PLÁN, NA ROZPOZNANIE ZLÉHO PLÁNU

Každý moment ľudského života je možné naplánovať.

Juraj Belanji

Slovenská technická univerzita
Fakulta informatiky a informačných technológií
Ilkovičova 3, 842 16 Bratislava
juraj[.]belani[zavináč]yahoo[.]com

Abstrakt. *Keď sa pred človeka postaví nejaký problém, prvá inštinktívna vec, ktorá ho napadne, je vytvoriť si plán na vyriešenie daného problému. Aj keď ten plán nie je oficiálne napísaný, každý človek si v mysli nejaký plán vytvorí. Na vytvorenie softvérového projektu je taktiež potrebný plán. Plán je základný kameň vytvorenia dobrého softvérového produktu. Tak ako je každý človek jedinečný, tak aj každý nový plán musí byť jedinečný. Je však možné vytvoriť plán, ktorý bude základnou kostrou na vytváranie nových plánov? V eseji sa identifikujú chyby, ktoré sa môžu stať pri tvorbe plánov. Esej ponúka iný pohľad na univerzálne plánovanie, a poukazuje, že občas univerzálne vytvorený plán zabezpečuje pozitívnejšie výsledky ako vytváranie nového plánu.*

Kľúčové slová: *plán, plánovanie, zlé plány, preanalyzovanie, univerzálne plánovanie, zlyhanie plánu, softvérový produkt*

Úvod

Kerzner [2] opisuje plánovanie ako proces výberu cieľa a ustanovenia zásad, postupov a programov, potrebných na dosiahnutie toho cieľa. Predstavme si softvérový projekt, ktorého cieľom je vyvinúť agenta, ktorý bude schopný zahrať si futbal v simulovanom prostredí RoboCup 3D. Po ustanovení cieľu je potrebné už postupne vytvárať samotný plán, na dosiahnutie tohto cieľu. Plánovanie projektu je možné opísať ako zavedenie vopred ustanoveného postupu práce. Pri plánovaní však narážame na veľké množstvo problémov, na ktoré väčšina projektových manažérov nie je pripravená. Najväčším

problémom, z vlastnej skúsenosti, je nevytvorenie dobrého a stabilného plánu, teda vytvorenie zlého plánu.

Ako však môžeme odhaliť zlý plán? Existuje spôsob ako sa vyhnúť zlému plánovaniu? Je plánovanie naozaj také dôležité? Sú univerzálne plány naozaj také zlé?

A bol plán, a bol prvý deň... a plán bol zlý!

Prvým krokom pri vytváraní plánu je uvedomenie si, že neexistuje dokonalý plán. Keďže teda neexistuje dokonalý plán, každý plán má chybu. Je teda možné povedať, že každý plán je zlý? To určite nie je tak. Musíme iba pochopiť, že aj tie dobré plány, vždy môžu byť aspoň o bodku lepšie.

Po uvedomení si, že nemôžeme vytvoriť dokonalý plán, najdôležitejším krokom, pri vytváraní toho nášho nedokonalého plánu je rozpoznanie chýb v plánovaní. Potrebujeme teda identifikovať zlé plánovacie zvyky. McConnell [3] opisuje 9 smrteľných „hriechov" plánovania projektu, ktorým by sme sa mali vyhýbať:

1. *Neplánovanie*
2. Nebrať do úvahy všetky projektové aktivity, teda *nedostatočné plánovanie*
3. *Neplánovanie rizík*
4. Použitie *rovnakého plánu pre každý projekt*
5. *Použitie cudzieho plánu*, bez vlastného kritického premýšľania
6. Dovolenie plánu *vzdialiť sa od projektovej reality*
7. Plánovanie dopodrobna príliš skoro, teda *preplánovanie*
8. *Plánovanie dobehnutia neskôr*
9. *Nepoučenie sa z minulých chýb*

Je vhodné sa pozastaviť a vysvetliť ako sa vyhnúť zlým zvykom v plánovaní.

