tento způsob odhadování, je dobré každou úlohu odhadovat jiným způsobem. Například, se to dá provést tak, že jednu úlohu vypočítáme přímo a další odhadneme. Také velmi užitečné je, když využijeme seznam předdefinovaných řešení (pokud jej máme k dispozici), které přiřadíme k jednotlivým úlohám a pomocí nich je odhadneme. [14]

Protože, zde můžeme volit mezi více metodami odhadování pomocí FP, je potřeba, aby odhadci pověřeni tímto odhadem, [14] byli obeznámeni s jednotlivými metodami a z hlediska dostupných dat a atributů jednotlivých metod, vybrali tu nejvhodnější pro daný odhad.

Pro nastudování jednotlivých metod, bych jako metodiku odhadování doporučila, odhadování založené na analogii, u co nejvíce možných úloh, které lze touto metodou odhadnou. Mezi ostatní úlohy, bych pak zařadila ty, u kterých lze přímo odhadnout počet FP a u zbylých úloh, bych volila způsob odhadu, z hlediska dostupných dat.

Výsledný odhad, nám potom poslouží pro návrh další fáze Solution Design a jako podklad pro kontrolu správnosti projektů. [15] V případě, že vzniknou velké odlišnosti, budou tyto rozdíly zdůvodněni.

### Odhad fáze Solution Design

Z odhadu celého projektu, který byl vytvořen v této fázi, dostaneme odhad rozsahu projektu ve FP. Nezbytné náklady na další fázi vývoje vypočítáme prostřednictvím koeficientů historických dat. Nakonec získaný odhad rozpočítáme mezi jednotlivé role a rozčleníme je na interní a externí.

Výsledný odhad, nám potom poslouží pro návrh další fáze Implementation a jako podklad pro kontrolu správnosti projektů. [15] V případě, že vzniknou velké odlišnosti, budou tyto rozdíly zdůvodněni.

Z odhadu celého projektu provedeného v této fázi získáme velikost projektu.

### **Fáze Solution Design**

Z odhadu celého softwaru, který byl vytvořen v této fázi, dostaneme odhad velikosti softwaru ve funkčních bodech. [14] Tento odhad následně prostřednictvím koeficientů histo-

rických dat, převedeme na počet dnů práce, potřebné pro jednotlivé role. Výsledek pak bude, z hlediska navrhnutému IT řešení, který byl vytvořen na začátku této fáze, zkontrolován. V případě, že vzniknou velké odlišnosti, budou tyto odlišnosti zdůvodněny.

**Tabulka 41:** Metodika fáze Solution Design [14]

Metodika odhadu na celý software:	Pro zpřesnění odhadu v předešlé fázi použijeme metodu odhadu pomocí počtu FP nebo jakoukoliv metodu odhadování pomocí FP.
Metodika odhadu na fázi SD:	Odhad je tvořen metodou založenou na expertním úsudku a kontrolován s odhadem celého projektu
Vstupy:	Dokument Solution Odhad počtu FP z předešlé fáze
Je vymezen:	Rozpracované požadavky, procesy a řešení Rozčlenění projektu na detailnější seznam úloh
Výstupy:	Odhad na další fázi

### Odhad fáze Implementation

Pro odhad fáze Implementation použijeme metody odhadování pomocí funkčních bodů. Odhad můžeme provádět na celý scope projekt, protože se, ale jedná už o mnohem rozsáhlejší etapu vývoje, je přívětivější projekt rozčlenit na jednotlivé úlohy, a ty pak odhadovat jednotlivě.

V případě, že zvolíme tento způsob odhadování, je dobré každou úlohu odhadovat jiným způsobem. [14] Například, se to dá provést tak, že jednu úlohu vypočítáme přímo a další odhadneme. Také velmi užitečné je, když využijeme seznam předdefinovaných řešení (pokud jej máme k dispozici), které přiřadíme k jednotlivým úlohám a pomocí nich je odhadneme.

Protože, zde můžeme volit mezi více metodami odhadování pomocí FP, je potřeba, aby odhadci pověřeni tímto odhadem, byli obeznámeni s jednotlivými metodami a z hlediska dostupných dat a atributů jednotlivých metod, vybrali tu nejvhodnější pro daný odhad.

