Analýza a plánovanie rizík v softvérovom projekte

JÁN HRIC

Slovenská technická univerzita
Fakulta informatiky a informačných technológií
Ilkovičova 3, 842 16 Bratislava
janhricbs@gmail.com

Abstrakt. Tvorba každého projektu so sebou prináša riziká. Sú riziká, ktoré možno len ťažko predvídať, a sú také, ktorým sa dá vhodným manažmentom vyhnúť. Správny manažment softvéru zahŕňa aj identifikáciu možných rizík a plánovanie ich predchádzania. Mnoho projektov zlyhá alebo sa skomplikuje ich vývoj práve zanedbaním tejto sféry manažmentu a považovaním všetkých rizík za dačo nepredvídateľné, s čím sa treba vyrovnať, až keď riziko nastane. Táto esej ukazuje potrebu včasnej analýzy možných rizík a ich vhodného plánovania, opisuje ich výhody aj nevýhody pre rôzne projekty. Diskutuje rozličné pohľady na danú problematiku a z nich sa pokúša odvodiť vhodný postup najmä pre menšie tímové projekty.

Podstata analýzy rizík

Softvérový manažment je ešte stále veľmi mladý. Suvisí to s faktom, že samo softvérové inžinierstvo sa začalo vyvíjať len nedávno. Ako pri každom novo vzniknutom odbore jeho jednotlivé súčasti sa vyvíjajú postupne a ich zavedenie do praxe trvá istý čas. Manažment rizík nie je žiadnou výnimkou. Štatistiky ukazujú opakujúcu sa nedbanlivosť pri vývoji softvéru a z toho plynúce veľké straty. Správna analýza a plánovanie rizík dokáže zmierniť dopad rizika na vývoj softvéru a často napomáha úplnému predchádzaniu rizík.

Riziko

Každý sa určite stretol s pojmom riziko a vie, čo približne znamená. Jedna z definícií opisuje riziko ako pravdepodobnosť, že sa nepodarí dosiahnuť požiadavky na cenu, výkon alebo čas a zároveň je to dôsledok tohto zlyhania [1]. Často sa pojem riziko chápe iba ako pravdepodobnosť a zanedbáva sa jeho dôsledková časť.

Manažment v softvérovom inžinierstve, október 2006, s. 1-7.

Riziká sprevádzajú každú činnosť v bežnom živote, softvérové inžinierstvo nie je žiadnou výnimkou. Hoci sa mu nedá stopercentne vyhnúť ani odhadnúť všetky možné zdroje, sú známe všeobecné zásady, ktorých dodržiavanie a vyvarovanie sa chýb napomáha vývoj softvéru značne zjednodušiť. V tabuľke 1 je uvedené rozdelenie zdrojov rizík pri vývoji softvéru [1].

Skupina rizík	**Zdroj rizika**
Projekt	nepresné, nereálne či nestabilné požiadavky
	nezúčastnenosť používateľa
	podcenenie komplexnosti projektu či jeho dynamickosti
Vlastnosti projektu	slabý výkon (chyby, zlá kvalita)
	nereálna cena či plán
Manažment	neefektívny manažment projektu
Obsluha	neefektívna integrácia, inštalácia a testovanie, riadenie kvality, špeciálna či systémová technika
	nepredvídané ťažkosti s používateľským rozhraním
Pracovné prostredie	nezrelý či neodskúšaný návrh, proces, technológia
	nevhodné pracovné plány či konfiguračné nástroje
	nevhodné metódy, výber nástrojov či nepresná metrika
	slabý výcvik
Iné	nevhodný či prehnaný proces dokumentácie či revízie
	problémy s legálnosťou či zmluvou
	strata na hodnote (pridlhý plán)
	nepredvídané ťažkosti so zmluvnými položkami
	nepredvídaná cena údržby a(lebo) podpory

Tab. 1. Súhrn kľúčových zdrojov rizík [1]

Dva plány

Analýza rizík znamená identifikáciu a určenie pravdepodobnosti výskytu a závažnosti rizika. Aby vypracovaná analýza mala zmysel, musí sa odraziť na plánovaní.

