PROFINIT
new frontier group

# Doporučení pro tvorbu odhadů

**Historie dokumentu**

| Verze | Datum | Autor | Poznámka |
|---|---|---|---|
| 1.0 | 23.02.2012 | Profinit, s.r.o. | Finální verze dokumentu |



## Obsah dokumentu



# 1. Úvod

Smyslem tohoto dokumentu je stručný přehled metodiky tvorby odhadů v rámci společnosti Profinit, s.r.o. Tvorba odhadů pracnosti je svým způsobem umění a proto nelze přesně exaktně určit, který postup je zaručeně nejlepší. Jako podklad pro tvorbu tohoto dokumentu posloužily následující zdroje:

- [1] McConnell, S.: Software Estimation: Demystifying The Black Art. Microsoft Press, 2006,
- [2] Brooks F. P.: The Mythical Man-Month. Addison-Wesley, 1995.

Obě knihy lze zakoupit v každém dobrém knihkupectví s odbornou literaturou a hlavně u první ze zde uvedených se čas strávený přečtením zaručeně vrátí.

# 2. Smysl dokumentu

Většina požadavků na tvorbu odhadů má za cíl pouze jedno jediné – kolik ta daná věc bude ve finále stát, přičemž odpověď by byla vhodná ihned. Nevýhodou je, že v dané chvíli máme k dispozici většinou nejméně potřebných informací, a pokud zrovna neumíme věštit budoucnost, existuje velká šance, že se náš odhad zatíží nemalou chybou. Abychom mohli odhady zpřesňovat, potřebujeme je také porovnávat. Ovšem toto je právě hlavní problém – odhady nejsou často dobře porovnatelné. Každý odhad se člení na jiné kategorie, různě reprezentuje rizika, někdo uvažuje pracnost oprav v záruce, někdo ne, apod. Pokud pak chceme srovnat aktuální odhady vzhledem k historickým údajům (což je základ pro zpřesňování nových odhadů na základě předchozích dat), nevíme přesně, co vlastně srovnáváme.

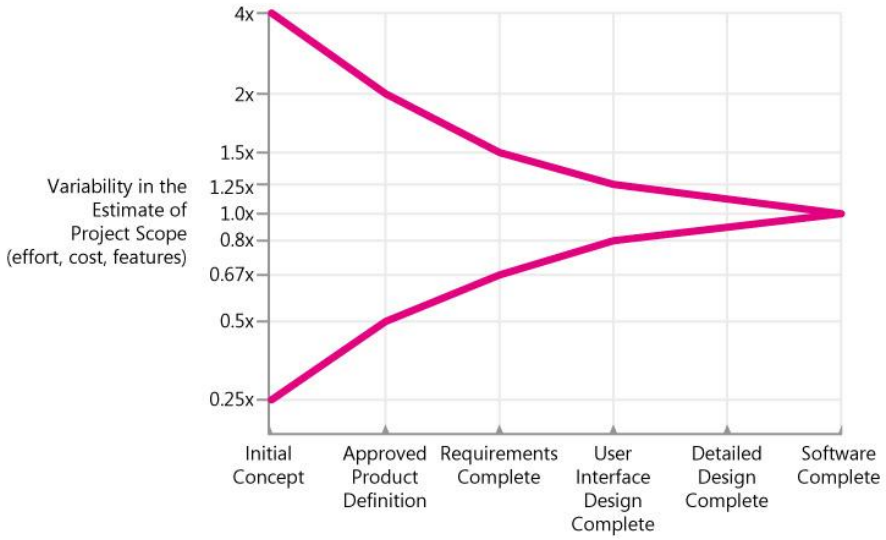
V ideálním případě bychom díky dodržování několika rad z tohoto dokumentu měli docílit konzistentní tvorby odhadů, které by měly vždy pevně danou strukturu a vždy by bylo jasné, co je obsahem daných kategorií. Takové odhady můžeme snadněji revidovat, porovnávat, i zpětně analyzovat vzhledem k reálně získaným hodnotám.