Pro nastudování jednotlivých metod, bych jako metodiku odhadování doporučila, odhadování, pomocí metody FP, kde lze přímo odhadnout počet FP, a to u co nejvíce možných úloh, u kterých lze tuto metodu použít. [15] U zbylých úloh, bych volila způsob odhadu, z hlediska dostupných dat.

### Fáze Implementation

Po provedení všech fází projektu, se prostřednictvím metody odhadování pomocí FP, vypočte reálná velikost vyvíjeného softwaru a získanou hodnotu s hodnotou spotřebovaných nákladů uložíme do připravené databáze. [14] Následně provedeme zhodnocení odhadů jednotlivých etap vývoje, a to z hlediska jejich přesnosti. V případě, že dojde k významnějším než běžným odlišnostem, zjistíme příčiny jejich vzniku a navrhneme vhodnou korekci odhadování.

### Přepočítání velikosti projektu na náklady

V rámci historických dat můžeme určit průměrnou cenu 1 FP v člověkohodinách nebo v nákladech, budeme-li se domnívat, že vývoj podobných projektů bude realizován s podobnou efektivností a kompozicí nákladů. [14]

Poněvadž v dnešní době máme ve firmách k dispozici rozdílné projekty, které mohou být v produktivitě odlišné, je mnohem užitečnější nepočítat koeficienty pro převod FP z každého projektu, ale jen z projektů s podobnými atributy.

#### Výpočet koeficientů

- 1) Koeficient pro převod velikosti projektu na náklady [14]

$$kNP = \frac{\sum_{k=1}^N \frac{NK_k \cdot D_{k,p}}{FP_k}}{\sum_{k=1}^N D_{k,p}} \quad (32)$$

kde

NK	Náklady na projekt
FP	Počet FP
D	Podobnost projektů

- 2) Koeficient pro převod velikosti projektu MD [14]

$$kMD = \frac{\sum_{k=1}^N \frac{MD_k \cdot D_{k,p}}{FP_k}}{\sum_{k=1}^N D_{k,p}} \quad (33)$$

kde

MD	Počet MD na projekt
FP	Počet FP
D	Podobnost projektů

Převod FP na náklady a člověkohodiny

Jakmile máme vypočítané koeficienty pro před velikosti projektu, vypočítáme z FP náklady a počet MD. [14]

Vzorce pro výpočet nákladů a MD:

## 1) Výpočet nákladů

$$cNK = FP \cdot kNP \quad (34)$$

kde

kNP	koeficient pro převod velikosti projektu na náklady
FP	Počet FP

## 2) Výpočet MD

$$cMD = FP \cdot kMD \quad (35)$$

kde

kMD	koeficient pro převod velikosti projektu na MD
FP	Počet FP

## ZÁVĚR

Odhadování je běžná lidská činnost, se kterou se každý z nás někdy setkal a každý z nás tvoří odhady, které se od sebe liší formou a účelem.

I když je to hodně překvapující, dá se říct, že velký počet vývojářů, kteří provádí odhady softwarových projektů, nemají žádnou průpravu k tomu, jak správně odhadovat, ale provádí odhady pouze z vlastní zkušenosti. Dá se říct, ale že to není zas tak špatný způsob odhadování, alespoň na úrovni malých projektů. Z hlediska přesnosti je, ale stále nejlepší provádět odhady na základě nějaké standardní metody, kde je větší šance získání přesných odhadů. Ovšem jen tehdy je-li metoda vhodně zvolena a správně využita.

Hlavním cílem diplomové práce bylo zhodnotit problematiku odhadování v softwarových firmách v ČR. Zabývala jsem se tím, jaké metody odhadování využívají, co při vytváření odhadů upřednostňují, podle čeho odhady provádí, a zda jsou schopny provádět odhady správně. Pro získávání a shromažďování informací k danému tématu jsem zvolila dotazník, který jsem sestavila po prostudování dané problematiky v odborné literatuře a na internetu. Dotazník jsem následně rozeslala softwarovým firmám po celé české republice.

