Analýza a plánovanie rizík v softvérovom projekte

FILIP ŠTIGLIC

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

Abstrakt. Nové technológie, stúpajúce nároky, rýchle pracovné tempo. To je len malá skupina rizík ktorá ovplyvňuje prácu na softvérovom projekte. Často krát, práve podcenenie rizík môže mať obrovský dopad na ďalší priebeh vývoja softvérového projektu. Preto je nevyhnutné sa týmito rizikami zaoberať, identifikovať ich a snažiť sa zvoliť najvhodnejšie riešenie v prípade ich výskytu, no najlepšie im úplne predísť, čím sa minimalizuje dopad na výsledný produkt. Práve to má za úlohu analýza a plánovanie rizík v softvérovom projekte. Účelom tohto dokumentu je poskytnúť pohľad na najčastejšie riziká, porovnať metodiky eliminácie rizík z pohľadu softvérového inžinierstva a extrémneho programovania a taktiež porovnať vhodnosť ich začlenenia, či už do konkrétnych softvérových projektov alebo tímov.

Riziko z pohľadu histórie a súčasnosti

Riziká sú súčasťou nielen softvérových projektov, ale všetkých projektov vo všeobecnosti. S postupným rozvojom informačných technológii, aj vďaka novým možnostiam, sa samotné softvérové projekty stali rozsiahlejšími a začali sa objavovať prvé problémy s ich vývojom. Tieto problémy vyústili v softvérovú krízu v rokoch 1968 a 1969 a podnietili vznik softvérového inžinierstva. Táto kríza sa prejavovala a ešte stále prejavuje neúnosným predlžovaním projektov, nízkou kvalitou výsledných produktov, nemožnou alebo príliš problematickou údržbou, nízkou produktivitou práce programátorov a podobne. Na vážnosť situácie poukazuje štúdia Standish Group Report z roku 1995 [6], kde je ročne v Spojených štátoch investovaných približne 250 biliónov amerických dolárov do približne 175 000 softvérových projektov. Z toho 31.1% projektov je zrušených pred samotným dokončením, polovica (52.2%) prekročí pôvodné náklady až o 89%. Iba 16.2% zo všetkých projektov je dokončených načas a bez prekročenia plánovaného rozpočtu. Vo väčších spoločnostiach je situácia ďaleko horšia. Tu je dokončených na čas a v rámci rozpočtu len 9% projektov. Ďalším problémom je, že aj keď je projekt úspešne dokončený, vo väčšine prípadov obsahuje

Manažment projektov softvérových a informačných systémov, október 2008, s. 1-9.

len 42% pôvodne plánovaných funkcii. Menšie spoločnosti si vedú štatisticky oveľa lepšie, na koľko 78.4% ich produktov je úspešne dokončených a obsahujú vo väčšine prípadov minimálne 74.2% pôvodne plánovaných funkcii. Podľa môjho názoru bola softvérová kríza nevyhnutná, pretože podnietila rozvoj softvérového inžinierstva, čo prispelo k zvýšeniu samotnej efektivity práce, lepšiemu využitiu samotných zdrojov, či už ľudských, technických, ale aj finančných, vďaka čomu je možné v súčasnosti vytvárať neporovnateľné zložitejšie systémy. Prv než sa však budeme môcť zaoberať samotným manažmentom, je nutné zadefinovať pojem rizika.

Riziko a jeho vzťah k softvérovému projektu

Riziko – možnosť utrpieť stratu, poškodenie, nevýhodu, alebo zničenie (Webster Dictionary). Barry W. Boehm [3] v roku 1980 počas svojho pôsobenia v odvetví Americkej obrany zostavil top desiatich najčastejších rizík v softvérových projektoch:

1. Nedostatok personálu
2. Nerealistické rozvrhy a rozpočty
3. Vytvorenie inej(než požadovanej; zlej) funkcionality
4. Vytvorenie nevyhovujúceho používateľského rozhrania
5. Pozlátenie systému
6. Spojite zmeny požiadaviek
7. Nedostatky v externe vytvorených moduloch
8. Nedostatky v externe zabezpečených úlohách
9. Nedostatok výkonu v reálnom čase
10. Precenenie technológie

