PLÁNOVANIE S VYUŽITÍM ZREŤAZENÝCH UDALOSTÍ

Najtiaž sa rozožrať, potom to už ide.

Juraj Jakabovič

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

Abstrakt. *Plánovanie by malo byť dôležitou súčasťou všetkých projektov väčšieho rozsahu. Pomáha lepšie odhadnúť množstvo zdrojov, potrebných na projekt, alebo rozvrhnúť čas, ktorý máme k dispozícii. V tejto eseji sa zameriavam na plánovanie v softvérovom projekte. Popíšem a porovnám niektoré spôsoby plánovania, ktoré sa v praxi používajú. Z uvedených metód bližšie opíšem metódu zreťazeného plánovania. Plány samy o sebe však nestačia a preto je rovnako dôležité vytvorený plán dodržiavať, alebo ho v prípade potreby vedieť doplniť, či upraviť. Popíšem teda aj spôsoby, ako plány udržiavať a sledovať. Na záver zhrniem význam uvedených metód pri tvorbe plánov v softvérových projektoch.*

Kľúčové slová: *zreťazené udalosti, premenlivé parametre, aktualizácia plánov, softvérový projekt, delenie úloh, plánovanie*

Úvod

Každá úloha väčšieho rozsahu si vyžaduje určitú prípravu na to, aby sme ju mohli úspešne splniť. Takáto príprava je vlastne plánom, v ktorom si premyslíme postupnosť krokov vedúcich k splneniu vytýčeného cieľa. Čím má úloha vyššiu dôležitosť, tým viac času treba venovať plánovaniu. Postup vytvárania plánu ako takého býva často veľmi podobný nezávisle na tom, o aký typ úlohy sa jedná. Existujú tu však určité rozdiely, preto sa ďalej zameriam na plánovaine pri tvorbe softvérových projektov.

Softvérový projekt je dielo, ktoré zhotovuje programátor, alebo firma pre svojho zákazníka. Môže sa jednať buď o dielo na objednávku od konkrétneho zákazníka, alebo o produkt, ktorý nie je vyvíjaný pre jedného zákazníka, ale má uspokojiť širšiu skupinu

klientov na trhu (tzv. generický softvér). V obidvoch prípadoch má však vývojárska firma určené zdroje, ktoré má pri vývoji konkrétneho softvéru k dispozícii a čas, za ktorý musí výsledný produkt dokončiť. Zdroje sú ľudské, teda zamestnanci, ktorí sú k dispozícii a finančné, teda celý rozpočet použitý v súvislosti s daným projektom. Optimálne využitie zdrojov je kľúčom k úspechu, pretože ich nedostatok sa väčšinou ťažko dopĺňa. Hoci vytváranie plánu nie je tvorivá činnosť, ktorá by sa priamo podieľala na tvorbe výsledného produktu – naopak, môže sa zdať, že ho brzdí – v skutočnosti je to práve naopak. Dôsledné naplánovanie nás často upozorní na rozdiely oproti pôvodným odhadom. Napríklad rozdlenie jednej úlohy na viac menších nám pomôže presnejšie odhadnúť prostriedky na vykonanie každej z nich a tak získame aj presnejší odhad o celej úlohe. Na základe toho potom môže vyjsť najavo, že daná úloha si vyžaduje viac prostriedkov, než sme na ňu pôvodne mali vyčlenených a prinúti nás to upraviť pôvodný plán tak, aby korešpondoval s novými zisteniami [1]. Výsledkom takéhoto zistenia potom môže byť hľadanie nových zdrojov, alebo zjednodušenie niektorých úloh tak, aby bolo možné celý projekt dokončiť s využitím času a prostriedkov, ktoré sú k dispozícii. Väčšina projektov začína s väčším množstvom požiadaviek, než je reálne možné splniť za čas, ktorý je k dispozícii a tak sa často uprostred vývoja jeho tvorcovia rozhodnú tieto požiadavky skresať. Takéto neplánované škrty môžu vyústiť až k znechuteniu klienta s pocitom jeho premárnenej snahy. Oveľa lepší prístup je určiť si k požiadavkám ich priority a podľa toho ich zaradiť do prioritnej fronty. Takto sa porarí zaručiť úspešné ukončenie projektu, pričom je skoro isté, že budú splnené všetky dôležité požiadavky, ktoré boli na projekt kladené. Menej dôležité požiadavky však aj napriek tomu majú šancu, že budú zapracované do výsledného projektu – ak zostane čas. Chýbajúca prioritizácia požiadaviek na projek je jedným z dôvodov oneskorenia väčšiny projektov, rozhodne však nie je jediná. Neschopnosť tradičných plánovacích metód vyrovnať sa s premenlivými parametrami sa prejaví na zlyhaní odhadov, na ktorých sú plány založené. Taktiež sa často nepríde na to, že vývoj nepostupuje lineárne, ale pred koncom sa spomaľuje. Väčšina projektových manažérov predpokladá, že úloha postupuje stále konštantým tempom a tak prehliadnu počiatočné príznaky oneskorenia úloh až kým nie je príliš neskoro a nezostáva nič iné, než odstrániť niektoré funkcie z projektu. Takýto postup je potom kompromisom medzi kvalitou a preplánovaním celého projektu.

