SAMOZREJME, ŽE TO STIHNEM...

Čas plynie tým rýchlejšie, čím viac sa blíži termín odovzdania.

Ivan Polko

Slovenská technická univerzita
Fakulta informatiky a informačných technológií
Ilkovičova 3, 842 16 Bratislava
ivan12[zavináč]gmail[.]com

Abstrakt. *Plánovanie času je neoddeliteľnou súčasťou nielen softvérových projektov. Čas si plánujeme aj v každodennom živote, mohli by sme si preto myslieť, že v tom budeme mať prax. Z vlastných skúseností však vieme, že to tak bohužiaľ nie je. Ako ukazujú mnohé psychologické štúdie, optimizmus pri plánovaní vlastného času je úplne bežný. V eseji bližšie popíšem tento jav a budem sa mu venovať vo vzťahu k softvérovým projektom, kde sa termínom a odhadom určite nevyhneme. Jednou z metód, ktorá sa snaží dôsledky tohto javu riešiť, je evidencia vlastných časových odhadov, ako aj reálne spotrebovaného času potrebného na vyriešenie úloh. Takto si môže každý vývojár zistiť nepresnosť svojich odhadov a zlepšiť tak svoje odhady do budúcnosti. Manažéri získajú reálne odhady dokončenia projektu, a ak využijú koncept dvoch plánov a v pláne pre zákazníka zohľadnia problémy, ktoré môžu nastať, výsledkom bude splniteľný plán.*

Kľúčové slová: *plánovanie, plán, odhadovanie, optimizmus*

Úvod

Prečo často nestíhame dokončiť svoju prácu načas, či pravidelne meškáme na dohodnuté stretnutia? Odpoveď sa zdá jednoduchá, zle sme si naplánovali čas. Svoj čas si pritom plánujeme v podstate každý deň, preto by sme sa postupne mali stať v plánovaní času expertmi. Psychologické štúdie však ukazujú, že to tak bohužiaľ nie je. Optimizmus pri časovom plánovaní vlastných úloh pretrváva a pri vytváraní plánov neberieme svoje minulé neúspechy do úvahy.

Jednou z vlastností softvéru, ktorú musíme zohľadniť pri časovom plánovaní softvérového projektu, je jeho neviditeľnosť. Nevieme teda presne povedať, čo všetko nás čaká pri jeho vývoji, s akými chybami sa stretneme, ani koľkokrát zmení zákazník svoje požiadavky. Keďže veľa vecí nevieme, zostáva nám pri časovom plánovaní len odhadovať. Ako však odhadovať čas, keď vieme, že sa mýlime, a že sa nepoučíme?

Zlé odhady spôsobujú veľké problémy nielen v softvérových projektoch. Odhadovať pritom treba na všetkých stupňoch organizačnej hierarchie. Vývojári si odhadujú čas potrebný na splnenie svojich úloh, a manažéri odhadujú čas potrebný na splnenie celého projektu alebo jeho časti. Chyby pri odhadoch vývojárov pre jednotlivé úlohy sú v absolútnom meradle malé, hodiny možno dni. Posuny v pláne, vyplývajúce práve z takýchto zlých odhadov sa však potom kumulujú a nakoniec meška celý projekt. Relatívne to môže byť rovnaké percento ako pri jednotlivých úlohách, v absolútnom vyjadrení sú to však mesiace a v niektorých prípadoch aj roky. Dôsledkom oneskorení pri projekte sú veľké sumy, o ktoré sa celý projekt predraží, ak sa samozrejme vôbec dokončí. Nestíhanie termínov sa tak stáva bežnou situáciou, a nakoniec žiadna zo zúčastnených strán v projekte nie je vôbec prekvapená, keď sa stanovený termín nestihne.

Nevidím v tom žiaden problém

Najprv sa pozriem na jav prílišného optimizmu pri vytváraní vlastných časových plánov (angl. *planning fallacy*). V psychologickej štúdii [1] sa autori snažili tento jav preskúmať. Testované boli tri hypotézy, ktoré boli potvrdené aj experimentálnymi výsledkami:

1. Ľudia sú príliš optimistickí pri vytváraní časových odhadov pre vlastné úlohy. Neplatí to však, keď odhadujú čas, ktorý potrebujú iní ľudia na dokončenie svojich úloh.
2. Pri odhade času sa ľudia spoliehajú skôr na naplánovaný scenár, ako na relevantné skúsenosti z minulosti.
3. Ľudia znižujú význam minulých skúseností pri časovom odhadovaní aktuálnych úloh.