Osobne si však myslím že tento zoznam rizík v súčasnosti neodzrkadľuje situáciu v bežných spoločnostiach a dovolím si tvrdiť že v niektorých prípadoch môže byť organizačne a technologicky zastaraný. Hlavne z dôvodu zmeny niektorých, či už organizačných, alebo technologických postupov (objavili sa nové formy organizácie, nové prostredia vývoja a nové mechanizmy (ako napr. outsorcing)). Tak isto centralizovaný systém architektúry sa v súčasnosti viac sústredí na distribuované systémy. Tým pádom sa menia aj priority jednotlivých rizík. Môj názor podporuje aj zoznam rizík získaný na základe štúdie Keil a spol. z roku 1998 [5], kde skúsení softvéroví manažéri identifikovali a zoradili na základe svojich dlhoročných skúseností, jednotlivé rizika, ktoré sú podľa nich nasledovné:

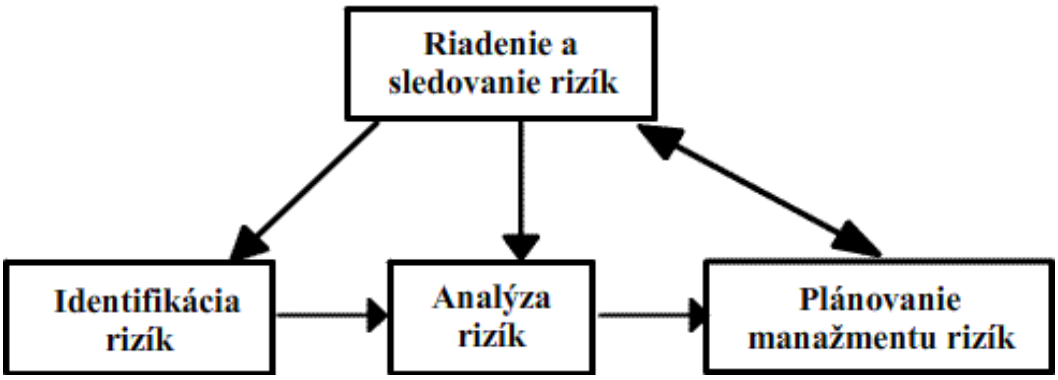
1. Nedostatok angažovanosti vrcholového manažmentu do projektu
2. Nezainteresovanosť zákazníka
3. Neporozumenie požiadavkám

4. Nedostatok angažovanosti používateľa
5. Neúspech splniť požiadavky používateľa
6. Zmena rozsahu/cieľov projektu
7. Nedostatok požadovaných znalostí/zručností/skúseností vývojového tímu
8. Nedostatok zmrazených požiadaviek
9. Zavedenie novej technológie
10. Nedostatočné/nevhodne obsadenie pozícii v tíme
11. Nezhody medzi oddeleniami v organizácii zákazníka

Vo všeobecnosti sa však jednotliví manažéri zhodli, že za najväčšie rizika považujú tie, ktoré nemôžu priamo ovplyvniť, alebo majú minimálnu možnosť ich riadenia. V nasledujúcich častiach dokumentu si uvedieme ako sa s rizikom vysporiadavajú metodiky softvérového inžinierstva, extrémneho programovania, no a napokon si tieto dve metódy porovnáme a vyhodnotíme.

Softvérové inžinierstvo – manažment rizík

Softvérové inžinierstvo obsahuje samostatnú časť venovanú manažmentu rizík. „Väčšina projektov má riziká. Cieľom manažmentu rizík je minimalizovať možnosť zhmotnenia rizika, ak je to možné, alebo minimalizovať účinok jeho skutočného zhmotnenia“ [2]. Hľadá spôsoby ako jednotlivé rizika identifikovať, zanalyzovať, roztriediť, klasifikovať a nakoniec znovu analyzovať vzťahy medzi nimi, ich vplyvy navzájom, ktoré by mohli pri tvorbe softvérového systému nastať. Model manažmentu rizík (Obr. 1) popisuje jednotlivé časti manažmentu rizík. Hlavnou časťou „riadenie a sledovanie rizík“, ktoré si kladie za úlohu reagovať na zmeny v rizikách počas realizácie projektu a monitorovať stav identifikovaných rizikových udalosti.



