MONITOROVANIE V MALOM AGILNOM PROJEKTE

Každý projekt môže byť presne odhadnutý
(akonáhle je dokončený).

príslovie

Miroslav Hudák

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

Abstrakt. *Monitorovanie je dôležitou súčasťou manažmentu softvérového projektu. Cieľom tejto činnosti je včas identifikovať vzniknutý problém a podniknúť kroky k minimalizovaniu prípadných škôd. Častým dôvodom neúspechu projektu je zlý časový odhad a s tým spojené predraženie projektu. Preto je dôležité získavať relevantné informácie o stave projektu, aby sa včas dal aplikovať manažment zmien. V tejto eseji rozoberiem jednu z najpoužívanejších metód na monitorovanie projektu – analýzu dosiahnutej hodnoty, rozoberiem jej použitie pri agilnom vývoji softvéru a navrhnem jednoduchú metódu ako využiť túto techniku v malom projekte akým je študentský tímový projekt.*

Kľúčové slová: *monitorovanie, earned value analysis, agileEVM*

Úvod

Súčasný trend vo svete si vyžaduje primerané a konzistentné informácie o stave vývoja projektu. Softvérové projekty aj napriek tomu, že sú mnohokrát precízne naplánované, sa nevyvíjajú podľa plánu. Sú rozsiahle, komplexné s mnohými aktivitami. Neprimerané odhadovanie často vedie k neúspechu. Až 70% projektov presiahne plánovaný rozpočet alebo je pozadu za plánom. 52% projektov skončí na úrovni 189% ich plánovaného rozpočtu a niektoré po premrhaní veľkých investícií a času ani nikdy neskončia (zdroj: The Standish Group).

Je možné niečo s tým urobiť? Áno. V prvom rade je potrebné adekvátne naplánovanie a stanovenie rozpočtu a potom neustále a vhodné monitorovanie stavu projektu. Práve monitorovanie je tá činnosť, ktorá má krízový stav včas odhaliť a zohľadniť ho pri tvorbe ďalších plánov.

Pri monitorovaní sa sledujú niektoré atribúty alebo vlastnosti projektu a následne sa vyhodnocujú. Každý projekt má iné charakteristiky a vývoj, preto je potrebné položiť si otázku, akú metódu zvoliť aby sme boli schopní sledovať stav projektu a aby použitá metóda dávala správne výsledky.

V tejto eseji sa pokúsim priblížiť jednu z najpoužívanejších a najpresnejších metód monitorovania projektu – analýzu dosiahnutej hodnoty a jej modifikáciu pri agilnom vývoji a navrhnem možnosť jej uplatnenia v malom projekte.

Analýza dosiahnutej hodnoty

V súčastnosti sa na monitorovanie a vyhodnocovanie stavu projektu používajú viaceré metódy. Pri tradičnom vývoji je jednou z najčastejšie používaných a najefektívnejších techník analýza dosiahnutej hodnoty (Earned Value Analysis – EVA) [3].

Je to technika na meranie pokroku a efektívnosti postupu projektu v danom čase oproti plánu a pomocou toho odhadnúť budúci postup [1]. Pritom zohľadňuje tri dimenzie: plánované výdavky, skutočné výdavky a rozpočtové výdavky pre skutočne vykonanú prácu. Ale postačujú nám len prvé dve dimenzie, aby sme si mohli vytvoriť približný obraz o stave projektu.

Pri monitorovaní nás zaujímajú hlavne tieto otázky:

- ako sme na tom s plánom?
- ako sme na tom s rozpočtom?
- ako sme na tom s vykonanou prácou?

EVA nám dáva odpoveď na všetky tri otázky. Princíp tejto techniky spočíva v ohodnotení každého segmentu v projekte (úloha, softvérový modul, softvérová súčiastka) v nejakých jednotkách ceny, napr. cena práce, vynaloženého úsilia,... Z týchto cien sa následne definujú základné parametre [3]:

- *Planned Value (PV)* – plánovaná hodnota – rozpočtové náklady na naplánovanú prácu. Je to časť rozpočtu projektu vyhradená do daného časového okamihu.
- *Actual Costs (AC)* – skutočná hodnota za už vykonanú prácu.
- *Earned Value (EV)* – cena za vykonanú prácu v danom časovom okamihu percentuálne vyjadrená z rozpočtu.

