ABY SA TO NEPOKAZILO

Ideálnym plánovaním o krok bližšie k ceste do pekiel.

Marek Mego

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

Abstrakt. *Máloktorý projekt, a to nie len v IT sa skončí presne tak, ako sa skončiť mal. V časovom limite, bez prekročenia dohodnutého finančného zabezpečenia a pri splnení požadovanej kvality. Naopak veľa projektov skončí so zvýšenými finančnými alebo časovými nárokmi. A aj to je ten lepší prípad. Prečo je to tak? Za všetkým sú "neočakávané" udalosti a ich dôsledky. Vo väčšine prípadov sa však tieto neočakávané udalosti očakávať dajú, ba dokonca by sa očakávať mali. Úlohou mojej eseje je identifikovať riziká v softvérových projektoch a to najmä tie, ktoré sa často bagatelizujú a predsa sú významné vo vývoji. V práci sa taktiež venujem príčinám nevhodného plánovania a ďalších zaujímavých rizík vývoja softvérového projektu. Taktiež je cieľom odpovedať na otázku, či predchádzanie možným rizikám je efektívne, respektíve možné a v ktorých situáciách.*

Kľúčové slová: *riziko, softvérový projekt, plán, zlyhanie, chyby, ľudský faktor, manažér, vývojár*

Chyby, chyby, chyby...

Každý sme sa raz ocitli v situácii, keď niečo nevyšlo podľa predstáv. Veľa ľudí takýto výsledok hodnotí ako chybu či katastrofu. Niektorých zlyhanie ovplyvní natoľko, že pod tlakom chybujú znovu a od katastrofy už nie je ďaleko. Ako takúto situáciu riešiť? Istotne, je tu možnosť, nechybovať vôbec. Dosiahnuť takýto stav by asi značne znížilo efektivitu našej práce. Ako teda? Teraz asi nepoviem nič nové, ale robiť chyby je ľudské. Myslím si, že práve chyby sú základom úspešných projektov. Aj Krištof Kolumbus chyboval pri ceste do Indie a napriek tomu bol z jeho plavby úspešný projekt. Objavil Ameriku. A možno, keby bol dôsledný, nechyboval a neexperimentoval, pripláva naozaj do Indie. Rázom by bol z jeho projektu nie úspešný ale iba priemerný projekt. Chyby teda nemusia nutne viesť

k zániku projektu, ba nemusia pre neho prestavovať ani nijaké riziko. Domnievam sa však, že tak, ako v iných oblastiach aj tu, je potrebné dodržiavať ten správny pomer rizika a potencionálneho zisku. Nie je predsa mysliteľné, aby sme riskovali osud celého projektu napríklad na jedincovi, o ktorom je známe, že nie je ten z tých spoľahlivých. Na druhej strane úspech projektu nezávisí len na dodržaní normy STN ISO 690 a iných chuťoviek. Aj z týchto dôvodov by som nebral chyby ako nejaký strašiak, ktorého sa treba strániť. Rozdelil by som ich však na malé a veľké. Malé ti pomôžu, veľké ublížia. Cieľom je zbaviť sa tých veľkých, fatálnych. A práve tento stav môžeme dosiahnuť včasným odhalením chýb malých. Prirovnal by som to k rakovine. Je lepšie už na začiatku vedieť, že tam je a riešiť vzniknutú situáciu, než sa to dozvedieť, keď sú už následky fatálne. Z chýb sa preto treba tešiť, najmä keď sú malé a skoro odhalené.