Obr. 1. Model manažmentu rizík

Aby sme však mohli uplatniť riadenie a sledovanie rizík je potrebne riziká identifikovať, analyzovať a naplánovať manažment rizík.

Identifikácia rizík a analýza rizík

V tejto fáze určíme aké rizika môžu ohroziť projekt a zdokumentujeme ich charakteristiky. Identifikáciu je nutné vykonávať počas celého života projektu, nakoľko je každá etapa vývoja softvéru špecifická, no hlavne pri plánovaní. Pre každé riziko sa určí jeho pravdepodobnosť výskytu, odhad rozsahu škôd pri výskyte. Na miesto presných číselných hodnôt sa pravdepodobnosť nastania udalostí a aj rozsah škôd určuje v stupnici hodnôt napr. veľmi vysoká, vysoká, nízka, veľmi nízka. Niekedy je však náročné, až nemožné určiť pravdepodobnosť výskytu a rozsah škôd. Vtedy sa podľa softvérového inžinierstva nejedna o riziko, ale o neurčitosť. Pri identifikácii rizík je možné použiť viaceré metódy a techniky ako napr.:

- *Zoznam rizík* – klasifikačné schémy zdrojov rizika, formou dotazníkov, používajú sa veľmi často
- *Dekompozícia* – pomáha pri odhalení rizikových časti systému (20% modulov spôsobuje 80% škôd)
- *Analýza rozhodnutí* – skúma sa zdroj všetkých dôležitých rozhodnutí
- *Analýza predpokladov* –skúmajú sa najmä optimistické predpoklady, ktoré môžu predstavovať riziko
- *Interview* – diskusia viacerých zainteresovaných do problému. Používajú sa dotazníky.

Po následnej identifikácii rizík nasleduje ich analýza, kde sa vyhodnotia jednotlivé rizika a určia sa tie, ktoré vyžadujú nejakú akciu a stanoví sa ich priorita. Tu je možné použiť tiež rôzne metódy a techniky, podobne ako pri identifikácii rizík:

- *Určenie vplyvu rizika, resp. očakávanej škody* – Tu môže vhodne poslúžiť matica pravdepodobnosti a vplyvu
- *Rozhodovací strom* – skúmajú sa vzťahy medzi rozhodovaním (očakávaná hodnota škody) a pravdepodobnosťou výskytu udalosti (tak ako ich vníma ten, kto rozhoduje)
- *Simulácia* – analýza správania sa systému napr. simuláciou sieti činnosti v rozvrhu
- *Expertný odhad* - napr. Metoda Delphi - systematická, interaktívna metóda ktorá závisí na skupine vopred vybraných expertoch. V jednom alebo dvoch kolách odpovedajú na konkrétne otázky tykajúce sa danej problematiky. Po každom kole, moderátor anonymne vyhodnotí všetky odpovede, spolu s odôvodneniami ich rozhodnutí a poskytne ich spať účastníkom. Vďaka tomu sú účastníci neustále povzbudzovaní v svetle nových poznatkov, prehodnotiť svoje predchádzajúce odpovede. Tým sa jednotlivé odpovede neustále približujú k danému cieľu a je zabezpečená spätná väzba. Nakoniec, po

dosiahnutí vopred určeného stop kritéria (počet kôl, dosiahnutie konsenzu) sa výsledné odpovede vyhodnotia, získajú sa potrebne výstupy a vyhotoví sa finálna správa. Počas celého procesu je zachovaná anonymita všetkých účastníkov a totožnosť niekedy nie je známa ani po vyhotovení finálnej správy. To prispieva k tomu že jednotliví zúčastnení, nezneužívajú svoje dominantné postavenie, neprichádzajú do priamej konfrontácie s ostatnými a tým pádom môžu voľne vyjadriť svoje názory a v neposlednom rade sú prístupnejší voči názorom druhých. Túto metódu vyvinula spoločnosť RAND v rokoch 1950-1960, kde od vtedy prešla viacerými zmenami.

Výstupom tejto fázy sú identifikovane rizika, usporiadane podľa priority a zoznam udalostí ktorými je nutné sa zaoberať a ktoré možno ignorovať (avšak za cenu akceptácie vzniknutých škôd).