Ako som spomínal, rozdelenie úloh na menšie podúlohy nám názornejšie ukáže, ktorá úloha si na svoju realizáciu vyžaduje aké zdroje - napríklad zamestnancov. Vyhneme sa tak prípadným kolíziam, keď by sa bez kvalitného plánovania neskôr napríklad zistilo, že jeden pracovník má súčasne pracovať na viacerých úlohách, následkom čoho by sa navýšil čas potrebný na dokončenie každej z jemu pridelených úloh, čo by negatívne zapôsobilo na celkový stav projektu. Môže to však byť aj zámer, keď sa pridelením viacerých úloh jednému zamestnancovi firma snaží ušetriť finančné zdroje. Tento postup je však značne rizikový a potom risk nie je ziskom, ale skôr stratou. Jednotliví programátori sa podľa pridelených úloh môžu rozdeliť do tímov, aby si v prípade problémov dokázali vzájomne pomôcť a zbytočne sa nezdržiavali sami na probléme, s ktorým nevedia pohnúť. V tomto je tiež plánovanie veľmi nápomocné. Úlohy spísané napríklad na veľkej tabuli pomáhajú sledovať, v akej fáze sa rozpracovaný projekt práve nachádza. Postupným odškrtávaním úloh a dopĺňaním nových potom vidno ako sa situácia mení a na čom treba aktuálne najviac pracovať. Zároveň úloha, ktorá je na tabuli

napísaná príliš dlho na seba upozorňuje, že je potrebné sa na ňu viac zamerať, prípadne zmeniť stratégiu pri jej riešení, aby neohrozila celkový výsledok projektu. To tiež vytvára nenápadný tlak na jednotlivých členov tímu, aby to neboli práve ich úlohy, ktoré zotrvávajú na tabuli najdlhšie.

Aj veľmi dobrý plán je však vždy len plánom, preto treba myslieť aj na nepredvídateľné situácie. Keďže s takýmito situáciami už z ich podstaty nevieme dopredu počítať, mali by sme do projektového plánu zahrnúť aj určité rezervy (či už časové, alebo finančné), vďaka ktorým by sme dokázali vzniknutú mimoriadnu situáciu vyriešiť s vynaložením menšieho úsilia, než keď sa na takúto možnosť nepripravíme vôbec.

Plánovanie sa netýka len programovaina, ale práce na celom projekte od vytvorenia samotného plánu, cez analýzu, vytvorenie softvéru až po testovanie, ktoré je tiež nevyhnutnou súčasťou vývoja softvérového diela. Firmy pracujúce bežne na veľkých projektoch musia plánovanie v čo najväčšej miere zefektívniť. Využívajú preto rôzne programy na podporu plánovania, ale tiež techniky, ktoré sa v priebehu rokov vyvinuli a používaním osvedčili.

Prehľad niektorých existujúcich metodík

Ako som už spomínal, plánovanie väčšinou začína rozdelením celého projektu na množínu nadväzujúcich nezávislých úloh, ktoré treba ohodnotiť čo sa týka prostriedkov vynaložených na ich vykonanie. Potom projektový manažér definuje priority jednotlivých úloh. Toto poradie sa potom dá zakresliť ako pospájaná sieť úloh, alebo Ganttov diagram. Čas potrebný na dokončenie projektu je určený najdlhšou cestou v tejto sieti. Táto cesta sa preto nazýva aj kritická. Na určenie tejto cesty môže projektový manažér použiť tzv. metódu kritickej cesty CPM (Critical Path Method), ktor je dostupná vo väčšine programov na podporu riadenia projektov [2].