O výsledkoch experimentov sa v štúdii ďalej diskutuje. Samotný optimizmus vychádza z toho, že ľudia si vytvoria v hlave príbeh, ako úlohu úspešne vyriešia. Neuvažujú pritom s predchádzajúcimi skúsenosťami. Štúdia ponúka nasledujúce dôvody pre toto správanie:

- Ľudia si ťažko vytvárajú prepojenia medzi odhadom, ktorý reprezentuje budúcnosť, a minulosťou.
- Ak nastali v minulosti nejaké neúspechy, tak sa ich ľudia snažia vysvetliť pomocou externých faktorov, za ktoré oni nemohli.
- Ľuďom sa nepáči dôsledok zohľadnenia minulých skúseností, teda že projekt by trval dlhšie ako dúfajú.

Zaujímavé je, že tento optimizmus podľa štúdie vymizne, keď odhadujeme čas iným ľuďom. Nie sme prekvapení, keď kolegovi, či známemu trvajú úlohy dvakrát dlhšie ako si pôvodne naplánoval, aj keď o jeho konkrétnom pláne nevieme. Pritom za oneskorenie viníme jeho, nie externé udalosti. Keby sme však boli na jeho mieste, pohľad sa úplne

otočí. Ľudia, ktorí sú nezaujatí, skôr berú do úvahy možné prekážky, a vytvoria tak reálnejší odhad. Je to podľa mňa paradoxné, lebo intuitívne by som povedal, že čím viac viem, tým lepší odhad spravím.

Ak sa však nad tým zamyslím, z vlastných študentských skúseností môžem potvrdiť pravdivosť záverov štúdie. Keď si plánujem, koľko mi bude trvať práca na zadaní, nezohľadňujem veľmi skúsenosti s predchádzajúcimi zadaniami. Nakoniec aj tak pri vypracovávaní zadania strávim viac času, rovnako ako veľakrát predtým. Keď však svoj plán prezradím spolužiakovi, tak mi povie, že to nestihnem. A väčšinou sa nemýli. Študenti majú ešte tú výhodu, že zlý časový plán môžu dohnať na úkor spánku.

Mýliť sa je ľudské

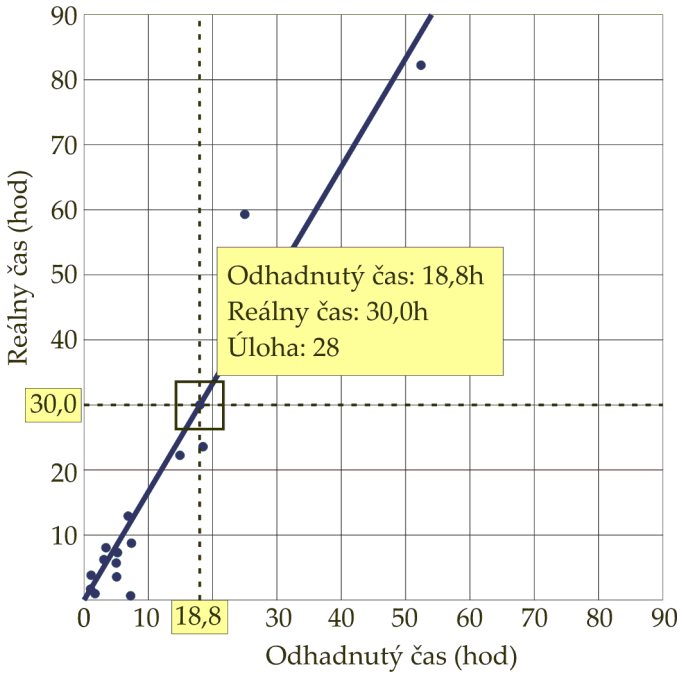
Ako som už spomínal v úvode, v softvérových projektoch spôsobujú nepresné odhady problémy. Základom softvérového projektu sú vývojári, ktorí neradi robia odhady a časové plány, pretože je to náročné, a nikto sa nakoniec podľa nich neriadi. Otázkou potom je, ako vytvoriť metódu na odhadovanie, ktorá bude vyhovovať vývojárom, a taktiež bude zohľadňovať prirodzený sklon ľudí k optimizmu pri odhadovaní času. Joel Spolsky navrhol jednoduchú metódu, založenú na evidencii odhadov z minulosti [2] (angl. *evidence based scheduling* – ďalej budem používať skratku EBS), ktorá sa snaží na túto otázku odpovedať.