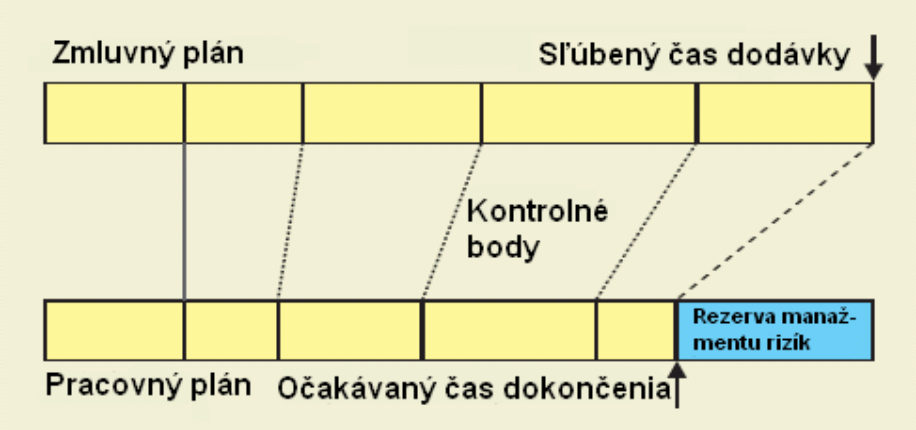
Hodnota, ktorú plánovanie do projektu vnáša, sa často podceňuje. Súvisí to najmä s faktom, že plány sa takmer nikdy nesplnia do bodky. Prečo teda plánovať, ak priebeh vývoja je aj tak iný? Odopveďou je práve prínos plánovania do projektu. Na plán

netreba nazerať ako na niečo, čoho sa projektový tím musí držať. Lepšie je pozrieť sa naň zo strany výhod pre projekt. Plán plní tieto hlavné úlohy [3]:

1. zverejnenie cieľov projektu
2. príprava na pravdepodobné udalosti
3. predvídanie nepravdepodobných udalostí
4. poskytnutie zdrojov

Poskytnuté zdroje sú potrebné na očakávanú prácu a zohľadňujú aj rozumnú časť „predvídaných nepredvídaných“ udalostí podľa ich predpokladaného rizika. Samozrejme aj pri prepracovanom plánovaní nastávajú odchýlky od plánu. Riziku sa teda v žiadnom prípade vyhnúť nedá. Ak riziko nastane a manažment sa s ním nevie vyrovnať, kto zaň platí? Ak sa dodržia termíny a cenové náklady, obeťou je *zákazník*. Dostane produkt nižšej kvality, než očakával. Ak sa manažment rozhodne riziko za každú cenu skryť, doplatí na to *projektový tím*, ktorý musí pracovať nadčas za rovnú cenu. Jedným z riešení tohto problému je stanovenie dvoch plánov.

Pracovný plán je záväzný pre projektový tím. Týmto plánom sa riadi spoločnosť. *Zmluvný plán* je ten, ktorý spoločnosť oznámi zákazníkovi. Zákazník nevie nič o existencii pracovného plánu, zmluvný plán je preňho jediný. Na obrázku 1 je znázornený rozdiel medzi týmito plánmi [3]. Pri ich vytváraní sa rátalo s rezervou, ktorá by mala pokryť vzniknuté riziká.



Obr. 1. Tvorba dvoch plánov [3]

Rezerva manažmentu rizík predstavuje najvyššiu predpokladanú časovú stratu pri vývoji softvéru. Počas vývoja sa pri každom kontrolnom bode určí, do akej miery sa darí pracovný plán plniť. Ak vývoj za plánom zaostáva, posunie sa čas naplánovania ďalších kontrolných bodov a uberie sa z rezervy. Hovorí sa tomu *odkúpenie rizika*.