Výsledky dotazníkové šetření přinesly řadu zajímavých informací týkající se metodiky odhadování softwarových firem v ČR. Bylo zjištěno, že na výzkumu se podílely více méně malé softwarové firmy, které zaměstnávají méně jak 25 zaměstnanců. Zbytek, pak tvořili velké softwarové firmy, které zaměstnávají 25 a více zaměstnanců. Firmy, které mi odpověděly, se většinou zabývají vývojem webových aplikací a desktopových aplikací. Zbytek se pak zabývá speciálními mobilními aplikacemi a tvorbou webových stránek.

Dále bylo zjištěno, že většina softwarových firem nepoužívá žádné složité metodiky odhadování, ale vychází z předchozích zkušeností. Kromě odhadů na základě zkušeností používají i některé firmy formální metody pro odhad pracnosti. Většinou se jedná o provádění odhadů pomocí metody založené na analogii, tedy porovnáním s již realizovanými projekty nebo pomocí metody Use Case Point a metody Funkčních bodů.

V ČR republice se setkáváme s nejrůznějšími typy softwarových firem, z nichž každá využívá jinou metodiku odhadování. Přesně bylo zjištěno, že máme 4 typy firem. První typ představují firmy, které si nevedou žádné statistiky skutečně stráveného času nad jednotlivými projekty a při tvorbě nabídek vychází z předpokladů, že zákazník to chce levně a co možno nejdříve a snaží se, nabídnout zákazníkovi cenu, které je pro něj přijatelná. Na dru-

hé straně máme firmy, které neposuzují jen to, že zákazník to chce levně, ale posuzují jak rozsah uživatelského rozhraní, množství funkcí tak i složitost a speciální přání klienta. Do posudku pak zahrnují i úplnost konečných vstupů, které zákazník požaduje a předchozí zkušenosti se zákazníkem a jeho znalost problematiky projektu. Na základě toho pak stanovují vstupy a z nich výslednou cenu. Třetí typem jsou firmy, které nejdříve provádí rozpad projektu na menší části a ty následně odhaduje kvalifikovaných odhadem dle zkušeností, např. ze zkušeností z dosud realizovaných projektů. K těmto částem nakonec přičtou odhadované náklady za kompletaci projektu a vývoj. Poslední typem jsou potom firmy, které provedou odhad pracnosti v hodinách a nastaví nějakou hodinovou sazbu, kterou pak snižují s velikostí projektu.

Dále bylo zjištěno, že nadpoloviční většina softwarových firem, při vytváření odhadu má k dispozici neúplný soubor požadavků. Většinou tento problém, není chybou firmy, ale zákazníka, že sám dobře neví, co chce nebo dané problematice nerozumí.

Velmi zajímavým zjištěním bylo, že většina firem provádí úvodní sezení, do něhož je zapojeno jen několik málo osob (1-2 osoby), které jsou většinou při vytváření odhadu ovlivněny vnějšími vlivy. Dá se tedy říct, že na odhadu se podílí buď sám projektový manažer, nebo spolu s dalšími vedoucími pozicemi.

Bylo taky zjištěno, že většina softwarových firem i přesto, že neaplikuje žádné formální metody pro odhad pracnosti, a většinu odhad provádí na základě vlastních zkušeností v MD (člověkohodinu), nemají téměř žádné problémy s tím, že by vyvíjený software nebyli schopni předat zákazníkovi včas. Naopak větší problém firem je, že musí velmi často projekty podhodnocovat. Je to ve velkém množství firem způsobeno tím, že na začátku provedli špatný odhad, nebo neměli od zákazníka dostatečné úvodní informace.

Na závěr této části, jsme vyhodnotili tři stanovené hypotézy, které jsme na základě provedeného výzkumu zamítli. První hypotéza hovořila o tom, že softwarové firmy používají pro odhady standartní metody odhadování. Bylo ale zjištěno, že většina softwarových firem provádí odhady na základě zkušeností. Druhá hypotéza hovořila o tom, že softwarové firmy jsou schopni stanovit cenu projektu správně. I tato hypotéza byla zamítnuta. Dokládá to i odpověď, že většina firem, musí svoje projekty podhodnocovat. Už toto svědčí o tom, že firmy nejsou schopni stanovit cenu správně. Třetí a poslední hypotéza hovoří o tom, že velké softwarové firmy provádí odhadování pomocí standartních algoritmů častěji než menší softwarové firmy. I tato hypotéza, ale byla zamítnuta.