Plánovanie manažmentu rizík

Plánovanie manažmentu rizík je definovanie krokov pri výskyte udalostí spôsobujúcich škodu. Sú tri možnosti:

- *Vyhnutie sa riziku* – Eliminovaním škody, spravidla odstránením príčiny výskytu rizika. Najúčinnejšie je to v tom prípade, ak je možnosť riadenia rizika vysoká a ma vysokú dôležitosť.
- *Redukcia rizika* – chápeme ako redukciu samotných nákladov dôsledku škody. Buď znížením pravdepodobnosti výskytu danej udalosti (použitie novej technológie, metód), alebo znížením hodnoty škody (napr. poistenie)
- *Akceptovanie dôsledkov* – pasívne alebo aktívne (pasívne – akceptovanie rizika a prípadných škôd, aktívna – určenie plánu akcii v prípade škodovej udalosti)

Pri vytváraní plánu reakcii na rizikové udalosti je potrebné brať do úvahy aj požiadavky na vyhnutie sa riziku a taktiež prihliadať na možnosť, že elimináciou niektorých rizík, môžu vzniknúť nové.

Samotný plán sa vytvára pomocou nasledovných metód a techník:

- *Zoznam rizík* – v závislosti od typu rizika obsahuje zodpovedajúce postupy
- *Vypracovanie alternatívnych stratégii*
- *Skúmanie vzájomného vplyvu viacerých rizík*

Extrémne programovanie

Extrémne programovanie je metodika pre malé až stredne veľké tímy, ktoré sa musia vyrovnať so zadaním, ktoré sa rýchlo a často mení alebo nie je úplné jasné. Extrémne programovanie samozrejme tiež pokladá riziko za problém, no na rozdiel od manažmentu rizík, pristupuje k tomuto problému inak. Nesnaží sa riziká identifikovať, analyzovať a vypracovávať akcie v prípade výskytu. Extrémne programovanie so

samotným rizikom počíta počas celej fázy vývoja projektu a jeho metódy sú navrhnuté tak, aby sa riziko úplné potlačilo. Čo je vlastne extrémne programovanie? Je to ľahký, účinný, nie príliš rizikový, pružný, predvídateľný, vedecký spôsob ako pracovať na softvérovom projekte. Prečo ma teda v názve slovo extrémne? Je tomu tak preto, lebo využíva bežne známe princípy, postupy, ktoré však doťahuje do extrémov:

1. Krátke vývojové cykly medzi jednotlivými verziami - nie týždne, mesiace, roky, ale sekundy, minutý, hodiny.
2. Zákazník je súčasťou vývojového tímu – vďaka krátkym vývojovým cyklom je zabezpečená nepretržitá spätná väzba
3. Schopnosť pružne a rýchlo reagovať na meniace sa potreby a požiadavky
4. Dvojica programátorov pracuje spoločné – úzka spolupráca
5. Vývoj je riadený pomocou série vytvorených automatizovaných testov navrhnutých nielen programátormi, ale aj samotnými zákazníkmi. Najprv sa testuje a až potom sa pridávajú nové funkcie. Testuje sa hoci aj niekoľko kráť denne
6. Evolučný proces pri vytváraní návrhu, ktorý prebieha neustále počas celej doby vývoja
7. Spoliehanie sa na verbálnu komunikáciu, testy, a zdrojové kódy, ktoré slúžia na komunikáciu, ale aj ako dokumentácia.
8. Jednoduchosť – výsledný systém ma byť čo najjednoduchší a pritom spĺňať presné požiadavky

Stručné povedané, extrémne programovanie sľubuje zníženie projektových rizík, zlepšenie schopnosti reagovať na zmeny a zlepšenie produktivity počas celého životného cyklu projektu. Pre najlepšie pochopenie je lepšie uviesť niektoré rizika a spôsoby ako sa s nimi extrémne programovanie vysporiadava:

- *Meškanie Harmonogramu* – Príde deň uvedenia projektu do praxe a my musíme zákazníkovi povedať, že systém nebude hotový skôr ako za pol roka. Riešenie: Extrémne programovanie vyžaduje krátke vývojové cykly uvoľňovania nových verzii. Vďaka úzkej spolupráci so zákazníkom je tu prítomná neustála spätná väzba a projekt „nevybočí“ zo správneho smeru, čím sa zmierňuje riziko meškania.
- *Krachujúci systém* – Systém je úspešne uvedený do prevádzky, avšak vďaka obrovskému množstvu chýb, sú náklady na údržbu a zmeny, príliš vysoké, až sa nakoniec situácia vystupňuje na toľko, že systém musí byť nahradený iným. Riešenie: Extrémne programovanie vytvára a udržuje kompletnú sadu testov, ktoré sú spúšťané pred a po každej zmene, niekedy aj niekoľkokrát denne. Tým sa udržuje projekt v prvotriednej kondícii a chybám nie je umožnené hromadiť sa.