Prvým krokom metódy EBS je rozdeliť si úlohy na malé časti. Spolsky uvádza, že keď vidí časové plány, v ktorých jednotlivé časti trvajú dni alebo aj týždne, tak hneď vie, že takýto plán je odsúdený na neúspech. Úlohy by podľa neho mali byť tak malé, aby mohli byť merané v hodinách. Žiadna úloha by pritom nemala trvať viac ako 16 hodín. Rozdelením úloh na malé časti si má vývojár uvedomiť, čo všetko úloha zahŕňa, a zamyslí sa tak aj nad tými krokmi, nad ktorými by sa inak nezamyslel.

Osobne si však neviem predstaviť, že mám nejakú úlohu a odhadnem, že mi jej splnenie bude trvať 12 hodín. Prečo nie 11 alebo 13 hodín? Takýto časový odhad je v poriadku, ak môžem vychádzať z minulých skúseností s podobnými úlohami. Pri úlohách, ktoré by sa však svojou dĺžkou blížili k spomenutému limitu 16 hodín, je podľa mňa nutné aspoň v hlave si spraviť rozdelenie na menšie časti a tie odhadnúť samostatne.

Základom Spolskeho metódy je vytváranie si tabuľky z histórie odhadov. V tabuľke budú zaznamenané samotné časové odhady pre jednotlivé úlohy, ako aj reálny čas potrebný na ich splnenie. Z týchto údajov je potom možné vytvoriť graf znázornený na Obr. 1, v ktorom sú dané do vzťahu odhadované a reálne časy. Takýto graf hovorí o konzistentnosti odhadov daného vývojára a tiež o jeho rýchlosti, čiže pomere odhadnutého času a reálneho času. Spolsky potom podľa rýchlosti rozdeľuje vývojárov na 3 skupiny:

- Dokonalý vývojár, ktorý je samozrejme nereálny, odhadne všetky úlohy presne, jeho rýchlosť je teda vždy 1.
- Zlý vývojár, ktorý odhaduje veľmi nepresne, jeho rýchlosť významne kolíše.
- Väčšina vývojárov má však odhady pomerne konzistentné, v nejakom pomere k reálne potrebnému času.



Obr. 1. Graf odhadov vývojára (podľa [2]).

Ako sa vývojárom postupne zlepšuje schopnosť odhadovať, treba staršie údaje zahodiť. Novým vývojárom navrhuje Spolsky priradiť vymyslenú históriu s rôznymi rýchlosťami, kým neskončia dostatok reálnych úloh na to, aby mali vlastné údaje. Ďalej už v eseji nerozoberá, či sú to skúsení alebo neskúsení vývojári. Skúsení vývojári vedia už lepšie odhadovať, nemali by teda dostať rovnako vymyslenú históriu ako neskúsení vývojári. Podľa mňa by mohli dostať históriu od iného vývojára, ktorý má podobnú rýchlosť. Skúsený vývojár by totiž zo svojich skúseností mal vedieť aspoň približne určiť svoju rýchlosť, aj keď si ju detailne neevidoval. Cieľom je, aby zbytočne nevnášal nepresnosť do odhadov, tak ako neskúsený vývojár, u ktorého je to oprávnené.

Problémom môže byť tiež Parkinsonov zákon [3], že práca sa vždy natiahne do takej miery, aby vyplnila všetok čas, ktorý je k dispozícii. Vývojár si odhadne čas, povedzme na 4 hodiny, a na základe svojej histórie odhadov vidí, že mu to bude trvať asi 8 hodín. Čo ak mu napadne myšlienka, že mám na to 8 hodín, nemusím sa tak ponáhľať? Keď nebude pracovať tempom, pri ktorom odhadoval čas na 4 hodiny, môže sa ľahko stať, že svoje prácu nestihne ani za 8 hodín. Vývojára však podľa mňa motivuje možnosť mať robotu skôr hotovú, a dokázať tak, že odhaduje lepšie. Ak sa teda bude snažiť mať úlohu hotovú za 4 hodiny, tak to síce aj tak nestihne, ale mu to bude trvať práve tých 8 hodín.