Dalším cílem práce bylo navrhnout metodiku pro odhadování softwarových projektů. Metodiku jsem navrhla tak, aby byla univerzální a byla více méně použitelná pro všechny softwarové firmy. Tato metodika zavádí jednotný systém sběru a údržby dat jednotlivých projektů. Odhadování velikosti projektů je prováděno pomocí FP, a proto v metodice zavádím metody odhadování založené na FP. V každé etapě vývoje jsou určeny metody odhadování, které odpovídají rozpracovanosti projektu a požadavkům na přesnost. Celá metodika je navržena tak, aby byla využitelná pro všechny softwarové firmy v ČR. Navíc metodiku jsem navrhla tak, aby byla srozumitelná a pochopitelná pro všechny uživatele, kteří by ji chtěli využívat.

Protože zde nejsou používána žádná konkrétní data, ale pouze data obecná, tak se dá říct, že přesnost metodiky se pohybuje dle provedené studie na hranici požadovaných mezí.

## CONCLUSION

Estimating is a normal human activity with which each of us had ever met, and each of us making estimates which differ by form and purpose.

Although it's very surprising, you might say that a large number of developers who makes estimates of software projects do not have any training on how to correctly estimate, but estimates performs only from their own experience. We can say it's not a bad way of estimating, at least at the level of small projects. In terms of accuracy, it is always best to make estimates based on a standard method, where is a greater chance of obtaining accurate estimates. However, only if the method is properly selected and properly used.

The main objective of the thesis was to evaluate the issue of estimation software companies in the CR. We dealt with what methods for estimating are used when trying to create estimates, by what according performs estimate, and whether they are able to make estimates correctly. For collecting information on the topic I chose a questionnaire which I prepared after studying the issue in scientific literature and on the Internet. I then sent a questionnaire to software firms all over Czech Republic.

The results of a survey brought a lot of interesting information on the methodology of estimating software companies in the CR. It was found that research contributed more or less small software firms that employ fewer than 25 employees. Rest, then formed the large software companies, which employ 25 or more employees. Companies that replied to us, mostly engaged in the development of web applications and desktop applications. The residue is then engaged in special mobile applications and creating web pages.

Furthermore, it was found that most software companies does not use any complicated estimating methodology, but comes from previous experience. Besides estimates based on experience also some companies use formal methods for estimating labor. Mostly the estimation method is based on an analogy, a comparison with the already implemented projects or by using Use Case Point method and Function Points Method.

In the Czech Republic we meet with a variety of software companies, each of which uses a different methodology for estimating. Precisely, it was found that there are 4 types of businesses. The first type consists of companies that do not keep any statistics actually spent time on individual projects and the creation of offers is based on the assumptions that the customer wants it cheaply and without delay and try to offer the customer a price that is

acceptable to him. On the other side we have companies that do not consider just that the customer wants it cheaply, but how to assess the extent of the user interface, multiple features and complexity and special wishes. Into then report include the completeness of the final inputs that the customer requires and previous experience with the customer and his knowledge of the project. On that basis, they set inputs and their final price. The third type are the companies that initially performs the disintegration of the project into smaller parts and then estimate the qualified estimate according to experience, e.g. from the experience of previously implemented projects. To these parts finally add the estimated cost for completion of the project and development. The last type is then firms that shall estimate the time consumption in hours and set an hourly rate, which then reduces with the size of project.

It was also found that the absolute majority of software companies in creating the estimate has an incomplete set of requirements. Mostly this problem is not the fault of the company but the customer who does not know what he wants or does not understand the issues.

An interesting observation was that most firms carrying out an initial session in which it is involved only a few people (1-2 people), which are mostly in creating estimate affected by external influences. We can therefore say that estimate is participating either alone project manager, or together with other management positions.

It was also found that most software companies, despite not apply any formal methods for estimating the labor intensity, and most estimates made on the basis of personal experience in the MD (man-hour), they have almost no problems with the software that they developed were not able to pass customer on time. Conversely, larger problem for companies is that they must often underestimate the projects. It's in a large number of companies caused by that at the beginning made a miscalculation or did not from customer sufficient initial information.