# 3. Tvorba odhadů

## 3.1 Základní doporučení

Tato kapitola uvádí pár základních rad pro tvorbu odhadů:

- Vývojáři jsou od přírody optimisté, a proto většinou bývají optimistické i odhady. Po dokončení odhadu jej raději na chvíli odložíme a pak pro jistotu ještě jednou osobně zrevidujeme (na jednotlivé položky se pak velmi často díváme kritičtěji). Značně nám pomůže také zpětná vazba vzhledem k již realizovaným odhadům (tedy jak byl odhad nepřesný…).
- Je dobré, když odhady tvoří lidé, kteří nakonec realizují vlastní implementaci. Nicméně musíme uvažovat i možnost, že implementace připadne někomu jinému. Neboli při práci na odhadu pracnosti musíme rozmýšlet o lidech, kteří budou na dané věci pracovat.
- Musíme důsledně rozlišovat **odhad pracnosti** a finální **závaznou pracnost**. Pokud se jedná o odhad, je naprosto v pořádku, pokud uvedeme rozsah (v dané chvíli máme k dispozici nejmenší množství informací). Pokud uvádíme finální závaznou pracnost, musíme uvést konkrétní číslo. V tomto dokumentu uvažujeme tvorbu výhradně technických odhadů – nezatížených obchodními a politickými hledisky.
- Dle [1] se při odhadech pracnosti setkáváme s problematikou tzv. kužele nejistoty (ověřeno praxí). Ve zkratce se jedná o to, že čím více se projekt blíží ke konci, tím více můžeme odhad zpřesňovat – tento stav však pro nás (jako tvůrce odhadu) není ideální a jde přesně proti našim potřebám. Dle dosavadních statistik se odhad v úvodní analýze můžeme lišit až čtyřnásobně oproti skutečnosti (a to v obou směrech), přičemž nadcenění je vždy menším zlem (tedy alespoň z pohledu dodavatele) než podcenění. V případě odhadů je tedy naprosto v pořádku, pokud uvedeme větší rozsah (pokud jej samozřejmě lze realisticky zdůvodnit).




**Obrázek 1: Kužel nejistoty (převzato z [1])**

- Odhady musíme sestavovat konzistentně, abychom mohli zpětně revidovat původní předpoklady.
- Pokud něco v době odhadu nemůžeme zjistit a nebo pro účely odhadu předpokládáme, je nezbytné toto do daného odhadu uvést. Pouze tak v případě potřeby určíme, kde jsme zanesli chybu.
- Při odhadech bychom měli primárně počítat (řádky, počty scénářů, obrazovky, …), pokud nelze počítat, následuje využití historických dat. Expertní odhad bychom měli použít až jako poslední možnost.
- Při odhadech bychom měli používat minimální a maximální hranici. Toto představuje podíl rizika v dané kategorii. Minimální hranice je std. nejvíce optimistická (vše jde naprosto ideálně, nebudou žádné chyby), vrchní hranice by měla odpovídat nejhoršímu předpokládanému případu (horní hranice může růst nade všechny meze, ale toto zákazník pravděpodobně neakceptuje…).
- Pokud použijeme minimální a maximální hranici odhadu, můžeme ji ještě efektivně doplnit dle expertního názoru nejvhodnější hodnotou v daném rozmezí. Při takovém přístupu lze aplikovat lehce sofistikovanější metodiky pro určení finálních odhadů (viz [1], kapitola 10.4).
- Při odhadech bychom měli využívat checklisty založené na zkušenostech z předchozích úloh. Dané seznamy mohou být lehce delší, ale pokud důsledně projdeme všechny body, značně snižujeme riziko, že něco opomeneme (tzn. čas strávený s checklistem se nám může velmi vyplatit).
- Při odhadech bychom měli volit vhodné jednotky a zaokrouhlení - nejčastěji používáme MH a MD (člověkohodina a člověkoden). Pokud činnost odhadujeme na 1 hodinu a uvedeme 0,5 MD, tak odhad značně zkreslíme. Na druhou stranu uvedení 523,5 hodiny, je určitě nesmysl - zde bychom měli zvolit vhodnější jednotku 65 MD a nebo rovnou 3 MM (uvažujeme 20 pracovních dní na jeden měsíc).
- Měli bychom se řídit pravidlem, že vývojáři mají tendenci být za každou cenu rychlejší než odhad. Mimo větší pravděpodobnosti vzniku chyby bychom také měli zjistit, zda byli vývojáři opravdu rychlejší (odhad nebyl přesný) a nebo byli rychlejší pouze „papírově“ (reálně stráveno 8 hodin, nahlášeny 4, apod.).
- Musíme pamatovat na pravidlo, že vývojáři se snaží vždy dodržet odhad. Toto může být v rámci projektu žádoucí, ale pro potřeby budoucích odhadů musíme důsledně evidovat navíc strávený čas.
- Odhady musíme vždy revidovat – detailnější popis viz kapitola 4.
- Znalost [1] a [2] nám neuškodí ☺.