Vo väčšine prípadov sú parametre ako začiatok a koniec projektu, jeho trvanie, celkové náklady a ďalšie dopredu nepredvídateľné. Preto bol vyvinutý model PERT (Program Evaluation and Review Technique), aby určil tieto nepresnosti v odhadoch. Podľa modelu PERT je očakávaná dĺžka úlohy vypočítaná ako vážený priemer medzi najoptimistickejším, najpesimistickejším a najpravdepodobnejším časovým odhadom. Očakávaná dĺžka ľubovolnej cesty sa potom dá vypočitať ako súčet očakávaných trvaní jednotlivých úloh. Hlavný problém s klasickým PERT-om je, že správne výsledky dáva iba vtedy, keď je v sieti úloh jedna hlavná cesta. Pokiaľ nie je jedna cesta dominantná, tento model obyčajne predpokladá optimistickejšie výsledky. Výhodou je, že väčšina softvérových projektov má jednu cestu dominantnú. Podobne by to malo byť aj pri našom tímovom projekte.

Na prekonanie týchto prekážok je možné ako alternatívu použiť simulácie Monte Carlo. Sú to procesy, ktoré opakovane nastavujú hodnoty pre každú premennú podľa rozdelenia každej premennej v štatistickom rozdelení. Týmito premennými môžu byť trvanie úlohy, náklady, začiatočný a konečný čas a pod. Hoci bolo dokázané, že simulácie Monte Carlo patria medzi efektívne metódy na analýzu projektového plánu, v praxi sa používajú len zriedkavo. Je to hlavne z dvoch dôvodov. Po prvé, väčšina softvérových systémov vyžaduje aspoň určité znalosti zo štatistiky a analýzy rizík na to, aby bolo možné vytvoriť vstupné dáta a následne interpretovať výsledky analýzy. Po druhe, simulácie

Monte Carlo nie sú vhodné pre vývoj softvéru, lebo nedodávajú presné odhady projektových parametrov. Je to spôsobené hlavne väčším počtom premenlivých parametrov (nástroje, zdroje, rozpočet, požiadavky, atď) v porovnaní s inými priemyselnými odvetviami.

Ďalší prístup k projektovému plánovaniu s nestálymi parametrami bol Goldrattov prístup. Aplikoval teóriu ohraničení (TOC – Theory of constraints) na projektové plánovanie. S využitím ohraničení sme sa mohli oboznámiť napríklad v predmete Metódy a prostriedky špecifikácie. Kľúčovým prvkom TOC je zdroj ohraničujúci kritickú cestu nazývanú kritickou reťazou. Goldrattov prístup je založený na deterministickej metóde kritickej cesty. Na určenie týchto nestálych parametrov Goldratt navrhuje použiť voľné zdroje projektu a odporúča skoré dokončenie úloh. Táto myšlienka sa mi zdá celkom logická, pretože nevyužité zdroje neprinášajú žiaden úžitok – dokonca často nás stoja nemalé prostriedky bez ohľadu na to, či sú využité, alebo nie. Taktiež skoršie dokončenie naplánovaných úloh je snom asi každého, kto mal niekedy na zadanú úlohu pevný termín. Pevný termín je taký, pri ktorom je čas odovzdania dôležitý rovnako ako odovzdanie samotné. V praxi sa to však darí málokedy. Príčiny môžu byť rôzne. Jeden z dôvodov, častý hlavne u študentov, je posunutie začiatku prác na úlohe až na dobu, kedy už nie je času nazvyš a za úspech sa považuje stihnutie dedlajnu. Rovnako však môže skončiť ten, kto síce začal hneď ako úlohu dostal, ale práce na nej si naplánoval tak, že využije celý čas, ktorý má k dispozícii a teda skončí až tesne pred termínom odovzdania. Tu sa vynára myšlienka, že cestou k splneniu úlohy pred zadaným termínom je naplánovať si ju tak, aby sme si jej ukončenie naplánovali na skorší čas, než je skutočný termín. Aj napriek tomu sa však môže stať, že sa to nepodarí. Podľa Paretovho princípu „80 : 20“ nám dokončenie posledných 20% projektu môže zabrať až 80% celkového času, ktorý mu venujeme. V takom prípade nepomôže ani vlastný posunutý termín odovzdania. Toto je však podľa mňa skôr výnimka a s použitím vhodných plánovacích metód sa jej dá vyhnúť, resp. ju obmedziť. Teória ohraničení je v projektovom plánovaní dobre prijímaná, aj keď použitie daného prístupu v odvetví softvérového vývoja je relatívne nízke.

Problémom vyššie spomenutých metód je v odhade vstupných premenných: trvanie úloh, začiatočné a konečné časy, náklady, zdroje, atď. Ak sú vstupné parametre odhadnuté nepresne, môže to viesť k nepresným výsledkom bez ohľadu na použitú metódu projektového plánovania. Riešením môže byť použitie metódy zreťazených udalostí.