To conclude this section, we evaluated three stated hypotheses, which we based on our research declined. The first hypothesis talked about the fact that software companies use to estimates of standard estimation methods. But it was found that most software companies makes estimates based on experience. The second hypothesis talked about the fact that software companies are able to determine the cost of the project properly. Even this hypothesis has been rejected. This is evidenced by the answer that most companies should underestimate their projects. Already this suggests that companies are not able to set the

price correctly. The third and final hypothesis talking about that large software companies performs standard estimation algorithms often than smaller software companies. Even this hypothesis has been rejected.

Another goal was to propose a methodology for estimating software projects. I suggested a methodology so that was universal and was more or less applicable to all software companies. This methodology introduces a unified system for collecting and maintaining the data on individual projects. Estimating the size of projects is done by FP and, therefore, I introduce the methodology of estimation methods based on FP. In each phase of development they are determined by estimation methods that meet the requirements stage of the project and accuracy. The entire methodology is designed so as to be usable for all software companies in the CR. Additionally, the methodology I designed so as to be clear and understandable for all users who would like to enjoy it.

Since there are no specific data used, but only general data, so we can say that the accuracy of the methodology varies according to studies carried out at the border required limits.

## SEZNAM POUŽITÉ LITERATURY

- [1] Boehm, B. W., et al.: COCOMO II Model Definition Manual. USA, Los Angeles, University of Southern California 1999. International Function Points User Group: Function Point Counting Practices Manual: Release 4.1.1999.
- [2] Sedláčková, J.: Cenové odhady softwarových projektů. Masarykova univerzita v Brně, Fakulta informatiky, Brno, 2005. Diplomová práce.
- [3] Anda, B. Comparing Use Case based Estimates with Expert Estimates. Proc. of Empirical Assessment in Software Engineering (EASE), Keele, United Kingdom, April 8-10, 2002.
- [4] SMITH, John. The Estimation of Effort Based on Use Cases [online]. Somers : IBM, 2003 [cit. 2011-01-24]. Dostupné z WWW: <ftp://ftp.software.ibm.com/software/rational/web/whitepapers/2003/finalTP171.pdf>.
- [5] SILHAVY, Radek; SILHAVY, Petr; PROKOPOVA, Zdenka. Requirements Based Estimation Approach for System Engineering Projects. In: Innovations and Advances in Computing, Informatics, Systems Sciences, Networking and Engineering. Springer International Publishing, 2015. p. 467-472.
- [6] MCCONNELL, Steve. *Odhadování softwarových projektů: Jak správně určit rozpočet, termíny, zdroje*. Brno: Computers Press, a.s., 2006. ISBN 80-251-1240-3.
- [7] HUDEC, Aleš. *Požadavky zákazníků jako východisko pro určení rozsahu softwarového projektu* [online]. Pardubice, 2010 [cit. 2015-04-03]. Dostupné z: [https://dspace.upce.cz/bitstream/10195/39646/1/HudecA\\_PozadavkyZakazniku\\_SS\\_2010.pdf](https://dspace.upce.cz/bitstream/10195/39646/1/HudecA_PozadavkyZakazniku_SS_2010.pdf). Bakalářská práce. Fakulta ekonomicko-správní, Univerzita Pardubice.
- [8] SURŇÁK, Petr. *Odhadování softwarových projektů* [online]. Praha, 2010 [cit. 2015-04-03]. Dostupné z: <http://www.vse.cz/vskp/eid/20734>. Diplomová práce. Vysoká škola ekonomická v Praze.
- [9] KAŠČÁK, Pavol. *Odhady software založené na scénářích* [online]. Zlín, 2011 [cit. 2015-04-03]. Dostupné z: [http://digilib.k.utb.cz/bitstream/handle/10563/17348/ka%C5%A1%C4%8D%C3%A1k\\_2011\\_bp.pdf?sequence=1&isAllowed=y](http://digilib.k.utb.cz/bitstream/handle/10563/17348/ka%C5%A1%C4%8D%C3%A1k_2011_bp.pdf?sequence=1&isAllowed=y). Bakalářská práce. Univerzita Tomáše Bati ve Zlíně.