Myslím si, že dôležitým aspektom metódy EBS je aj samotný graf, ktorý vie vizuálne lepšie presvedčiť človeka o tom, čo je reálne schopný zvládnuť. Ťažko bude niekto optimisticky trvať na tom, že stihne úlohy v kratšom čase, keď jasne vidí, že výsledný záznam v grafe by bol úplne mimo trendu. Ako sa hovorí, obrázok je viac ako tisíc slov. Pohľadom do svojej histórie je tak vývojár prinútený zohľadniť minulé skúsenosti, ktorých výsledkom je práve tento graf.

Metóda EBS teda kompenzuje prirodzene optimistické odhady vývojárov na základe ich optimizmu v minulosti. Na tejto metóde sa mi páči, že vývojári sa nemusia učiť žiaden zložitý postup. Postačuje, keď budú konzistentní v nepresnosti svojich odhadov. Nemusia sa snažiť korigovať svoj optimizmus, aj keď bude len dobre, ak sa budú vo svojich odhadoch na základe svojej histórie zlepšovať. Samozrejme, nevyhnutnosťou je dobrý softvérový nástroj, aby s touto evidenciou strávili naozaj minimum času.

A čo na to manažéri?

Keď už máme k dispozícii historické údaje a časové odhady pre úlohy aktuálneho projektu, pomocou štatistických metód vieme zistiť dátum, kedy bude projekt hotový. Dátum dokončenia bude však určený len s istou pravdepodobnosťou, ktorá vychádza z nepresnosti odhadov jednotlivých vývojárov pracujúcich na projekte. Použité štatistické metódy sú opísané v [2], nebudem sa im preto ďalej venovať. Manažér, ktorý vytvára celkový odhad ukončenia projektu, má tak možnosť získať pomocou metódy EBS kvalitný odhad. A ani jeho optimizmus nezmení nič na dátume dokončenia a jeho pravdepodobnosti.

Do odhadov je však ťažké zahrnúť vyrušenia, nepredpokladané chyby, problémy s hardvérom a podobne. Metóda EBS však ponúka jednoduché riešenie, tieto záležitosti neodhadovať a zarátať ich do času stráveného nad úlohami. Či sú tieto zdržania pravidelné alebo nepravidelné, výsledný odhad ich zohľadní, a bude podľa toho presný alebo nepresný. Najhoršie sú nepravidelné vyrušenia, ktoré znižujú presnosť výsledných odhadov. Dobrý manažér podľa mňa môže zistiť, že inak dobrý vývojár začal mať nepresné odhady, a preto by mal nájsť príčinu a tento problém vyriešiť. Samozrejme, toto nie je vždy možné, hlavne ak vývojára vyrušuje samotný manažér.

Nie vždy nastane ideálny stav, že v projekte sa implementujú len tie funkcie, čo sa naplánovali na začiatku. Ako riešiť pridávanie nových funkcií? Spolsky navrhuje vytvoriť si časovú rezervu v projekte práve na pridávanie funkcií, a aj na riešenie možných problémov, ktoré môžu nastať pri integrácii, ladení, testovaní použiteľnosti, či beta testovaní.

Na koniec svojej eseje ešte Spolsky uvádza pár rád určených skôr pre manažérov:

- Iba programátor, ktorý naozaj danú úlohu vykonáva, môže vytvárať odhad. Keď to robí manažment, projekt je odsúdený na neúspech.
- Čas potrebný na odstraňovanie chýb sa zarátava k samotnej úlohe. Metóda EBS tak umožňuje predpovedať, koľko bude trvať získanie plne odladeného kódu.
- Nenechať manažérov tlačiť vývojárov do kratších odhadov.
- Ak máme časový plán s reálnymi odhadmi a zistíme, že nestíhame, potom radšej neimplementujme niektoré menej dôležité funkcie. Môžeme sa tak zamerať na funkcie, ktoré sú podstatné a užitočné.

K odhadovaniu času manažmentom by som poznamenal toľko, že manažér je zainteresovaný do projektu, vidí plán, ktorý hovorí ako všetko hladko pôjde, teda je optimista. Netuší však, čo všetko je potrebné spraviť pri jednotlivých úlohách, a čo nečakané môže nastať. Preto by do týchto odhadov naozaj nemal zasahovať.