Čo s projektovým tímom a jeho vzťahu k zmluvnému plánu? Mala by ho spoločnosť pred nimi skryť? Ak to zistia, môžu sa cítiť „podvedení“ a predpokladať, že

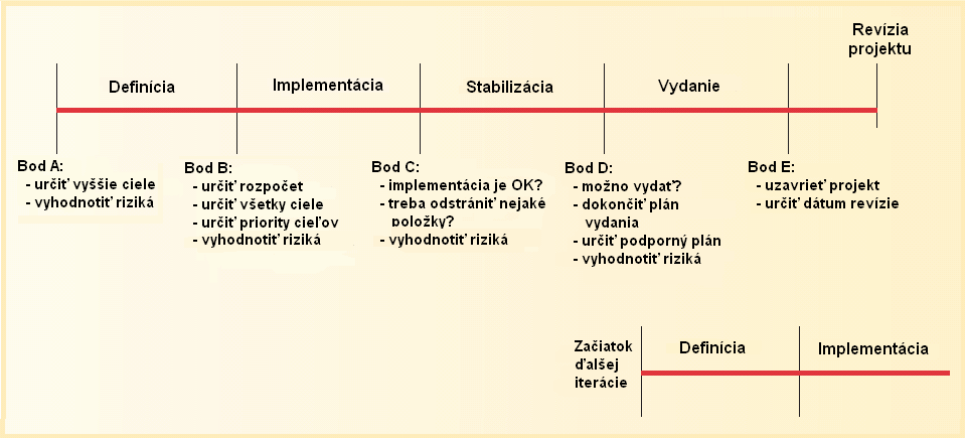
kratší plán je len nástrojom na prinútenie, aby pracovali viac. Ak však spoločnosť vysvetlí svoje plány a oboznámi ich so zmluvou, nemala by nastať potreba utajenia „skutočného“ plánu.

Iteratívny prístup k tvorbe projektu

Tradičný projektový manažment hovorí o troch rozmeroch projektu: rozsah (požiadavky), zdroje a čas. Tieto rozmery spolu súvisia; projekt väčšieho rozsahu potrebuje viac zdrojov a času, menej času zas vyústi do potreby viacerých zdrojov či menšieho rozsahu [2].

Ak má byť manažment úspešný pri zvládaní rizík, vývoj sa musí stať pružnejším, prispôsobivým nepredvídaným udalostiam. Jedným zo spôsobov dosiahnutia tejto pružnosti je rozdelenie projektu na menšie podprojekty. Vývoj jedného podprojektu predstavuje jednu iteráciu. Nemala by trvať dlhšie než dva-tri mesiace. Každá iterácia (okrem prvej) pridáva novú funkcionalitu k projektu z predošlej iterácie.

Tento prístup môže trocha pripomínať známy špirálový model. Rozdiel je však v bode ukončenia iterácie. Pri špirálovom modeli nemusí každá iterácia končiť použiteľným produktom, pričom iteratívny prítup vyžaduje *funkčný softvér po skončení každej iterácie*. Tento softvér môže byť verejné vydanie produktu, ktorý sa hneď nasadí do praxe. Niekedy je to však iba vylepšená vnútorná verzia softvéru, ktorá sa k zákazníkovi nedostane. Obrázok 2 znázorňuje jednotlivé fázy každej iterácie [2].



Obr. 2. Štyri fázy jednej iterácie počas vývoja softvérového projektu [2]

Vo fáze d*efinície* sa určia hlavné ciele projektu. Musí sa pritom pamätať na to, že projekt má predstavovať funkčný produkt, nie nejakú „ukážku“. Určenie nereálnych cieľov je ideálny spôsob, ako projekt vopred odsúdiť na neúspech. Na konci tejto fázy sa určia priority cieľov.

Implementácia predstavuje samotný vývoj projektu. Tím pracuje na dosiahnutí cieľov v poradí priority. Týždenné vydávanie „ukážkového“ produktu pomáha zachytiť možné problémy v tejto fáze. Malo by tu byť zavedené i pravidelné testovanie. Fáza končí revíznym stretnutím, aby sa zistilo, či je implementácia vhodná a kompletná.

Stabilizácia pozostáva zo silného testovania kvality a opravovania hlavných chýb. Softvér testuje omnoho viac používateľov než v predošlých fázach. Dokončí sa dokumentácia a určia sa podporné plány. Fáza končí revíznym stretnutím, kde sa uvidí, či je projekt schopný vydania. Súčasne s ďalšou fázou (vydanie) môže začať nová iterácia.