Metóda zreťazených udalostí

Metóda zreťazených udalostí (Event Chains Methodology) bola navrhnutá na prekonanie problémov spojených s odhadom projektových parametrov, ako aj na zjednodušenie procesu softvérového plánovania. Podľa tradičnej metódy projektového riadenia je úloha (činnosť) plynulý rovnomerný proces. V skutočnosti je úloha ovplyvnená vonkajšími udalosťami. Tieto udalosti môžu zmeniť stav udalosti z jedného na druhý. Stav môže odkazovať na proces, alebo časť procesu s konštantnými vlastnosťami. Vo väčšine prípadov, hlavne pri výskumných a vývojových projektov, ako je vývoj softvéru, je ťažké predpovedať potenciálne udalosti vo fáze projektového plánovania. Nové udalosti sa môžu objaviť náhodne v priebehu inej udalosti. Jedna úloha môže byť ovplyvnená viacerými udalosťami naraz. Tieto udalosti sa pridajú na “zoznam udalostí”. Napríklad

počas vývoja nejakej softvérovej funkcionality sa zistí, že pôvodne uvažovaná architektúra softvéru je nevyhovujúca. Toto zistenie môže mať za následok vyškrtnutie tejto funkcionality z projektu, alebo dokonca zrušenie celého projektu. Taktiež to môže spôsobiť nárast nákladov a času potrebného na splnenie úlohy. Šanca na výskyt takejto udalosti na základe predchádzajúcich skúseností s programovaním podobných úloh je 20%. Na základe údajov z podobných predchádzajúcich projektov by sa mala objaviť v prvých dvoch týždňov vývoja [3].

Okrem pravdepodobnostných udalostí sú tu aj podmienené udalosti. Vyskytujú sa v niektorých prípadoch a závisia na projektových premenných. Napríklad ak úloha dosiahne svoj plánovaný termín na jej ukončenie, vygeneruje sa udalosť "zrušiť úlohu". Je tiež možné kombinovať pravdepodobnostné a podmienené udalosti – ak sa dosiahne plánovaný termín na ukončenie úlohy, je 20%-ná šanca, že úloha bude zrušená.

Udalosti môžu výrazne ovplyvňovať úlohy, skupiny úloh aj celý projekt. Úlohy v skupine môžu mať medzi sebou rôzne vzťahy. V skupine sa tiež môže nachádzať úloha so spoločným zdrojom, alebo iný známy parameter, na ktorý vplývajú tie isté úlohy. Je dôležité identifikovať skupinu úloh za účelom zjednodušenia procesu modelovania s udalosťami.

Jedna udalosť môže viesť k ďalším, alebo k vytvoreniu reťaze udalostí. Napríklad zmena architektúry softvéru si môže vyžadovať refaktoring softvérového komponentu. Výsledkom bude, že zdroj bude stiahnutý z inej úlohy, ktorá zmení stav – bude pozastavená. Takže jedna udalosť (zmena architektúry) môže spustiť reťazovú reakciu, prípadne môže viesť aj k veľkej zmene v pláne pre celý projekt. Rozdelenie úloh na rôzne kategórie, ktoré nám táto metóda ponúka je podľa mňa dobré preto, že sa vďaka tomu dá projekt lepšie popísať a teda aj presnejšie navrhnúť jeho plán.

Podstata je, že výpočty v metóde zreťazených udalostí sú obmenou simulácií Monte Carlo. Počas procesu simulácie sa počítajú vstupné premenné pre každú úlohu zvlášť, na základe vlastností udalosti. Výsledkom výpočtu je štatistické rozdelenie pre trvanie, začiatočný a konečný stav, mieru úspešnosti a náklady na celý projekt, alebo jeho ľubovoľnú samostatnú úlohu. Výsledok môže byť reprezentovaný vo forme početnosti, alebo celkového pravdepodobnostného počtu. Pre každú výstupnú premennú sa tiež vypočítajú štatistické parametre ako stredná hodnota, rozptyl, štandardná odchýlka, maximálna a minimálna hodnota.

Všetky metódy plánovania vyžadujú počiatočný odhad pre vstupné premenné projektu. Goldratt odporúča na odhad trvania úlohy použiť medián. Zo skúsenosti aj ja viem, že použitie mediánu na odhad nejakej veličiny je presnejšie, než keby sme brali do úvahy priemer, pretože medián menej ovplyvňujú extrémne hodnoty v štatistickom súbore. Simulácie Monte Carlo umožňujú projektovému manažérovi definovať si štatistické rozdelenie. Keďže metóda zreťazených udalostí je založená na simuláciách Monte Carlo, projektový manažér môže určiť štatistické rozdelenie pre projektové premenné. Napriek tomu sa to neodporúča, pretože ak je definovaná reťaz udalostí, môže to zapríčiniť dvojité počítanie tých istých nestálych premenných.