Projekty najčastejšie zlyhajú práve nevytváraním žiadnych plánov. Riešenie je jednoduché: „Plánovať, plánovať a iba plánovať". Keď aj vytvoríme plán, musíme dbať aby sme dostatočne plánovali. Príkladom nedostatočného plánovania je povedzme neplánovanie odchodu jedného z členov tímu, čo je chyba na ktorú sme narazili aj v našom tíme. Toto je tesne späté s nepripravenosťou na riziká, ktoré boli identifikované. Neuvažovanie o rizikách, môže mať pre softvérový projekt katastrofické následky.

Na vyhýbanie sa takýmto chybám je vhodné povedzme priebežné vytváranie plánu. Ako príklad vytvorenia takéhoto plánu zoberiem plán ktorý sme priebežne tvorili na tímovom projekte. Najprv sa vytvorí hrubý návrh, teda malý plán, ktorý bude obsahovať všetky základné kroky vývoja. Ďalšou analýzou každého kroku sa určia riziká, ktoré sú späté s daným krokom. Následne sa priebežne plánuje iba krátkodobo (iba zopár krokov).

Musíme si aj uvedomiť, že každý projekt má svoje unikátne časti. Použitím rovnakého plánu na každý projekt znamená ignorovanie unikátnosti toho projektu. Aj napriek tomuto „hriechu", existuje množstvo kníh a metód, ktoré navrhujú generalizovaný plánovací proces, ako napríklad *Rational Unified Process*, *Extreem Programming* a pod. Nie je vhodné používať takéto plány, ale aj všeobecne cudzie plány, bez vlastného kritického pohľadu na ne.

Akokoľvek dobre vytvorený plán sa môže vplyvom problémov odtrhnúť od reality. Treba si uvedomiť, že plánovanie je kontinuálny proces. Nie je vhodné na začiatku samého

projektu vytvoriť plán na príliš nízkej úrovni abstrakcie (teda s veľa detailov). Treba plánovať priebežne.

Jedna z najčastejších chýb pri vytváraní plánov je práve „hriech" 8. Toto je problém s ktorým aj ja osobne prichádzam najviac do styku. Často nestíham urobiť nejakú časť projektu, a následne si naplánujem, že to dobehnem neskôr. Chyba je v tom, že si často myslíme, že ten stratený čas dobehneme v nasledovnej fáze. Podľa McConnella [3], zistilo sa, že projekty, ktoré sa nedodržiavajú termínov, málokedy úspešne dobehnú stratený čas neskôr. Je však zaujímavé, že ďalší „hriech", ktorého sa najčastejšie osobne dopúšťam je, že sa aj napriek tomu, že viem, že nedobehnem ten čas, vždy dopustím tej istej chyby, a naplánujem si, že to dobehnem neskôr, teda nepoučím sa na svojej chybe. Riešenie? Učme sa na vlastných chybách!

Načo tvoriť, keď je už vytvorené

„Hriech", pri ktorom by sme sa mohli trochu dlhšie pozastaviť, je „hriech" 4. Použitie rovnakého plánu pre každý projekt, môžeme inak nazvať aj univerzálne plánovanie. Ginsberg [1] definuje univerzálny plán ako ľubovoľnú funkciu z množiny možných situácií S do množiny primitívnych akcií A. Podľa Ginsberga [1] je univerzálne plánovanie vo väčšine prípadov zlý nápad.

Na vytvorenie univerzálneho plánu potrebujeme vedieť všetky možné akcie, riziká a spôsoby vytvorenia softvérového produktu. Pre nás nie je praktické vypočítať spôsob konania pre každú situáciu, v ktorej sa budeme možno nachádzať. Pre naše účely však môžeme univerzálne plánovanie vysvetliť ako vytvorenie jedného všeobecného plánu na riešenie viacerých *podobných* projektov. Je toto až také zlé?

Ako už bolo spomenuté, hriech 4. hovorí o tom, že nesmieme použiť rovnaký plán pre viaceré softvérové projekty. 5. zas o tom, že nesmieme použiť cudzí plán bez vlastného kritického názoru. Dajme tieto dva „hriechy" do súvisu. Môžeme povedať, že je možné použiť už vytvorený plán, ale potrebujeme ho pre každú situáciu (čiže projekt) prispôsobiť. Ako však rozpoznať, či pri plánovaní môžeme použiť aspoň z časti nejaký už hotový plán?