- [10] MACINKA, Václav. *Techniky stanovení nákladů softwarových projektů* [online]. Brno, 2009 [cit. 2015-04-03]. Dostupné z: [http://is.muni.cz/th/139909/fi\\_m/dp.pdf](http://is.muni.cz/th/139909/fi_m/dp.pdf). Diplomová práce. Masarykova univerzita.
- [11] VAHALÍK, Tomáš. 2013. Odhadujete pracnost projektu. In: *Systémový integrátor* [online]. [cit. 2015-05-11]. Dostupné z: <http://www.komix-inxmail.cz/upload/noviny2004c.pdf>
- [12] KRÁL, Miroslav. 2012. *Odhadování pracnosti IT projektů* [online]. [cit. 2015-05-11]. Dostupné z: [index.php/aip/article/download/16/11](http://index.php/aip/article/download/16/11)
- [13] *Metoda Delphi: Delfská metoda* [online]. 2010. [cit. 2015-05-11]. Dostupné z: <https://www.kvic.cz/soubor/1358/MetodaDELPHI.pdf>
- [14] CIPRA, Jan. *Metody pro odhadování pracnosti softwarových projektů ve vývoji*. Praha, 2009. Diplomová práce.
- [15] DANĚK, František. *Information system offer*. Praha, 2014.
- [16] DANĚK, František. *Interní Informační Systém: Nabídka pro xxx*. Praha, 2011.

**SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK**

API	Aplikační programové rozhraní
APM	Application Point Model
apod.	A podobně
AS	Počet potřebných pracovníků pro vývoj
COCOMO	Constructive Cost Model 1.1
COCOMO II	Constructive Cost Model 2.0
ČR	Česká republika
ECF	Faktor složitosti prostředí
EDM	Early Design Model
EF	Hodnota faktoru prostředí
ENG	Externí dotazy
Fc	Korelační faktor
FILE	Vnitřní logické soubory
FILEE	Soubory vnějšího rozhraní
FP	Funkční body
FPA	Function Point Analysis
GUI	Graphical user interface
IF	Idea Formulation
IN	Externí vstupy
IT	Information technology
L	Velká
LOC	Počet zdrojových řádků
M	Střední
MD	Člověkohodiny

---

MRE	Magnitude of Relative Error
NFP	Neupravené funkční body
OUT	Externí výstupy
PAM	Post Architecture Model
PD	Project Definition
PERT	Program Evaluation and Review Technique
PF	Factor productivity
PR	Project Request
S	Malá
SD	Solution Design
SW	Softwarový
T	Doba vývoje
TCF	Faktor technické složitosti
TCP/IP	Transmission Control Protocol/ Internet Protocol
TF	Hodnota technického faktoru
UAW	Unadjusted Actor Weight
UCP	Use Case Point
UUCP	Unix-to-Unix Copy
UUCW	Unadjusted Use Case Weight
V	Velká
XL	Velmi velká

**SEZNAM OBRÁZKŮ**

Obr. 1: Kužel nejistoty [11] .....	15
Obr. 2 Cena za nadhodnocování & podhodnocování [6].....	19
Obr. 3: Rozložení pracnosti mezi etapy [11] .....	52
Obr. 4: Životní cyklus projektu [14] .....	83