# 3.2 Členění odhadu

Tato kapitola uvádí doporučené kategorie členění odhadu při jeho tvorbě. V rámci níže uvedených kategorií bychom měli určitě uvažovat i případné opravy v rámci akceptačního testování na straně zákazníka a tedy lehce předjímat budoucnost – projekt nikdy nekončí dodáním. Je jasné, že nikdy nemůžeme předem odhadnout chybovost, ale nabízí se nám poučení z historických dat (řešili jsme podobný problém, apod.). Rozhodně neplatí následující tvrzení: „Minule byla chybovost cca 30%, nyní máme již zkušenosti a lepší nástroje, … - chybovost bude určitě menší…“ Toto rozhodně neplatí a vzhledem k obecně dostupným informacím o projektech (viz zde uvedená literatura) můžeme tento fakt jednoduše ověřit. Nemusíme se omezovat pouze na chybovost, ale daný princip můžeme uvažovat i v případě problémů analýzy, implementace, apod. Historická data nám neříkají, z čeho jsme se (nejspíše) poučili, ale jak velké problémy v sobě daná úloha skrývá. Proto je velmi důležité průběh projektů sledovat, tedy minimálně oddělit vlastní tvorbu a následné opravy (v ideálním případě by měla být úloha důsledně členěna na níže uvedených 8 kategorií).

## 3.2.1 Analýza

V rámci odhadu již provádíme základní analýzu, ale velmi často budeme vytvářet spíše tištěný dokument, který bude procházet několika koly schvalování. Rozhodně bychom neměli opomenout to, že v rámci analýzy budeme hodně komunikovat a požadavky se mohou značně měnit a narůstat. Mimo základního odhadu této komunikace a vyjasňování bychom neměli opomenout ani toto:

- Budeme v rámci analýzy vytvářet PoC?
- Budeme komunikovat s dalšími partnery mimo našeho primárního zákazníka?
- Jací partneři s námi budou v rámci analýzy spolupracovat (tzn. jaké máme s danými partnery zkušenosti z předchozích analýz)?
- Předpokládáme v rámci analýzy nějaké schůzky (je nutné uvažovat čas cest, apod.)?
- Vlastní dokument nám zabere nemalý čas, například pokud musíme vytvářet náhledy obrazovek, ukázky rozhraní, apod.
- Pokud budeme vytvářet rozhraní pro třetí strany, je nutné uvažovat větší rizika analýzy.
- Nesmíme zapomenout na interní revize specifikace (toto také stojí další čas).
- Musíme uvažovat i případný čas pro zapracování změn do specifikace, které vyplynou v dalších částech projektu (v průběhu implementace, v opravách, …).
- Kolika systémů se analýza týká, jakou znalost o těchto systémech máme?
- Budeme v rámci analýzy mimo specifikace vytvářet i další dokumenty?
- Budeme muset v rámci analýzy řešit výkonové požadavky systému/aplikace?
- Budeme muset v rámci analýzy řešit bezpečnostní otázky systému/aplikace?

## 3.2.2 Design

Tato disciplína velmi často splývá s implementací (u menších změnových řízení nebo oprav), ale rozhodně bychom tuto kategorii neměli bezmyšlenkovitě přeskakovat. Neměli bychom opomenout minimálně toto (což může sloužit i pro zamyšlení se nad nutností designu):

- Budou měněna rozhraní pro třetí strany? Pokud ano, známe definované parametry a kritéria pro tato rozhraní?
- Jak zapadá koncepce řešení do architektury stávajícího systému?

- Budeme realizovat konverze stávajících dat? Pokud ano, co všechno bude zahrnuto a můžeme vhodným designem tuto konverzi ovlivnit (eliminovat, zjednodušit)?
- Plánujeme refaktoring stávajícího systému pro lepší integraci nových požadavků?
- Počítáme s komunikací se zákazníkem v rámci fáze designu (otázka rozhraní, konverze, …)?
- Nesmíme opomenout ani úpravu designové dokumentace (pro zákazníka, třetí strany, ale i interní – například datové modely, popis designu systému, apod.).