Jedna z najdôležitejších častí metódy zreťazených udalostí je sledovanie skutočného plnenia plánov a porovnávanie s pôvodným odhadom. Simulácia sa musí zopakovať vždy keď sa nové výsledky prejavia na projekte, alebo keď je známe plnenie niektorej úlohy. Keďže udalosti sú časovo závislé, nový výpočet nebude zahŕňať udalosti, ktoré sa mohli

odohrať skôr. Na základe toho sa vytvorí nová predpoveď, akým smerom sa projekt uberá. Toto je podľa mňa jedna z hlavných výhod tejto metódy. Odhady počítané z aktuálnych údajov sú vždy presnejšie, než staré odhady. Zároveň nám porovnanie s pôvodným odhadom umožní zistiť doterajšiu úspešnosť tejto metódy a vďaka tomu aj odhadnúť predpokladanú presnosť budúceho odhadu (počítaného z aktuálnych dát).

1. Popísaná metóda zreťazených udalostí môže byť aplikovaná na rôzne technologické procesy. Môže byť efektívna pri riadení softvérového projektu, kde dokáže výrazne zjednodušiť proces s viacerými nestálymi premennými.
2. Metóda zreťazených udalostí zahŕňa tieto hlavné princípy:
3. Vo väčšine skutočných projektov nie je úloha súvislý nemenný proces. Je ovplyvnená vonkajšími udalosťami, ktoré menia stav úlohy z jedného na druhý.
4. Udalosti môžu zapríčiniť vznik ďalších udalostí, ktoré vytvoria reťaz udalostí. Táto reťaz udalostí výrazne ovplyvní priebeh projektu.
5. Určenie kritickej reťaze udalostí dokáže zmierniť ich negatívny dopad.
6. Sledovanie postupu plnenia úlohy a súvislé porovnávanie skutočného postupu s odhadom zabezpečí, že projektový plán je počítaný na základe aktualizovaných informácií.
7. Analýza starších dát z podobných projektov súvisiacich s aktuálnym projektom je nevyhnutná na získanie informácií o pravdepodobnostiach vzniku niektorých udalostí a ich následkom.
8. Táto metóda zjednodušuje opakované plánovanie, analýzu projektu s cyklickými závislosťami a prideľovaní zdrojov.

Metóda zreťazených udalostí je praktický prístup na plánovanie softvérových projektov, ktoré obsahujú priveľa nestálych parametrov. Taktiež zjednodušuje opätovné vytváranie nových plánov počas vývoja projektu, takže plán zostáva aktuálny bez potreby vynaloženia veľkého úsilia na jeho udržiavanie. Vďaka tomu sa zbytočne nemíňajú prostriedky na aktualizáciu plánu a zároveň má projektový manažér prehľad, v akom stave sa projekt nachádza a môže adekvátne zasiahnuť, aby sa zdroje využívali podľa možnosti optimálne. Vďaka priebežnej aktualizácii plánu a jeho porovnávaniu so skutočným stavom projektu sú lepšie vidno prípadné odchýlky od pôvodného plánu a všetky situácie, ktoré by mohli viesť k neskorším problémom je tak možné vyriešiť ešte v zárodku, keď ich odstránenie ešte nie je také drahé.

Záver

Žiadna metóda nie je dokonalá a preto je normálne, že niekto uvedené postupy trochu vylepší na základe svojich skúseností. V takom prípade si ale treba dať pozor, aby sme sa od pôvodného zaužívaného postupu príliš nevzdialili, lebo takéto iniciatívy často prinášajú viac škody, ako úžitku.

Použitá literatúra

1. Kilger, Ch., Stadtler, H.: Supply Chain Management and Advanced Planning. *Springer-Verlag Berlin Heidelberg,* 2008.
2. Intaver Institute Inc.: *Software Project Scheduling under Uncertainties*. Calgary, AB, T2V0E5, Canada, 2007.
3. Miranda, E.: Combining Critical Chain Planning and Incremental Development in Software Projects. In: *PMI Global Congress Proceedings - Europe,* 2004.

Annotation

Planning with usage event chains

Planning is an essential part of every bigger projects. It is helpful to better estimation of resources used in project and make a time schedule. This essay describing planning in software projects. There are depict some common methodologies used in software planning. The event chain methodology will be described closer. Just plans are not enough. It is also important to know how to extend or modify existing plans. This techniques will be also depict. At the end I will sum up meaning of presented methodologies in software planning.