Vráťme sa k plánovaniu pre tím, ktorý vytvára agenta pre simulovaný robotický futbal RoboCup 3D. Každý rok sa pre vývoj tohto agenta zodpovedné dva tímy. Každoročne sa však tieto tímy menia. Na začiatku, si každý tím vytvorí svoj vlastný plán, podľa ktorého bude postupovať. Pre úspešné vytvorenie plánu, tím najprv musí analyzovať plán tímu, ktorý pracoval pred ním. Tuto prichádza k strate času. Podľa mojej mienky by bolo oveľa jednoduchšie a efektívnejšie, keby existoval nejaký všeobecný plán – nazvime ho *univerzálny plán* – podľa ktorého by bolo možné postupovať. Týmto by sa vývoj samotného agenta zrýchlil a skvalitnil.

Možno teraz však povedať, že je toto špeciálny prípad, pre ktorý by možno bolo aj vhodné mať nejaký preddefinovaný univerzálny plán. Iné softvérové projekty také však nie sú. Avšak podľa mojej mienky, Ginsberg nemá pravdu, keď povie, že sa nesmieme zaoberať vytváraním univerzálnych plánov [1]. Otázka nie je, či použiť univerzálny alebo už vytvorený plán, ale skôr, kedy ho použiť, a kedy nie.

Na určenie toho, či použiť nejaký už vytvorený plán, alebo nie, si musíme postaviť správne otázky. Prvá je, či ten projekt má súvislosti s naším projektom, ktorý práve

vytvárame. Nie je vhodné prebrať plán, ktorý sa vzťahoval na projekt, ktorý nemá žiadne body styku s projektom na ktorom teraz pracujeme. Ale pri podobných projektoch, podľa mojej mienky, nie len že je vhodné prebrať z časti plán, ale takmer určite ho treba prebrať.

Druhá otázka, ktorú si máme postaviť je, či je ten plán vhodný pre náš tím. Rettig a Simons [4] porovnávajú prácu malých a veľkých tímov, pričom tvrdia, že je plánovanie pre malý tím oveľa jednoduchšie a vhodnejšie ako pre veľký. Analogicky si musíme uvedomiť, že plán vytvorený pre malý tím nie je možné prebrať ak pracujeme vo veľkom tíme a opačne. Väčšinou sú takéto plány drasticky rozdielne.

Nesmieme zabudnúť, že aj keď je vhodné v niektorých prípadoch prevziať plán, alebo použiť univerzálny plán, ktorý bol pred tým vytvorený, ten plán nesmie byť použitý bez prispôsobenia sa k softvérovému projektu, na ktorom sa v danom momente pracuje.

Preto je podľa mojej mienky možno vhodnejšie preberať iba časti plánu a nie celé plány. Prebratím častí plánu projektu by sme tak dostali rámcový plán, nad ktorým by sme sa potom museli zamyslieť, a ďalej prispôsobiť k nášmu projektu a možnými rizikami, ktoré sú späté s ním.

Je to zlý plán, ktorý nepripúšťa žiadnu zmenu

Ako už bolo spomenuté, jedným z najčastejších dôvodov zlyhania softvérových projektov je aj nesledovanie plánov. Plán môže byť porušený na dva spôsoby. Buď nedodržíme vážne termíny, alebo prestaneme plánovať.

Je dôležité dodržiavať termíny. Nikdy nevieme, čo to môže mať za následky, ak sa termíny nedodržia. Aj na fakulte nás stále učia, že je dôležité dodržiavať termíny. Nedodržanie termínov väčšinou má za následok zmenu v pláne, ktorá predpokladá dobehnutie neskončených častí neskôr. To si ale dovoliť nesmieme. Z vlastnej skúsenosti môžem povedať, že som doteraz nikdy nedokončil neskôr to, čo som nedokončil na čas. Dôvodom je to, že som po tom, ako som si naplánoval, že to dobehnem neskôr, nezačal robiť produktívnejšie, ale skôr som si vždy povedal „Veď to aj tak už nestíham, takže mám času...“.