Z týchto parametrov sa vyjadrujú základné indikátory stavu:

- *Cost Variance (CV)* ako rozdiel (EV - AC) a *Cost Performance Index (CPI)* ako podiel (EV / AC) dosiahnutej a aktuálnej hodnoty,
- *Schedule Variance (SV)* ako rozdiel (EV – PV) a *Schedule Performance Index (SPI)* ako podiel (EV / PV) dosiahnutej a plánovanej hodnoty.
- *Cost Schedule Index (CSI)* ako súčin CPI a SPI

Tieto indikátory nám dávajú nasledujúci význam:

- SPI < 1 znamená, že projekt je pozadu za plánom
- CPI < 1 znamená, že projekt presiahol rozpočet
- čím ďalej je CSI od 1,0, tým je projekt v horšom stave

EVA v agilnom vývoji

Technika EVA sa opiera o počiatočný základný plán založený na úlohách a projekt s dobre definovanou pôsobnosťou, ktorá sa vyvíja v sekvenčnej lineárnej forme [1]. Zmeny v rozsahu sa neočakávajú a ak, tak len v minimálnej miere.

Agilný projekt sa vyvíja v iteráciách, nelineárnej forme, s pridanou spätnou väzbou, ktorá ovplyvňuje počiatočný plán. Spätná väzba z každej iterácie ovplyvňuje ďalšiu. Zmeny pri takomto vývoji sa aj očakávajú a sú časté, preto merať stav projektu vo vzťahu k východiskovému plánu by bolo zavádzajúce. Rozsah agilného projektu je na začiatku veľmi ťažké zistiť, preto nasadenie EVA by znamenalo zle nastavenú plánovanú hodnotu (PV) a tým pádom aj zavádzajúce výsledky.

Čo to potom znamená? Je EVA v agilných projektoch irelevantná? Odpoveď je nie. Koncept techniky EVA nebol postavený len pre tradičný vývoj. V tejto oblasti sa viedol výskum [4], ktorý sa pokúsil adaptovať manažment dosiahnutej hodnoty pomocou parametrov definovaných v Scrume. Dokázal, že táto technika sa dá použiť aj v agilnom vývoji, ale je potrebné zvážiť vhodnosť ohodnotenia a použitie rôznych metrík. Výsledok výskumu je nazvaný *AgileEVM* (Agile Earned Value Management).

Použitie EVA v malom agilnom projekte

Keďže bolo preukázané [4], že techniku EVA je možné použiť v agilnom projekte a spomínaná práca ponúka aj riešenie, pozrime sa, ako by sa dala uplatniť v malom projekte ako je študentský tímový projekt. Študenti tento projekt riešia metodikou SCRUM, ktorá patrí k predstaviteľom agilného vývoja.

Základ je stanoviť si, ako sa budú jednotlivé segmenty ohodnocovať. Výskum [4] používa na ohodnocovanie body scenárov (story points). Avšak problémom je, že v takom malom projekte nie je veľa scenárov, ktoré by sa dali ohodnotiť. Tento predmet nemá za cieľ vytvoriť rozsiahly systém, ale naučiť študentov pracovať v tíme. Tiež je potrebné rozbiť scenáre na približne rovnakú úroveň granularity, čo nie je ľahké odhadnúť.