- *Miera poruchovosti* – Systém je úspešne uvedený do prevádzky, ale miera poruchovosti je taká vysoká, že nakoniec muší byt stiahnutý. Riešenie: Extrémne programovanie testuje systém z pohľadu programátorov, ktorí vytvárajú testy pre jednotlivé funkcie a tiež z pohľadu zákazníkov, ktorí vytvárajú testy pre jednotlivé časti systému.
- *Nepochopenie zadania* – Systém je vytvorený, ale vôbec nerieši problém, pre ktorý bol určený. Riešenie: Na koľko Extrémne programovanie, požaduje od zákazníka aby sa stal súčasťou tímu, špecifikácie sa neustále upravujú, podľa zákazníkových predstáv.
- *Priveľa zbytočných funkcii* – Systém obsahuje veľké množstvo funkcii, avšak ani jedna nie je v praxi zákazníkom využiteľná. Riešenie: Extrémne programovanie trvá na tom, aby sa riešili len úlohy s maximálnou prioritou.
- *Fluktuácia pracovníkov* – Projekt trvá dlhšiu dobu a programátorom sa na toľko zhnusí, že odídu inam. Riešenie: Extrémne programovanie žiada od programátorov, aby prijali zodpovednosť za odhadovanie termínov. Tieto odhady sa rešpektujú. Extrémne programovanie taktiež povzbudzuje ľudský kontakt medzi členmi tímu a znižuje osamotenie jednotlivých členov, ktorá často býva dôvodom nespokojnosti. Noví členovia tímu sú povzbudzovaní, na postupné prijímanie väčšej zodpovednosti a je im poskytnutá asistencia už skúsenejších členov tímu. Tým sa znižuje riziko, že bude programátor frustrovaný zadaním evidentne nemožnej úlohy, alebo že stratí záujem práce na projekte.

Podľa môjho názoru je extrémne programovanie odpoveďou na otázku „Ako by sme pracovali na projekte kebyže máme dostatok času?“ Určite si teraz poviete, ale veď v projekte mame pevné termíny, nemáme dostatok času. Ale pokiaľ by sme dostatok času mali, tak verím že by sa testovalo oveľa viac, takisto by sa viac diskutovalo medzi jednotlivými členmi v tíme, ale aj so samotným zákazníkom. A to sa práve extrémne programovanie snaží dosiahnuť. Pri jeho zavadzaní do tímu si je však nutné uvedomiť niektoré veci. Pri extrémnom programovaní nemáme možnosť rozhodnúť, či budeme vytvárať testy alebo nie. Pokiaľ to nebudeme robiť, nie sme extrémni. Koniec diskusie. Netvrdím však. že extrémne programovanie je vhodné pre všetky projekty a ani neuvádzam, konkrétne projekty pre ktoré je metodika extrémneho programovania najvhodnejšia. Určite však existujú jednoznačné prekážky, ktoré bránia v jeho aplikácii ako napríklad: veľké tímy, nedôveryhodní zákazníci a takisto projekty ktoré nepodporujú elegantné zmeny [1] [4].