Druhým problémom, ako už bolo spomenuté, je neplánovanie priebežne. Z vlastnej skúsenosti viem, že aj keď si človek urobí plán, veľa vecí sa môže stať, ktoré mu ten plán porušia.

Ako príklad možno uviesť problém, ktorý nastál v našom tíme. Na začiatku vývoja agenta sme vytvorili plán pre náš šesťčlenný tím. Po mesiaci práce jeden člen tímu odišiel, takže sme zostali piati. Keby sme ďalej pokračovali po pôvodnom pláne s tým, že by sa úlohy odchádzajúceho člena tímu rozdelili na zvyšných členov, prišlo by k zlyhaniu projektu, lebo by zvyšný piati ľudia nestíhali dokončiť nielen tie úlohy, ktoré im boli pridané, ale ani tie svoje. My sme sa však pozastavili a plán upravili tak, aby neprišlo k zlyhaniu projektu.

Kerzner píše [2], že nie je vždy vhodné robiť zmeny v pláne. Meniť plány, ako už bolo poukázané, treba, ale tie zmeny nesmú byť veľké. Plán treba jemne upravovať. Myslím si, že ani nie je reálne počas práce na nejakom veľkom projekte očakávať, že veľká zmena v pláne bude pozitívne vplývať na projekt.

Záver

Identifikovali sme chyby a problémy, ktoré sú najčastejším dôvodom zlyhania softvérových projektov. Ukázali sme ako identifikovať zlé plány. Počas identifikácie problémov pri plánovaní a zlých plánov, sme ponúkli riešenie určitých problémov.

Ukázali sme, že je niekedy vhodné prebrať cudzí, alebo univerzálne vytvorený plán. Nie je však vhodné preberať takéto plány bez vlastného kritického pohľadu. Na záver sme ukázali, že je potrebné plánovať priebežne.

Tu je možné poviazať kontinuálne plánovanie a preberanie plánov. Keďže sme uzavreli, že je niekedy vhodné prebrať časť alebo aj celý plán, pričom ho treba zmeniť podľa vlastných požiadaviek projektu, tieto zmeny možno robiť aj postupne, počas vývoja projektu. Samozrejme, že je vhodné rámcový plán prispôsobiť hneď na začiatku, tak aby predstavoval základ projektu, ale keďže na začiatku vývoja softvérového projektu nevieme ešte presne určiť akým smerom ten projekt pôjde, je vhodné ho priebežne upravovať podľa informácií, ktoré priebežne budeme zisťovať.

Proces vytvorenia plánu pre softvérový projekt je zložitý a preto je potrebné nebrať túto tematiku zľahka. Prax ukázala, že je vždy vhodnejšie vytvoriť plán. Ten vytvorený plán, ako vidno z eseje, nielenže môže pomôcť človeku pri organizovaní času, ale aj výrazne môže zlepšiť kvalitu softvérového produktu.

Použitá literatúra

1. Ginsberg, M.L.: Universal Planning: An (Almost) Universally Bad Idea. *AI Magazine 10*, Vol. 4, (1989) 40-44.
2. Kerzner, H.: *Project Management: A Systems Approach to Planning, Scheduling and Controlling*. 10th edition. Wiley, 2009, ISBN: 987-0-470-27870-3
3. McConnell, S.: The nine deadly sins of project planning. In: *IEEE Software,* Vol. 18, No. 5, 2001, 5-7.
4. Rettig, M., Simons, G.: A project planning and development process for small teams. In: *ACM Communications Journal*, Vol. 36, No. 10, 1993, 45-55.

Annotation

A plan for recognizing a bad plan

When a man stands in front of a problem, the first instinctive thing that occurs to him is to create a plan to solve that problem. Although that plan is not formally written, each person creates a plan in his mind. For creating a software product, a plan is also needed. The plan is the cornerstone of creating a good software product. As each person is a unique individual, so must any new plan be unique. But is this really true? Is it possible to create a plan, which would be a blueprint for creating new plans? This essay identifies errors that occur while creating a plan. The essay offers a different view on universal planning, and shows that sometimes a universally created plan yields better results than a newly created one.