Ja sa pokúsim navrhnúť jednoduchšiu metódu. Jednoduchšiu preto, lebo študenti nemajú radi komplikované veci a tiež nemajú čas venovať veľkú pozornosť monitorovaniu. Čím teda ohodnocovať? Aká metrika je u študentov najobľúbenejšia? Z mojej skúsenosti môžem povedať, že počet riadkov kódu. Vždy, keď si študenti porovnávanú zadania, tak ich najviac zaujíma: koľko riadkov kódu? Ak je teda táto metóda taká obľúbená a študenti ju radi používajú, prečo ju neskúsiť aj pri monitorovaní. Táto metrika patrí aj medzi štandardne používané (LOC – Lines Of Code), ale nestretol som sa s jej použitím v technike EVA a už vôbec nie v AgileEVA.
Zadefinujme si potrebné pojmy:

- n – poradové číslo šprintu
- L – dĺžka šprintu
- $PLOC$ – plánovaný počet riadkov kódu definovaný ako súčet plánovaných počtov LOC po sledované vydanie
- $CLOC$ – dokončený počet riadkov kódu definovaný ako súčet všetkých LOC

Z týchto parametrov, modifikáciou vzorcov definovaných v [4], je možné určiť dátum sledovaného vydania (RD) ako posunutie od dátumu začiatku (SD) nasledovne:

$$RD = SD + L \cdot n \cdot \frac{PLOC}{CLOC} \tag{1}$$

Výhodou tejto metriky je jej jednoduchosť. Riadky kódu sa dajú ľahko spočítavať a je tu možnosť uplatniť automatizované meranie a sledovanie. Navyše ak má tím už nejaké skúsenosti s vývojom podobného softvérového projektu, môžu sa použiť historické dáta a skúsenosti na lepší a presnejší odhad potrebného počtu riadkov kódu. Tým pádom aj monitorovanie je rýchle a dostatočne presné.

Táto metrika sa zdá byť ideálna. Avšak je potrebné spomenúť aj jej obmedzenia. Najpodstatnejším obmedzením, ktoré je potrebné si uvedomiť, je špecifickosť pre rôzne jazyky. Na napísanie tej istej funkcionality je potrebný iný počet riadkov kódu v jazyku C ako napríklad v Perl či Python. Navyše aj v rámci jedného jazyka je možné použiť už vytvorené knižnice a tým si uľahčiť množstvo práce a problémov. Preto ak chceme porovnávať vyvíjaný projekt s už vytvoreným projektom, je potrebné si overiť, či bol napísaný v tom istom jazyku a či tam boli použité knižnice navyše.

Ďalšou nevýhodou tejto metódy je to, že ignoruje kvalitu kódu. Kvalitný programátor píše krátke a zložité konštrukcie a využíva množstvo predpripravených funkcií v danom prostredí, zatiaľ čo začiatočník nemá dostatočné znalosti a skúsenosti, preto je jeho kód zbytočne rozsiahlejší.

Ďalej je potrebné spomenúť nástroje, ktoré generujú kód, napríklad nástroje pre dizajn grafických komponentov v GUI. Generujú množstvo kódu a pri každej zmene GUI sa kód mení, čo je nevýhodné pri monitorovaní metódou LOC.

Nakoniec treba spomenúť aj nie zanedbateľný vplyv programovacieho štýlu, resp. predpísaných alebo odporúčaných štandardov písania.

Okrem spomenutej metriky počtu riadkov kódu možno pri monitorovaní v malom agilnom projekte používať aj iné vhodné metriky, napr. funkčné body alebo body prípadov použitia.

Metrika funkčných bodov vychádza zo špecifikácie funkcionálnych požiadaviek. Je určená primárne na odhad zložitosti softvéru pre interaktívne aplikácie. Funkčné body sa počítajú na základe transakcií v aplikácii. Pre neinteraktívne systémy (operačný systém, vnorené systémy) je vhodné použiť jej modifikáciu nazývanú *feature points*.

Medzi najväčšie výhody tejto metriky patrí nezávislosť od použitého programovacieho jazyka. Ďalšou výhodou je, že počet funkcií projektu je pomerne presne známy už na začiatku projektu. To je súčasne aj nevýhodou, pretože spoliehanie sa na prvotné špecifikácie projektu nemusí byť (a väčšinou ani nie je) v agilnom projekte spoľahlivé. Ďalšou nevýhodou je subjektívnosť pri určovaní charakteristík a ich váhovanie.