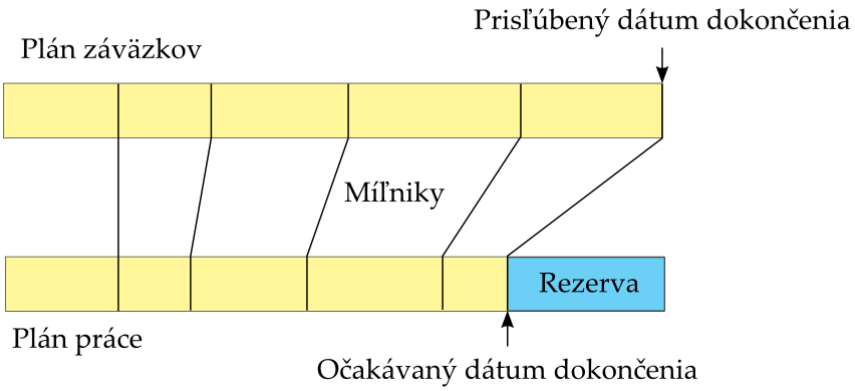
Chce však manažment jednoduchú metódu? Jednoduchosť nie je niekedy želaná. Manažéri vyzerajú dôležitejšie, keď povedia, že na odhadovanie bola použitá niektorá z parametrických metód, kde je veľa tabuliek, koeficientov a vzťahov. Tieto metódy na výstupe poskytnú odhad úsilia, ktoré je potrebné na dokončenie projektu. Sú tam použité rôzne empiricky zistené koeficienty, a stráca sa tam podľa mňa nejaké prepojenie na konkrétnych vývojárov, ktorí budú samotný softvér vytvárať.

Myslím si, že takéto metódy majú svoje miesto, napríklad pri vytváraní ponuky pre väčší projekt. Vtedy ešte nemáme známe rozdelenie na jednotlivé úlohy, a teda sa nedá použiť metóda EBS. Keď sú však známe detailnejšie úlohy a pridelenie ľudí, podľa môjho názoru dokáže táto metóda veľmi vhodne odhadnúť, koľko bude vývoj trvať. Problém nastane, ak sa zistí, že vývoj bude trvať dlhšie, ako sme pôvodne odhadli. Tu sa výborne hodí Spolskeho rada, že nemá zmysel tlačiť vývojárov do kratších termínov, treba radšej vyhodiť niektoré naplánované funkcie, alebo ich aspoň zjednodušiť.

Čo povieme zákazníkovi?

Teraz sme v situácii, že máme v projekte naplánované úlohy metódou EBS, ktoré sa budú vykonávať, a nejakú časovú rezervu na riešenie možných problémov. Samozrejme môžeme dúfať, že sa nič neočakávané nestane, ale podľa mňa je to rovnaké, ako dúfať vo výhru v lotérii. Phillip G. Armour vo svojej eseji [4] tvrdí, že ak sa s rizikom nepočíta, niekto musí znášať následky. Na jednej strane sú to zákazníci, ktorí dostanú produkt nižšej kvality, či menšieho rozsahu. Na druhej strane je to tím, ktorý musí pracovať nadčasy, čo sa samozrejme negatívne prejaví na medziľudských vzťahoch.

Armour teda ako racionálne riešenie navrhuje vytvoriť dva plány. Prvý je takzvaný pracovný plán. Zohľadňuje všetky zdroje, o ktorých si v čase vytvárania plánu myslíme, že ich bude projekt potrebovať. Toto je tradičný plán práce, podľa ktorého sa riadi tím. Druhým plánom je takzvaný plán záväzkov, ktorý povieme zákazníkovi. Zahŕňa pracovný plán a pridáva k nemu dostatočnú časovú rezervu. Koncept dvoch plánov je znázornený na Obr. 2.



Obr. 2. Koncept dvoch plánov (podľa [4]).

V eseji sa ďalej rozoberá otázka, či prezradiť projektovému tímu aj plán záväzkov, alebo sa pred nimi tváriť, že pracovný plán je reálnym plánom. Armour odpovedá, že keby sme to pred nimi tajili, mohli by mať pocit neprávosti alebo toho, že ich manažment tlačí do kratších časov, keď pritom je času dosť. Tým, že firma vyhradí zdroje na dlhší čas, aj keď nevie, či ich naozaj bude treba, znáša časť rizika. Robí to však aj preto, aby tím nemusel pracovať nadčasy.

S tým, že pred vývojármi netreba mať tajnosti plne súhlasím. V dobrom tíme to nemá miesto. Ak je na plánovanie pracovného plánu použitá metóda EBS, tak vývojári by mali tiež vedieť, že pracovný plán je naozaj reálny, pretože bol vytvorený na základe ich odhadov. Ale samozrejme, odhadovať mohli len tie úlohy, ktoré boli pri odhadovaní známe.