Porovnanie a použiteľnosť jednotlivých metód

Každá z vyššie uvedených metód má svoje klady aj zápory. Pre náš tím sa mi javí metóda extrémneho programovania vhodnejšia, na koľko zavedenie manažmentu rizík vidím ako veľmi nereálne. Jedným z hlavných dôvodov je nízky počet členov v tíme a takisto nikto z tímu nemá skúseností s manažmentom rizík . Netvrdím, že by to

nebolo možné, no myslím si, že potom by sa hlavnou hrozbou projektu stal práve manažment rizík, nakoľko by mu bolo nutné venovať príliš veľa času, na úkor samotného projektu a ani zďaleka by sme neodhalili všetky hrozby. Extrémne programovanie predstavuje podľa mňa zdravý prístup k vývoju projektu, ktorý je prirodzený a dalo by sa povedať, že niektoré princípy máme osvojené a ani o tom nevieme. Študenti sú počas štúdia zvyknutí programovať v pároch a tak si pomáhať, to platí aj v našom tíme. Taktiež úzko spolupracujeme s vedúcou nášho tímu, čím je zabezpečená neustála spätná väzba. Takisto momentálne riešime len úlohy s najvyššou prioritou, a postupne za neustáleho testovania, či už našej vývojárskej strany, ale taktiež testovaním zo strany našej vedúcej, budeme pridávať ďalšie funkcie. Komunikácia je na veľmi dobrej úrovni a pri tak malom tíme, je veľmi malé riziko že niekto ostane „odstrčený“. Týmto spôsobom sa minimalizuje široká škála rizík, ktoré by mohli ohroziť náš tím a prácu na projekte. Nie všetky prístupy extrémneho programovania sú však pre náš tím vhodné. Určite vidím ako možný problém, kebyže v našom tíme absentuje akákoľvek dokumentácia a dokumentáciou by boli len samotné zdrojové texty. Spôsobilo by to značné problémy v komunikácii s našou vedúcou, ktorej by čisto zdrojové kódy nič nehovorili. Inak si myslím že zavedenie princípov extrémneho programovania, by sa stretlo v našom tíme s úspechom.

Záver

V tejto eseji som sa zaoberal problémom rizík ktoré ovplyvňujú vývoj softvérových projektov. V úvode je uvedená problematika rizika z historického hľadiska vývoja softvéru nasledovaná zoznamom najčastejšie vyskytujúcich sa rizík. Dokument poskytuje prehľad dvoch prístupov k týmto rizikám. Prvým je manažment rizík, ktorý sa snaží vopred rizika identifikovať, analyzovať, klasifikovať a navrhnúť akcie, ktoré je nutné vykonať v prípade ich výskytu. Sú tu popísané metódy a postupy ktoré manažment rizík využíva na dosiahnutie týchto cieľov. Ďalšia časť dokumentu je venovaná extrémnemu programovaniu a jeho prístupom k rizikám. Naproti manažmentu rizík, extrémne programovanie počíta s rizikom počas celého priebehu vývoja projektu a svojimi metódami sa ho snaží eliminovať. V závere dokumentu je porovnanie oboch metód z hľadiska vhodnosti, pre malý tím, ako je ten na predmete Tímový projekt. Samotný dokument však nedáva odpoveď na to, ktorá metodika je lepšia, ale len usmerňuje pri výbere tej správnej, kde metodika manažmentu rizík je vhodnejšia skôr pre väčšie tímy a metodika extrémneho programovania, pre menšie tímy.

Použitá literatúra

1. Beck, K.: Extrémní programování. Grada Publishing, Praha, 2002.
2. Bieliková, M.: *Softvérové inžinierstvo - Princípy a manažment*, Vydavateľstvo STU, Bratislava (2000), 141-146.

3. Boehm, B.W.: *Software Risk Management: Principles and Practices.* IEEE Software, January 1991, 32-41.
4. Grisham, P.S., Perry, D.E.: *Customer Relationship and Extreme Programming*. Empirical Software Engineering Lab (ESEL) ECE, The university of Texas, 1995
5. Keil, M., Cule, P.E., Lyytinen K., Schmidt, R.C.: *A framework for identifying software project risks*. Communications of the ACM, 76, Vol. 41, No.2 (1998)
6. The Standish Group: *The Standish group report*, 1995

Annotation

Risk planning and analysis in software project

New technologies, growing requirements, fast work rate. It's just a small group of risks that affect the work on a software project. Many times, underestimating the risks may have great influence on the further software project development process. So it is necessary, to be concerned about these risks, to identify them and try to select the best solution in case of their occurrence, or to prevent them completely, which will minimize the influence on the final product. The aim of this document is to provide the view on the most frequent risks, compare the methods of risk elimination from the view of the software engineering and extreme programming and also compare their applicability on specific software projects or teams.