Dĺžka fázy *vydania* je často vopred neurčiteľná. Predstavuje vydanie produktu zákazníkovi a jeho podporu. Najmä počas tejto fázy sa zbierajú informácie o požiadavkách a možných vylepšeniach do ďalšej iterácie.

Autorov postoj

Nemám veľké skúsenosti s manažmentom softvéru. Ako aktívny vývojár softvéru mám však už istý názor na analýzu a plánovanie nielen rizík, ale i všeobecne. Moje skúsenosti súvisia najmä s vývojom softvéru jednotlivcom a v dvojčlennom tíme, čiže ide predovšetkým o projekty malé, s nevyvinutou zložkou manažmentu. Môžem povedať, že sa stávajú i prípady, keď ide všetko podľa plánu. Sú to plány na pár dní a stáva sa to najviac ak raz z desaťkrát. Vo všetkých ostatných prípadoch, nech sa problém zdá byť akokoľvek jednoduchý, sa vždy vyskytnú komplikácie. Mnohé z nich sa človek časom naučí predvídať, no daktoré prídu nečakane a po ich príchode asi každému preblysne hlavou: „Tak toto by som v živote nečakal“.

Ignorácia možných rizík a vývoj softvéru bez čo i len myšlienky na ne má stále svoje miesto pri písaní malých jednoúčelových programov. Nemožno prakticky hovoriť o projekte, lebo ich vývoj zväčša končí v priebehu dňa. Dokonca aj o správnosti použitia slova vývoj by sa dalo diskutovať. Pri tvorbe takéhoto softvéru sú často zanedbané všetky zložky manažmentu, nie je preto oblasťou záujmu tejto eseje. Treba však vedieť, že aj písanie malých programov má opodstatnenie a uznávam, keď autor, najmä ak je skúsený v danej oblasti a vopred má o tvorbe programu jasnú predstavu, zanedbá štandardný postup s cieľom zrýchliť jeho vývoj. Metóda „pokus, omyl“ je tu často rýchlejšia.

Používanie *dvoch plánov* je rozumná stratégia podniku. Často sa snažím, hoc vyvýjajúc softvér sám, nechať si rezervu a zákazníkovi povedať iný plán než ten, ktorého sa držím. Niekedy si však pod jeho nátlakom nepremyslím, koľko času naozaj budem potrebovať, a určím príliš skorý termín. To je najlepší návod na nesplnenie plánu, prípadne zbytočné ponocovanie v snahe daný plán splniť. Určovanie správnej rezervy nechávam na „intuíciu“, nerobím serióznu analýzu rizík, no aj tak je to veľká pomoc. Ak zákazník s navrhnutým termínom súhlasí, dva plány často vedú k jeho spokojnosti a zároveň k rovnomernému rozloženiu mojich síl.

Vo väčších spoločnostiach to nemôže byť inak. Rozdiel je len v dôkladnejšej analýze rizík a určení veľkosti rezervy. Hovoriť o klamaní projektového tímu je určite nezmysel, veď aj ľudia v tíme si musia uvedomovať potrebu včasnej dodávky

kvalitného produktu. Napriek tomu si myslím, že najlepší spôsob je ukázať im len pracovný plán a o zmluvnom pláne povedať len toľko, že existuje a od pracovného sa líši. Takto nebude spoločnosť nikoho klamať a vyhne sa neochote členov tímu splniť svoje úlohy v termíne s argumetmi typu „veď ešte máme čas“.

S *iteratívnym vývojom softvéru* mám v zjednodušenej podobe tiež isté skúsenosti. Fakt, že po každej iterácii vznikne funkčný produkt, je veľkým plusom. Často zákazníci kladú väčší dôraz na rýchlu dodávku softvéru než na jeho stopercentnú funkčnosť či jednoduché a zrozumiteľné ovládanie. Vďaka iteratívnemu prístupu sa dá vyvinúť prvá verzia softvéru za relatívne krátky čas. Zákazníka treba upozorniť, že nejde o finálny produkt a akékoľvek neskoršie zmeny podľa jeho želania sú možné, pričom zatiaľ môže používať aktuálnu verziu. Tento prístup považujem za nevyhnutný pri vývoji väčšieho softvéru. Navrhnúť naraz softvér spĺňajúci všetky požiadavky je (takmer) nemožné a snaha o to predĺži jeho vývoj o nezanedbateľný čas. Časť rizík sa týmto presunie na zákazníka, je to však s jeho vedomím. Vývojári sa môžu venovať práci na ďalšej verzii, kým zákazník už s väčšou či menšou spokojnosťou produkt používa a je schopný identifikovať jeho slabé stránky.