## SEZNAM TABULEK

Tabulka 1: Základní druhy komunikačních kanálů .....	20
Tabulka 2: Použitelnost jednotlivých metod [6] .....	22
Tabulka 3: Ukázka jednoduchého odhadu vytvářeného vývojářem [6] .....	23
Tabulka 4: Ukázka odhadu s použitím nejlepšího a nejhoršího případu [6] .....	24
Tabulka 5: Výpočet odhadu použitím nejlepšího, nejhoršího a nejpravděpodobnějšího případu [6] .....	25
Tabulka 6: Ukázka – Tabulka pro sledování přesnosti odhadu [6] .....	27
Tabulka 7: Použitelnost metod v této kapitole [6] .....	28
Tabulka 8: Použitelnost metod v této kapitole [6] .....	30
Tabulka 9: Odhad velikosti programu pomocí Fuzzy logiky [6] .....	31
Tabulka 10: Výpočet průměrného počtu LOC pomocí Fuzzy logiky [6] .....	31
Tabulka 11: Měřítko větším celků [6] .....	33
Tabulka 12: Použití konfekční velikosti pro hodnocení vlastností podle obchodní hodnoty a ceny vývoje [6] .....	34
Tabulka 13: Kombinované obchodní hodnoty založené na poměru ceny vývoje [6] .....	35
Tabulka 14: Řazení odhadů podle přibližné kombinované hodnoty. [6] .....	35
Tabulka 15: Použitelnost metod v této kapitole [6] .....	36
Tabulka 16: Hodnoty parametrů pro základní model. [2] .....	40
Tabulka 17: Hodnoty parametrů pro střední a detailnější model. [2] .....	40
Tabulka 18: Patnáct základních atributů modelu COCOMO. [9] .....	41
Tabulka 19: Měřítko zobecnění [9] .....	42
Tabulka 20: Ukázka - Výpočet neupravených funkčních bodů. [2] .....	46
Tabulka 21: Seznam 14-ti uvažovaných faktorů. [9] .....	47
Tabulka 22: Základní kategorie složitosti – případy užití. [7] .....	49
Tabulka 23: Základní kategorie složitosti – aktéři. [7] .....	49
Tabulka 24: Srovnání vybraných metod odhadu velikosti a ceny softwaru [6],[11],[12],[13] .....	54
Tabulka 25: Počet zaměstnanců [vlastní zdroj] .....	62
Tabulka 26: Oblasti vývoje jednotlivých firem [vlastní zdroj] .....	63
Tabulka 27: Metody odhadování [vlastní zdroj] .....	64
Tabulka 28: Počet firem, které používají softwarové nástroje pro stanovení odhadu [vlastní zdroj] .....	65

Tabulka 29: Parametry podle kterých firmy v ČR kalkulují [vlastní zdroj] .....	66
Tabulka 30: Co všechno firmy zahrnují do odhadu vyvíjeného projektu [vlastní zdroj] .....	66
Tabulka 31: Jednotlivé vstupy používané pro odhadování nákladů [vlastní zdroj] .....	67
Tabulka 32: Počet zaměstnanců podílejících se na vytváření odhadů v jednotlivých firmách [vlastní zdroj] .....	68
Tabulka 33: Počet firem, které jsou omezovány vedlejšími zdroji při stanovení odhadů. [vlastní zdroj] .....	69
Tabulka 34: Počet firem provádějící vstupní sezení [vlastní zdroj] .....	70
Tabulka 35: Počet nedodaných projektů včas [vlastní zdroj] .....	71
Tabulka 36: Počet podhodnocených projektů [vlastní zdroj] .....	72
Tabulka 37: Způsob odhadování malých a velkých sw firem [vlastní zdroj] .....	76
Tabulka 38: Metodika fáze Idea Formulation [14] .....	85
Tabulka 39: Metodika fáze Project Request [14] .....	85
Tabulka 40: Metodika fáze Project Definition [14] .....	86
Tabulka 41: Metodika fáze Solution Design [14] .....	88

**SEZNAM GRAFŮ**

Graf 1: Průměrný počet zaměstnanců v softwarových firmách [vlastní zdroj] .....	62
Graf 2: Oblasti vývoje v jednotlivých firmách [vlastní zdroj].....	63
Graf 3: Metody odhadování softwarových projektů [vlastní zdroj] .....	64
Graf 4: Jednotlivé vstupy používané pro odhadování nákladů [vlastní zdroj] .....	67
Graf 5: průměrný počet odhadovatelů ve firmách [vlastní zdroj].....	68
Graf 6: počet firem, které jsou ovlivňovány vedlejšími vlivy [vlastní zdroj].....	69
Graf 7: Průměrný počet firem provádějící úvodní sezení [vlastní zdroj] .....	70
Graf 8: počet nedodaných projektů včas [vlastní zdroj] .....	71
Graf 9: Počet podhodnocených projektů [vlastní zdroj] .....	72
Graf 10: metody odhadování malých a velkých sw firem v ČR [vlastní zdroj] .....	76

## SEZNAM PŘÍLOH

Příloha PI     Dotazník

Příloha PII    CD – Disk

## PŘÍLOHA P I: DOTAZNÍK

### METODY URČOVÁNÍ CENY SOFTWAREVÉHO PROJEKTU

Cílem dotazníku je zjistit, jakým způsobem české softwarové firmy kalkulují cenu za softwarové projekty.