Keďže väčšina projektov v súčasnosti sa vyvíja objektovo-orientovane (aj tímový projekt), na lepšie odhadovanie pri monitorovaní by som navrhol metriku bodov prípadov použitia (Use Case Points). V tomto prípade je základom ohodnocovania počet a veľkosť prípadov použitia. Používa sa najmä v objektovo-orientovanom vývoji, kde je rozpracovaná biznis logika pomocou prípadov použitia.
Výpočet ohodnotenia pozostáva z nasledujúcich krokov [2]:

1. klasifikácia hráčov a priradenie váh podľa kategórií,
2. klasifikácia prípadov použitia a výpočet ich hodnôt pre neupravené váhy,
3. výpočet neupravených bodov prípadov použitia (UUCP) sčítaním predchádzajúcich dvoch výsledkov,
4. určenie faktoru technickej zložitosti (TCF) a faktorov prostredia (EF),
5. úprava hodnôt prípadov použitia vzťahom:

$$UCP = UUCP \cdot TCF \cdot EF \tag{2}$$

Váhy a faktory zložitosti pri výpočtoch je možné si zvoliť alebo navrhnúť podľa skúseností, ale existujú aj štandardizované hodnoty (napr. v [2]).

Podobne ako pri metrike LOC si môžeme zadefinovať plánovanú (*PUCP*) a dokončenú (*CUCP*) hodnotu bodov prípadov použitia vypočítaných vzťahom (2). Modifikáciou vzťahu (1) je možné určiť dátum sledovaného vydania (RD) ako posunutie od dátumu začiatku (SD) nasledovne:

$$RD = SD + L \cdot n \cdot \frac{PUCP}{CUCP} \tag{3}$$

Výhodou tejto metriky je možnosť automatizácie procesu výpočtu a dobrá viditeľnosť. Na druhej strane pri detailnom plánovaní je prípad použitia príliš veľká jednotka.

Záver

Monitorovanie patrí medzi kľúčové aspekty úspešnosti projektu. Preto by sa naň nemalo zabúdať. V tejto eseji som sa snažil priblížiť jednu z najpoužívanejších metód monitorovania – analýzu dosiahnutej hodnoty. Rozobral som jej použitie v agilnom vývoji softvéru a navrhol jednoduchý a časovo nenáročný, pre študentov priateľský spôsob ako ju použiť v malom projekte ako je tímový projekt. Prvou možnosťou je odhadovať pomocou metriky počtu riadkov kódu (LOC), avšak oveľa presnejšou metódou je merať pomocou bodov prípadov použitia (UCP). V oboch prípadoch som ukázal spôsob, ako je možné určiť dátum sledovaného vydania. Ostáva už len na jednotlivcoch, ktorú metódu si zvolia.

Použitá literatúra

1. Cabri, A.; Griffiths, M.; , "Earned value and agile reporting," *Agile Conference*, 2006 , vol., no., pp.6 pp.-22, 23-28 July 2006.

2. Jinhua Li; Zhibing Ma; Huanzhen Dong; , "Monitoring Software Projects with Earned Value Analysis and Use Case Point," *Computer and Information Science*, 2008. ICIS 08. *Seventh IEEE/ACIS International Conference* , vol., no., pp.475-480, 14-16 May 2008.
3. Lucas A. Joseph. 2008. Earned Value Analysis – Why it Doesn't Work. *AACE International Transactions.* EVM.01.
4. Sulaiman, T.; Barton, B.; Blackburn, T.; , "AgileEVM - earned value management in Scrum Projects," *Agile Conference*, 2006 , vol., no., pp.10 pp.-16, 23-28 July 2006.

Annotation

Monitoring in a small agile project

Monitoring is an important part of software project management. The aim of this activity is to identify the problem early and take steps to minimize possible damage. A frequent cause of project failure is a bad time estimate and associated project overcharge. Therefore, it is important to obtain relevant information about the status of the project in order to apply change management. In this essay I will discuss one of the most used methods for monitoring the project - Earned Value Analysis, its use in agile software development, and propose a simple method to use this technique in a small project such as a student team project.