### 3.2.3 Implementace

Výsledný kód je vlastně to jediné, co zákazník opravdu potřebuje. V rámci odhadu bychom neměli opomenout následující otázky:

- Znalost technologie a dané problémové domény.
- Vztah řešení ke stávajícím datům v systému – konverze, datafixy, … (viz design).
- Nastavení vývojového prostředí.
- Tvorba jednotkových testů, úprava stávajících.
- Tvorba testovacích dat.
- Tvorba mock rozhraní pro základní otestování po implementaci.
- Čas pro otestování vlastní implementace (včetně debugování) – testování po vývoji.
- Programátorská dokumentace.

### 3.2.4 Testování

Testování je velmi důležité, a proto bychom pro něj měli vyčlenit nemalou část projektu. Bohužel většinou vzhledem k časovým možnostem se tato disciplína opomíjí (což se nám ovšem většinou zanedlouho vrátí…). Rozhodně bychom neměli opomenout následující činnosti:

- Tvorba testovacích scénářů pro zákazníka.
- Otestování na vývojovém prostředí, otestování na QA prostředí. Zde musíme počítat i s časem, kdy je nutné seznámit se s problematikou, abychom vůbec mohli testovat – dalším aspektem může být:
  - Musíme pro testování speciálně nastavit prostředí (například instalace Selenium IDE, SoapUI, jiný prohlížeč …)?
  - Musíme spouštět dávky nebo jinak modifikovat vývojové prostředí (například specifické hodnoty v databázi, emulace stavu aplikace, apod…)?
  - Potřebujeme rozhraní třetí strany (s tím mohou souviset certifikáty, přístupové údaje, …).
- Budeme vytvářet nové regresní testy/upravovat stávající?
- Testování finální dodávky před odesláním.
- Kvalifikační testování.
- Podpora akceptačního testování (testování na straně zákazníka).
- Pokud výkonnostní požadavky uvádíme v analýze, musíme se jimi zabývat i v rámci testování.
- Zde uvažujeme primárně testování nově přidané funkcionality, dále se testování může objevit i ve vlastní dodávce (viz 3.2.6) – tento čas zde tedy nezapočítáváme.

### 3.2.5 PM (Project management)

V této kategorii se skrývají všechny činnosti, které souvisí s vedením projektu. Tyto činnosti můžeme vzhledem k jejich charakteru předem velmi těžce odhadnout. Zde se nám vyplatí využití historických dat („ověřená“ konstanta, statistická data). Primárně nás zajímají tyto činnosti:

- Činnosti související s vedením týmu:
  - Rozdělování práce.
  - Tvorba plánů a jejich aktualizace.
  - Interní schůzky.
- Dojednávání termínů dodání a rozsahu (chyba/změna, apod.).
- Schůzky a diskuze nad rámec výše uvedených kategorií (problémy vzniklé při dodatečné analýze, implementaci, …).

Stejně jako v případě analýzy nesmíme opomenout ani počet účastnících se stran – každý další partner, se kterým musíme komunikovat, nám zásadně navyšuje pracnost v této kategorii.

### 3.2.6 Tvorba dodávky

Tvorba dodávky je nedílnou součástí každého projektu, proto musíme zohledňovat tuto činnost i v rámci odhadů. Neměli bychom opomínat tyto aspekty:

- Musíme uvažovat vlastní pracnost tvorby dodávky (částí dodávky), ale i další formální činnosti (sepsání součástí dodávky, Issue tracking, nahrání dodávky na FTP, …).
- Stejně jako u dříve uvedených částí odhadu musíme uvažovat možnost opravných dodávek (viz použití minimální a maximální hranice).
- Musíme zvážit všechna rizika spojená s dodávkou (podpora nasazování u zákazníka, integraci se systémy zákazníka, apod.).

### 3.2.7 Ostatní

V této kategorii se ukrývají všechny činnosti, které nelze jednoznačně zařadit do výše uvedených. Může se jednat například o toto:

- Vytváření dokumentace (popis rozhraní, konfigurace, začlenění postupu nasazování, …).
- Aktualizace interní dokumentace.
- Konfigurační řízení (například u nových projektů).
- …

### 3.2.8 Kam se ztratila rizika?

Rizika uvádíme pro každou konkrétní kategorii, proto není důvod abychom je uváděli samostatně. Rozdělení na minimální a maximální hranici u jednotlivých kategorií nám pomáhá při vlastní analýze odhadu, navíc můžeme jednoduše (na rozdíl od shrnutí všech rizik do jedné kategorie Rizika) určit, kde předpokládáme největší problémy.