Keď nás zabíjajú muchy

Samozrejme okrem toho, že sa z nájdených chýb tešíme, treba v prvom rade zabezpečiť aby sa neopakovali. Táto požiadavka je však celkom triviálna. Aj priemerne inteligentný človek pochopí, že ak na ceste spadne do jamy, najbližšie, keď pôjde po rovnakej ceste, jamu obíde. To však nestačí. Ideálne by bolo, ak by do jamy nespadol vôbec. Prečo padnúť do jamy, do ktorej padol už niekto iný? Práve z takého dôvodu existujú rôzne kontrolné zoznamy možných rizík softvérových projektov. Nie je vždy nutné preberať existujúce zoznamy, ale myslím, že je to dobrý začiatok. Ideálne by bolo postupne vytvárať vlastný zoznam, ktorý bude presne zodpovedať parametrom našich projektov. Myslím si, že práve toto je cesta, ktorá by v manažmente rizík každého softvérového projektu nemala chýbať. Domnievam sa, že môže ísť o relatívne účinnú metódu. Ťažko totiž hľadať chyby a riziká o ktorých nič nevieme. Naopak, ak vieme pomenovať riziko, vieme aj ľahšie nájsť chybu. Netvrdím, že ide o všemocnú metódu, to nie. Určite však pri špecifických bodoch v spomenutých zoznamoch ide o metódu, ktorej venujeme veľmi málo námahy a získame obrovský efekt. A o to práve ide. Nedajme sa zabíjať muškami, keď stačí tak málo.

Všetko je o ľuďoch

Mám v tíme správnych ľudí? Toto je prvý a najdôležitejší bod v mojom kontrolnom zozname. Koniec koncov sú to práve ľudia v tíme, ktorý vytvárajú a na ktorých závisí celý projekt, nikto iný. Dôraz však nekladiem len na ich znalosti a skúsenosti. Podstatný je ich charakter, motivácia, snaha a najmä dobrá komunikácia jednotlivých ľudí v tíme. Načo by vám bol tým plný geniálnych programátorov a manažérov s úžasnými referenciami, keď jednotlivci v tíme nie sú schopní z nejakých dôvodov medzi sebou normálne komunikovať? Okrem komunikácie je tiež treba zabezpečiť, aby všetci účastníci softvérového projektu mali rovnako veľkú motiváciu. Inak zistíme, že časť tímu sa skutočne snaží, aby výsledok bol čo najlepší a druhá časť sa snaží, aby si odrobili to potrebné minimum a ani o kúsok viac.

Manuál iba pre amatérov ?

Určite sa pri vývoji softvérového projektu stane, že potrebujeme pracovať s nástrojom, ktorý vidíme prvýkrát. Porovnal by som to so situáciou pri nákupe nového zariadenia,

napríklad mp3 prehrávača. V takom prípade väčšina z nás čo najrýchlejšie vybalí novú hračku, okamžite strká usb konektor do počítača, pričom neberie ohľad na rôzne varovania a upozornenia na tom nepodstatnom bielom papieri. Ale pozor, ako to neraz dopadlo ? Búchali sme do zariadenia, stláčali aj nemožné kombinácie kláves v nádeji že sa to spustí. Týmto sme stratili omnoho viac času, ako keby sme si prečítali zdanlivo nepodstatnú informáciu napísanú na dvoch riadkoch v časti "ako začať". Samozrejme netvrdím, že treba čítať celé manuály. Prečítanie krátkej pasáže so základnými informáciami o prvom použití zariadenia sa však mnohokrát oplatí. V opačnom prípade pravdepodobne nehrozí riziko pre celkový výsledok projektu. Takéto konanie však prispeje k vytváraniu problémov, ktoré ani nie sú problémami. Zbytočne strácame čas a energiu, ktoré budú tak cenné pri ozajstných problémoch.

Bez správnej technológie to nejde

V dnešnej dobe sú odvetvia, kde rozhodujú aj milisekundy. Preto je výber technológie vývoja softvérového projektu veľmi dôležitý. V prípade nesprávneho výberu je tu riziko neúspechu celého projektu. Nie je to totiž chyba, ktorá sa dá jednoducho opraviť. Samozrejme, pre výber technológie potrebujeme vedieť presne špecifikované presné požiadavky, čo môže byť niekedy problém. Zákazník respektíve zadávateľ projektu ako laik len ťažko odhadne všetky technologické požiadavky na výsledný produkt. Preto si myslím, že kľúč k výberu správnej technológie je v čo najhodnovernejšej simulácii podmienok zákazníka. Týmto spôsobom môžeme lepšie pochopiť požiadavky zákazníka a hlavne jeho nekonkrétne predstavy premeníme na presnú špecifikáciu. V prípade simulácie v určitých umelých podmienkach, sa nám produkt môže javiť ako bezproblémový a v skutočnosti bude pre zadávateľa bezcenný.