Pri aplikácii manažmentu rizík na *tímový projekt* vidím cestu v kompromisoch. Nemožno sa riadiť rovnými pravidlami ako pri vývoji softvéru jednotlivcom, v tíme je predsa niekoľko ľudí a ich prácu treba koordinovať. Zo skúsenosti vývoja softvéru vo dvojici, čo je už taký malý tím, viem, že platí „viac hláv viac vie“ a kolega dokáže byť veľkou pomocou, či už pridaním nových nápadov alebo opravou chýb po kolegovi. Zároveň však prináša nové riziká. Asi najväčším je nesplnenie svojich povinností resp. jeho vysoká chybovosť. Chybu síce spravil on, no má to dopad na celý tím. Rezerva pri plánovaní musí brať ohľad aj na takéto situácie. Poznať osobnosti tímu je v tomto prípade veľkou pomocou.

Tímový projekt zložený z tímov s malým počtom členov nemožno, na druhej strane, porovnávať ani s veľkými spoločnosťami. Koordinovať prácu šesť ľudí je dačo iné než koordinácia šesťdesiatich. Aj riziká sú tu menšie. Pri veľkých tímoch si môže spoločnosť dovoliť vyhradiť aj niekoľkočlenný manažérsky tím, pri malých aj jeden človek venujúci sa výhradne manažmentu môže tím nie posilniť, lež oslabiť. Preto vidím cestu v kompromisoch. Analýze rizík treba venovať istý čas, no zároveň treba vedieť odhadnúť, kedy je úsilie vynaložené na ich skúmanie „drahšie“, než by bola ich ignorácia.

Aj v tímovom projekte vidím miesto pre uplatnenie iteratívneho prístupu. Nepôjde síce o súčasné používanie jednej verzie produktu zákazníkom a práce na vývoji verzie nasledujúcej, no budú sa v iteráciách vyvíjať interné verzie a tie sa bude tím snažiť vylepšovať.

Záver

Ako manažment zohráva podstatnú úlohy pri tvorbe softvérového projektu, tak analýza a plánovanie rizík predstavuje silný nástroj v rámci manažmentu. Opodstatnenie a potreba dôslednosti vypracovania analýzy a plánovania je rôzna pri rozličných

projektoch. Ak však chce byť manažment úspešný, musí jej venovať dostatotok času. Nejestvuje zaručený recept na zvládnutie tejto problematiky, no metódy priblížené v tejto eseji môžu pomôcť zvládnuť hlavné riziká každého projektu. Do akej miery bude ich použitie úspešné, závisí hlavne od manažéra a jeho skúseností. Ja sám som zvedavý, ako sa osvedčia pri tímovom projekte.

Použitá literatúra

1. Conrow, E.H., Shishido, P.S.: *Implementing Risk Management on Software Intensive Projects*. IEEE Computer Society (1997), 83-89.
2. Murthi, S.: *Preventive Risk Management for Software Projects*. IEEE Computer Society (2002), 9-15.
3. Armour, P.G.: To Plan, Two Plans. Communications of the ACM, Vol. 48, No. 9 (2005), 15-19.

Annotation

Software Project Risk Analysis and Planning

Production of every project comes with risks. There are risks which can be hardly predicted and there are those which can be avoided using suitable management. Proper software management includes also an identification of possible risks and a planning how to avoid them. Many projects fail or their developement gets complicated just because of omission of this management field and treating all risks as something unpredictable which has to be handled only when the risk occurs. This paper shows the need of early analysis of possible risks and their suitable planning. It describes its advantages and disadvantages for different projects. It discusses various opinions about given problematics and based on them tries to find a suitable procedure especially for small team projects.