Musíme mít na paměti, že rozsahy ve výše uvedených kategoriích zahrnují toto:

- Opravy vzniklé na základě testování po vývoji a kvalifikačního testování.
- Případné opakování dodávek.

- Opravy vyžádané akceptačním testováním ze strany zákazníka (platí pro všechny výše uvedené kategorie).
- Problematiku výkonu systému/aplikace, pokud zákazník požaduje danou oblast řešit.
- Problematiku zabezpečení systému/aplikace, pokud zákazník požaduje danou oblast řešit.
- …

Výše uvedené nám tedy říká, že v odhadu musíme v rozsahu uvažovat všechny aspekty, které nás mohou potkat od počátku analýzy až do nasazení produktu do produkčního prostředí (tedy do doby akceptování ze strany zákazníka), záruku bychom již v rizicích uvažovat neměli.

### 3.2.9 Záruka

Pod zárukou se skrývá speciální kategorie, kterou bychom měli v rámci odhadu také uvažovat. Jedná se o předpokládaný čas, který si vyžádají opravy v záruce. Tuto hodnotu musíme opět sledovat pomocí historických dat, pro hrubý odhad však můžeme uvažovat 5% ze součtu kategorií 1-7 (jedná se o hodnotu vysledovanou napříč projekty v rámci Profinitu). Uvažovanou výši záruky bychom neměli zahrnovat do rizik – rizika končí nasazením aplikace do produkčního prostředí, záruka v tomto bodě naopak začíná,

# 4. Revize odhadů

Všechny odhady (nezávisle na velikosti) musíme revidovat. Revizi nesmí provádět tvůrce originálního odhadu, přičemž autor revize by neměl být zatížen originálním odhadem. Důvody pro tento požadavek jsou minimálně dva:

- Pokud známe originální výši odhadu, většinou se k ní snažíme podvědomě přiblížit.
- Při znalosti originálního rozpadu většinou použijeme obdobnou dekompozici.

Situace, kdy se revize liší od originálu je ve většině případů žádoucí, protože upozorňuje na možné problémy projektu. Na druhou stranu neplatí, že revize je vždy kvalitnější a reálnější než originál. Při tvorbě revizí je velmi dobré, pokud používáme odlišné techniky odhadu. Tuto problematiku detailně popisuje [1] v části II. Následující seznam uvádí pouze přehled možných technik:

- Výpočet – lze počítat například toto:
  - business požadavky,
  - funkční požadavky,
  - případy užití,
  - počty změnových řízení,
  - stránky/obrazovky/dialogy,
  - reporty,
  - databázové tabulky/třídy,
  - počet již napsaných řádků kódu,
  - … vše relevantní k danému projektu.
- Odhad na základě historických dat (obdobný již realizovaný projekt, …).
- Expertní odhad.
- Dekompozice.
- Analogie s obdobným projektem/problémem.

- Tzv. proxy odhad:
  - například T-Shirt sizing – projekt velikostí S, M, L, XL – každá kategorie je následně ohodnocena pro stanovení odhadu.
- Software pro stanovení odhadů.

Musíme mít na paměti, že každá z výše uvedených technik nabízí jinou míru přesnosti a je tedy vhodná pro jiné fáze projektu (T-Shirt sizing se na závazný odhad po analýze jistě nehodí).

# 5. Přílohy

## 5.1 Tabulková podoba odhadu

Pro větší projekty můžeme využít šablonu pro rozpady odhadů (viz `Profinit_metodika_odhadu.xlsx`). V této tabulce budeme vyplňovat minimální a maximální odhady pracnosti jednotlivých činností, tabulka však obsahuje i položku pro expertní odhad (nejvíce pravděpodobný případ). Na základě této položky se počítá tzv. očekávaný případ (viz [1] kapitola 9). Tuto tabulku můžeme využívat i pro malé projekty (cca do 10 MD), nicméně stále platí pravidlo, že její použití musí mít reálný význam a být účelné!