Keď máme reálny plán záväzkov, ktorý sme schopní dodržať, nie je ešte isté, či zákazník naozaj chce počuť splniteľný termín. Armour v inej eseji [5] uvádza, že len 20% softvérových projektov skončí úspešne načas, a z určovania termínov sa vlastne stala taká hra. Manažéri zvyknú dávať nereálne termíny a nakoniec ani zákazníci už neveria, že sľúbené termíny ukončenia sú reálne. Zo svojich skúseností potom Armour uvádza príklad niekoľkých zákazníkov, ktorí si od začiatku vytvorili svoj plán, ktorý počítal s neskorým dodaním produktu. Vidím v tom peknú analógiu so systémom dvoch plánov. Rozdiel je len v tom, že druhý plán si vytvoril zákazník a nie dodávateľ.

Je preto pochopiteľné, že sa zákazníci budú stavať skepticky k našim tvrdeniam, že daný termín je naozaj reálny. Môžeme napríklad súťažiť s konkurenciou, ktorá bude ponúkať nereálne termíny. Keď však vieme, že máme splniteľný termín, môžeme ponúknuť napríklad väčšie zľavy pre zákazníka v prípade, že termín nestihneme. Tým signalizujeme zákazníkovi, že si naozaj veríme. Konkurenčná firma s nereálnym termínom také podmienky nemôže ponúknuť, ak aspoň trochu racionálne uvažuje.

Záver

Samotné odhadovanie je vlastne hazard. Pri odhadovaní totiž dávame do stávky peniaze, o ktoré môžeme prísť zlým odhadom. A veľakrát o ne naozaj prichádzame. Ak by človek, zodpovedný za vytvorenie odhadu videl pred sebou všetky tieto peniaze, možno by si uvedomil, že naozaj ide o veľa. Nadnesene sa potom dá povedať, že pri odhadovaní sa správame ako gambleri. Aj keď sa nám nedarí a stále prichádzame kvôli tomu o peniaze, nevieme prestať. Ako som však uviedol, je to prirodzená ľudská vlastnosť. Jediný spôsob ako s týmto nedostatkom môžeme bojovať, je využiť ako svoju výhodu to, že sme si ho vedomí.

Metóda EBS ktorú som v eseji opísal, priamo počíta s optimizmom v odhadoch, a vhodne tak koriguje odhady do budúcnosti. Nezaťažuje veľmi vývojára a poskytuje mu priestor na zlepšovanie svojich schopností, čo sa týka odhadovania. Pre manažéra poskytuje reálny odhad dátumu ukončenia, a keď využije aj opísaný koncept dvoch plánov, zákazník dostane splniteľný plán, v ktorom sú zahrnuté aj možné problémy, ktoré sa pri každom projekte určite objavia. A splniteľný plán je predsa to, čo všetci chceme.

Použitá literatúra

1. Buehler, R., Griffin, D., Ross, M.: Exploring the „Planning Fallacy“: Why People Underestimate Their Task Completion Times. *Journal of Personality and Social Psychology*, Vol. 67, No. 3 (1994) 366-381.
2. Spolsky, J.: *Evidence Based Scheduling*. 2007. Dostupné na internete: http://www.joelonsoftware.com/items/2007/10/26.html, [cit. 2010-10-16]
3. Parkinson, C.N.: Parkinson's Law. Dostupné na internete: http://www.berglas.org/Articles/parkinsons_law.pdf, [cit: 2010-10-20]
4. Armour, G.P.: To plan, two plans. In *Communications of the ACM*, Vol. 48, No. 9, 2005, 15-19.
5. Armour, G.P.: Twenty Percent. In *Communications of the ACM*, Vol. 50, No. 6, 2007, 21-23.

Annotation

I will make it in time, of course…

Planning of time is essential not only for software projects. We plan our time every day therefore we should be experts in time planning. But we know, it is not so. A lot of psychological studies indicate that optimism in prediction of task completion times is common. I will describe this phenomenon and I will discuss it in relation with software projects, where deadlines and estimates are necessary. One method, which tries to solve consequences of this phenomenon, is evidence based scheduling. In this method estimated and actual hours needed for task completion are recorded. Every developer can find out his inaccuracy and improve his estimates in the future. Managers will get real estimates of project completion date from this method. If they also use concept of two plans customer will get achievable plan, which takes into account possible problems.