Plán, ktorý zlyhá

Okrem spomenutých ťažkostí samotného vývoja predstavuje veľké riziko práve neobjektívne plánovanie softvérového projektu. Podľa B. W. Boehma [2] najdôležitejšie rizikové faktory pre úspešnosť projektu sú realistický odhad potrebného času a nákladov na projekt. Nedá sa nesúhlasiť. Sú to samozrejme aj pre zákazníka tie najdôležitejšie parametre. Práve tu je hlavný problém. Zákazník by rád počul konečnú sumu, dátum kedy bude projekt hotový, následne si zoradí ponuky od najlepšej po najhoršiu a je vymaľované. Manažér projektu počas jeho vývoja samozrejme uisťuje zákazníka o splnení dohodnutého termínu až do chvíle, kedy je už zrejmé že sa to nestihne. V tom lepšom prípade zákazník si je vedomý možných časových sklzov. V opačnom prípade môže ísť o ďalší z neúspešných projektov. Všetko by však bolo v poriadku, pokiaľ by úspešnosť projektov, ktoré skončia na čas a bez prekročenia finančného limitu bola vysoká. V skutočnosti je to podľa niektorých zdrojov iba slabých 20%[1]. Nedostatky jednoznačne vidím v príliš optimistickom stanovení potrebného času a financií na projekt. Aké sú teda príčiny takéhoto optimizmu?

Ružové okuliare

V skutočnosti stanovenie času potrebného pre úspešné ukončenie softvérového projektu môže byť približne tak náročné, ako tipovanie výsledku športového zápasu. Nikdy totiž nevieme čo nás v priebehu vývoja čaká. Vytvorenie časového a materiálneho plánu je teda založené na našich aktuálnych poznatkoch. Čím máme viacej skúseností a vieme viacej predvídať možné problémy a skryté závislosti, tým presnejší bude plán. Táto úvaha logicky prináša ďalšiu. Keby každý pracoval len s takými technológiami a prostriedkami, ktoré dokonale pozná, bolo by plánovanie na 100% presné? Myslím, že by toto číslo síce nebolo 100% avšak bolo by dostatočne vysoké nato, aby sme tieto problémy nemuseli riešiť. Otázkou však je, či je možné zabezpečiť, aby pri každom vývoji projektu boli k dispozícii práve takí ľudia, akých potrebujeme. Myslím, že to možné nie je práve preto, že ak by každý robil len to čo dokonale ovláda, nikdy by sa nenaučil nič nové.

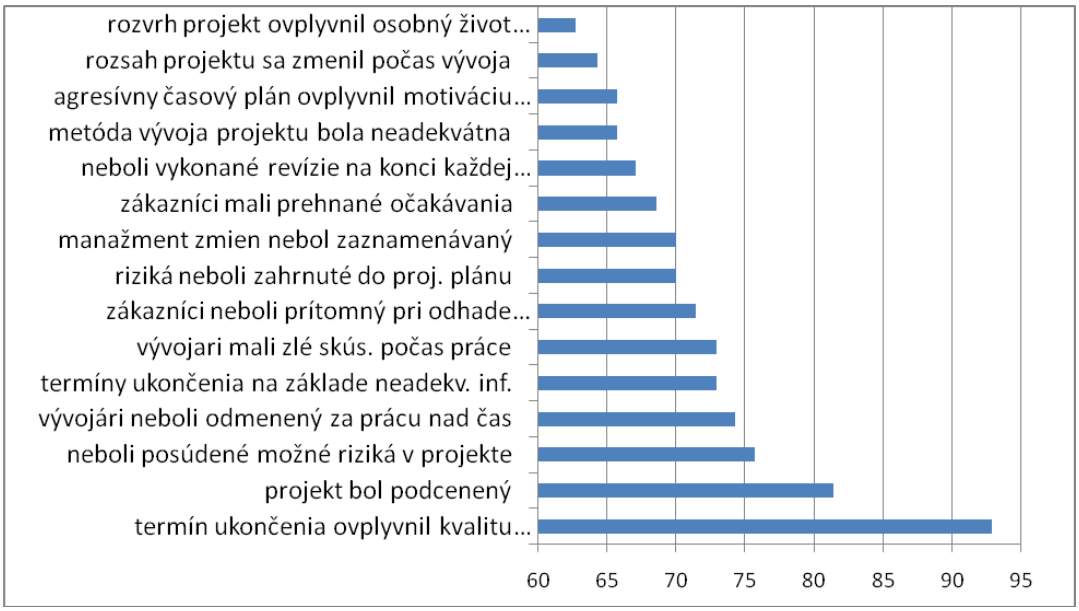
Ďalšie úskalie vidím v skutočnosti, keď sa zodpovedné osoby snažia odhadnúť zložitosť projektu ako ideálne možné riešenie (best case solution/Angl.) bez ohľadu na predvídateľné a veľmi triviálne skutočnosti. Neuvažuje sa napríklad o prestávkach, diskusiách, zvratoch v projekte, či všeobecne sa neuvažuje o istej rezerve. Je to v poriadku? Myslím, že najlepšie sa dá táto otázka zodpovedať tak, že si predstavíme seba v úlohe projektového manažéra a práve sa snažíme odhadnúť spomenuté atribúty pre nový projekt. Povieme zákazníkovi pravdu respektíve pridáme dostatočnú časovú a finančnú rezervu ? Riskneme, že sa ocitneme na podlahe jeho preferencií ? Prečo na podlahe? No práve preto, že bežného zákazníka nezaujíma nič iné, iba cena a odhadovaný čas. Použije magické tlačidlo pre zoradenie a jeho preferencie sú jasné. Myslím, že práve z toho dôvodu žiadny manažér nechce byť pomalší a drahší na prvý pohľad ako jeho konkurenti. Je efektívnejšie nastoliť nazačiatku veľmi výhodné podmienky a potom sa snažiť vyjednávať so zákazníkom o ich upravení, než použiť triezvy pohľad hneď na začiatku. V tom prípade by si nás nikto ani len nevšimol.