#### 1) Kolik zaměstnanců má Vaše firma?

- ☐ a) pracuji sám (freelancer)
- ☐ b) 2 až 5 pracovníků
- ☐ c) 6 až 10 pracovníků
- ☐ d) 11 až 25 pracovníků
- ☐ e) 26 až 50 pracovníků
- ☐ f) 51 až 250 pracovníků
- ☐ g) více jak 250 pracovníků

#### 2) Jakým projektům se Vaše firma věnuje?

- ☐ a) www stránky (návrh, realizace)
- ☐ b) webové aplikace
- ☐ c) mobilní aplikace
- ☐ d) desktopové aplikace
- ☐ e) jiné (*uved'te jaké*)

---

#### 3) Jakým způsobem Vaše firma určuje dobu (pracnost) realizace projektu?

- ☐ a) odhadem na základě zkušeností
  - ☐ b) porovnáním z již realizovanými projekty
  - ☐ c) metodou Use Case Point
  - ☐ d) metodou Funkčních bodů
  - ☐ e) pomocí počtu řádků zdrojového kódu
  - ☐ f) jinak (*uved'te jak*)
-

**4) Používá Vaše firma softwarové nástroje pro kalkulaci?**☐ a) ANO (*uved'te jaké*)☐ b) NE**5) Popište, co při kalkulaci ceny za projekt posuzujete a co upřednostňujete?**

.....

.....

.....

.....

**6) Podle jakých parametrů dělá Vaše firma odhad projektu?**☐ a) dle funkčních bloků☐ b) dle člověkohodin**7) Co všechno zahrnuje Vaše firma při kalkulaci do projektu?**☐ a) pouze cenu☐ b) pouze dobu realizace☐ c) cenu i dobu realizace**8) Jaké vstupy jsou používány pro odhadování nákladů?**☐ a) Neúplný soubor požadavků☐ b) Detailní soubor požadavků☐ c) Detailní uživatelské rozhraní, zahrnující prototypy jednotlivých obrazovek☐ d) Kompletní architektura systému☐ e) Jiná odpověď

.....

**9) Kolik osob je zapojeno do zpracování odhadu nákladů?**

- ☐ a) 1 až 2
- ☐ b) 3 až 5
- ☐ c) 6 až 10
- ☐ d) 10 a více

**10) Bývají osoby provádějící odhady nákladů omezeny dostupnými zdroji, např. rozpočet, omezená pracovní síla, čas atd.?**

- ☐ a) Ano
- ☐ b) Spíše ano
- ☐ c) Spíše ne
- ☐ d) Ne

**11) Realizujete na počátku každého projektu úvodní sezení týkající se odhadu?**

- ☐ a) Ano
- ☐ b) Spíše ano
- ☐ c) Spíše ne
- ☐ d) Ne

**12) Kolik procent projektů nebývá dodáno včas?**

- ☐ a) 0 až 25 %
- ☐ b) 26 až 45 %
- ☐ c) 46 až 75 %
- ☐ d) 75 až 100 %

**13) Kolik procent projektů bývá podhodnoceno?**

- ☐ a) 0 až 25 %
- ☐ b) 26 až 45 %
- ☐ c) 46 až 75 %
- ☐ d) 75 až 100 %

**14) Jaký je nejčastější důvod podhodnocení projektů?**

---

---

---

**15) Je Vaše firma ochotna se dále na projektu spolupodílet a poskytnou mi doplňující informace o Vašich projektech (např. poskytnout „Use Case Model“, cenovou kalkulaci projektu, skutečnou dobu realizace, ...)?**☐ a) ANO

*(V případě kladné odpovědi Vás budu kontaktovat a domluvíme se na další spolupráci)*

☐ b) NE**Kontaktní údaje (pouze při kladné odpovědi)**

jméno a příjmení:

email:

telefon:

mobil:

---

---

---

---

## **PŘÍLOHA P II: CD – DISK**

### **Obsah disku:**

- Konečná verze bakalářské práce
- Dotazník k DP
- Vyhodnocení pomocí programu Statistika.cz
- Grafy znázorňující výsledky dotazníkového šetření vytvořené v programu Excel