Posledným, nemenej dôležitým faktom je, že odhad projektu nerobia tí, ktorí projekt naozaj vyrábajú. Naopak tieto údaje sú odhadované manažérmi. Oni síce môžu kvalifikovane posúdiť zložitosť úlohy, avšak len ťažko odhadnú potrebný čas pre konkrétneho vývojára. Vo výsledku manažér nielen zle určí potrebný čas a tým pádom aj financie, ale v dôsledku faktov spomenutých vyššie tento čas aj trochu idealizuje. Manažér vtedy tvrdí, že vývojári sú schopnejší, ako oni samy predpokladajú. Alebo ako píše P. G. Armour : "*I have more confidence in you than you have in you*"[1].

Samozrejme, skutočnosť že projekt prekročil dohodnuté limity nie je ešte dôvod pre jeho totálny neúspech. V skutočnosti je ešte úspechom, ak projekt prekročí spomenuté hranice iba o malú čiastku, rádovo 10% - 20%. Situácia mi pripomína platenie faktúr medzi podnikateľmi. Málo kto dnes platí všetky faktúry na čas. Rovnako je to spôsobené z veľkej časti "idealizovaným" plánovaním. Z ekonomického hľadiska je totiž výhodnejšie akceptovať o istý čas neskoršiu platbu, než si uplatňovať náhrady súdnou cestou, prípadne riskovať stratu zákazníka. Rovnako v softvérových projektoch, zákazník už nemá veľmi na výber a zmenu podmienok akceptuje.

Keď jedno riziko nestačí

Situácia sa však mení, ak sa k „idealizovanému" plánovaniu pridá ďalší vysoko rizikový prvok. N. Cerpa píše: "*Príčinou k zlyhaniu projektu sú zvyčajne viac ako jeden či dva dôvody. Zvyčajne je to kombinácia zlých rozhodnutí*"[3]. Rovnako nemôžem nesúhlasiť aj vzhľadom na skutočnosť, že svoje tvrdenia podložil analýzou sedemdesiatich neúspešných projektov. Ešte zaujímavejšie sú však jeho výsledky.



Obr. 1. Percentuálne zastúpenie syndrómov neúspešných projektov

Nie je žiadnym prekvapením, aké boli najčastejšie 3 príčiny neúspechu. Zdá sa, že prvé dve položky sú medzi sebou o čosi viac naviazané ako ostatné. Svedčí o tom 76% projektov, ktoré trpelo oboma problémami. Koniec koncov ak bol projekt podcenený a vývojári sa teda zákonite ocitli v časovom sklze, nezostávalo im nič iné, iba sa snažiť postupovať neštandardne rýchlo. A práve, keď robíme veci na poslednú chvíľu vzniká najviac chýb. Tretia položka len umocňuje dôležitosť manažmentu risku. Pre mňa najzaujímavejšou časťou grafu sú však 4 položky a to tie, ktoré uvažujú ľudský faktor. Tieto čísla iba potvrdzujú moje tvrdenia z predchádzajúcej kapitoly, všetko je o ľuďoch. Toľko spomínaná dôležitosť motivácie bola dokonca štvrtým najčastejším príznakom zlyhania. Neodmeňovanie zamestnancov za ich nadčasy, rapídne znižuje ich morálku. Nie je žiadnym prekvapením, že rovnako na ľudskú morálku vplýva aj nepríjemné pracovné prostredie respektíve negatívne zážitky a skúsenosti počas práce na projekte. Prekvapením pre mňa osobne však je, že takéto možno na prvý pohľad drobnosti, môžu rapídne ovplyvniť výsledok projektu. Nakoniec, veľké softvérové spoločnosti vedia čo robia, keď sa snažia vytvoriť svojim vývojárom tie najlepšie podmienky na prácu, nech to stojí čo to stojí. Zvýšia tým ich produktivitu, čo sa zákonite prejaví na výsledkoch spoločnosti.

Tenký ľad

Ako som spomenul, najkomplikovanejšie hrozby a riziká softvérových projektov jednoznačne vidím v „ideálnom plánovaní“ a v ľudskom faktore. Myslím že, práve toto sú riziká z kategórie nevyspytateľných, respektíve nie vždy ovplyvniteľných. Ťažko totiž nájsť tých správnych ľudí a to správne pracovné prostredie. Zväčša sa pracuje s tým čo je k dispozícii a nie je možnosť veľkého výberu. „Ideálne plánovanie“ je na tom podobne. Manažéri pri plánovaní nie vždy zapájajú do svojej práce aj vývojárov a keď náhodou, tak ich zaujíma odpoveď na otázku, koľko to bude trvať. Vývojár vedomý si skrytých rizík odpovie konkrétnym číslom a pravdepodobnosťou, pri ktorej toto číslo bude splnené. Takáto odpoveď je nepredstaviteľná pre manažérov či zákazníkov. Dobre túto situáciu vystihol P.G Armour: "*Probabilities and percents were so much mumbo-jumbo to him.*"[1]. Jeho vedúci však požadoval presný dátum, kedy bude vývoj určite hotový a zároveň chcel aby tento dátum bol rovnaký alebo skorší ako ten ktorý on sľúbil zákazníkom. V zadaní dostal teda protichodné podmienky a projekt sa nestihol na čas. Aj týmto príkladom som chcel demonštrovať, že vývoj softvérového projektu bude vždy určitý boj o čas na tenkom ľade medzi vývojármi, manažérmi a zákazníkmi.

Použitá literatúra

1. Armour, P. G. 2007. Twenty percent. Commun. ACM 50, 6 (Jun. 2007), 21-23. DOI= http://doi.acm.org/10.1145/1247001.1247020
2. Barry W. Boehm, Software Risk Management: Principles and Practices, IEEE Software, v.8 n.1, p.32-41, January 1991
3. Cerpa, N. and Verner, J. M. 2009. Why did your project fail?. Commun. ACM 52, 12 (Dec. 2009), 130-134. DOI= http://doi.acm.org/10.1145/1610252.1610286.

Annotation

To prevent being broken

There is only few projects which finishes exactly the same as they are expected. In the time limit, without exceeding the budget and with providing sufficient quality. On the contrary, a lot of the projects finish with slightly overrun budget or time. And even this case is the one of the better. Why is it so? It is because of "unexpected" events and their consequences. In the most cases, these unexpected events can be expected and they should be expected. The challenge is to identify risks in software projects, especially those which often are neglected but still they are important in development. The paper also deals with the causes of improper planning and other interesting risk of software project development. The aim of this paper is also to answer the question whether prevention of possible risks is effective or even possible, and in which situations it is so.