SCRUM : PLÁNOVANIE, ZMENY A ROZDELENIE ÚLOH V SOFTVÉROVOM PROJEKTE

Mať dobrý plán dnes je lepšie ako perfektný zajtra.
[Anglické príslovie]

Samuel Snopko

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

Abstrakt. *Spôsob vývoja, ktorý si zvolíme v softvérovom projekte, hrá dôležitú úlohu v celom životnom cykle projektu. Scrum ako vývojová stratégia výrazne ovplyvňuje spôsob plánovania projektu, rozdeľovania úloh jednotlivým členom tímu a zapracovávanie zmien do celkového plánu projektu. Keďže samotný Scrum sa delí na behy, pre ktoré sa vytvárajú špecifické plány na vopred určené kratšie obdobie dvoch až štyroch týždňov, aj plánovanie sa musí prispôsobiť tejto stratégii. Úlohy sa rozdeľujú medzi členov tímu s prihliadnutím na predchádzajúce behy, pričom plán práve bežiaceho behu nie je možné meniť, len vynechať niektoré úlohy. Výsledkom je krátkodobé plánovanie a častá revízia plánu v momente skončenia jedného z behov. Vďaka tomuto novému prístupu k vývoju softvéru nám v oblasti plánovanie vyplávajú na povrch nasledovné otázky. Ako čo najlepšie prispôsobiť plánovanie Scrumu? Prináša Scrum viac výhod, či nevýhod pre plánovanie a vedenie softvérového projektu? Vieme pomocou Scrumu lepšie rozdeliť úlohy medzi členov tímu tak, aby tím pracoval, čo najefektívnejšie?*

Kľúčové slová: *Scrum, plánovanie projektu, softvérový projekt, rozdelenie úloh, šprint, beh, agilné plánovanie, odhad úloh*

Úvod

V dnešnej dobe už nie je jednoduché vytvoriť pre zákazníka program, ktorý požaduje. Tento fakt nenastal, pretože by boli programátori menej schopní ako v minulosti, ale hlavne s postupom technológií, ktoré na jednej strane zjednodušujú vytváranie programových kódov, no na druhej strane výrazne zvyšujú požiadavky zákazníka na vyvíjaný softvérový produkt.

Zákazník nepožaduje jednoduché a primitívne programy na správu údajov v jednoduchom grafickou dizajne, ale požaduje veľakrát internetové aplikácie s vysokou úrovňou bezpečnosti a ešte vyššou úrovňou grafickej príťažlivosti. Nie len z týchto dôvodov už dávno nie je schopný vyvíjať takéto aplikácie jeden človek, prípadne malá garážová firma.

Vznikajú vysoko špecializované tímy odborníkov na jednotlivé oblasti, ktorí často nepôsobia na jednom meste, či dokonca krajine alebo kontinente. Týmto spôsobom sa výrazne zvyšujú výdavky na vedenie softvérových projektov. Faktom však ostáva, že nie vždy je možné presunúť tímy na jedno miesto, alebo zamestnať odborníkov na viaceré oblasti.

Zle vytvorený plán softvérového projektu môže viesť nedodržaniu termínov, ktoré sú pri väčších softvérových projektoch zakotvené v príslušných zmluvách, alebo až k zlyhaniu daného projektu. Preto je dôležité nepodceňovať plánovanie a vždy mať v tíme niekoho, kto má skúsenosti s plánovaním projektov, ideálne softvérových.

Plánovanie v softvérových projektoch

Z pojmu „plánovanie“ nemusí byť hneď jasné, o čo vlastne ide. Preto je časté, že ľudia, ktorí neprišli v predchádzajúcej pracovnej kariére do styku s plánovaním na žiadnej úrovni, tak nemajú reálny prehľad, čo všetko tento pojem zahŕňa.

Definujú sa požiadavky na zdroje, požiadavky na prácu (činnosti) a definuje sa kvalita a kvantita práce (činnosti). Plánovanie je dôležité – projekt predstavuje vytvorenie niečoho jedinečného. Plánovanie by malo byť tak podrobné, ako je nevyhnutné – a nie tak, ako je možné [2].

Pri písaní tejto eseje som pracoval v rámci tímového projektu na fakulte Informatiky a informačných technológií STU, ako manažér plánovania jedného z tímov. Pri plnení svojej úlohy v tíme som získal veľmi dôležité praktické informácie, ktoré sa stali základnými piliermi tejto eseje. Opisy plánovania sa opierajú aj spôsob deľby práce vo firme IBM, kde som v tejto dobe pracoval. No napriek skúsenostiam z IBM sa mi nepodarilo na začiatku plánovania vyhnúť určitým základným chybám.

Ja som si na začiatku projektu nedal pozor na čas a neuvedomil si dôležitosť čo najskoršieho naplánovania úloh svojim kolegom. Toto bolo o to dôležitejšie, že sme postupovali vývojovou stratégiou SCRUM, kde je to priorita celého plánovania. Dôvody popisujem nižšie v tejto eseji. Samozrejme som podcenil rozsah svojich právomocí. No v skutočnosti som zistil, že manažér plánovania je pre vedenie tímu dôležitý minimálne ako projektový manažér a preto sa často tieto dve pozície spájajú do jednej a stáva sa tak z nej

plnohodnotná funkcia. Vykonávanie takejto funkcie zaberie všetok pracovný čas jedného z členov tímu.

Ako plánovať a na čo si dať pozor!

Pod plánovaním si treba predstaviť nie len vytvorenie dokumentu, v ktorom zadeklarujeme, ktoré časti projektu majú byť v akom časovom horizonte hotové, čo je vlastne dlhodobý plán projektu, ktorý sa vypracuje spolu s manažérom a zadávateľom projektu. Veľmi dôležité až kritické sú aj nasledovné povinnosti manažéra plánovania.

Deľba práce alebo každý má svoju zodpovednosť

Veľmi dôležitou povinnosťou manažéra plánovania je rozdeľovanie úloh medzi jednotlivých členov tímu. Pri veľkých projektoch, na ktorých pracuje niekoľko desiatok až stoviek pracovník však nie je možné, aby jeden projektový manažér dokázal jednotlivé úlohy rozdeľovať a preto rozdeľuje úlohy medzi menšie špecializované tímy. Tu už je možné úlohy rozdeľovať dvoma spôsobmi a to, že každý tím na najnižšej úrovni má vlastného manažéra plánovania, alebo sa vytvára zoznam dostupných úloh a z tohto zoznamu si jednotliví členovia lokálneho tímu tieto úlohy berú.

Odhady, veď na všetko treba čas

Pre šetrenie zdrojov v softvérových projektoch je veľmi dôležité, aby manažér plánovania mal aspoň základné skúsenosti alebo prehľad v oblasti, ktorú plánuje. Je totiž veľmi dôležité, aby každej úlohe určil čas, ktorý je na jej riešenie vyhradený. Toto je asi najcitlivejšia časť plánovania. Je jednoduché si predstaviť , že na úlohu sa vyčlení málo času a tým sa dostáva programátor do časového stresu, kde je náchylnejší na chybovosť. Ak nastane prípad, že od určitej úlohy sú závislé iné úlohy a táto úloha sa kvôli zlému časovému odhadu dostane do meškania, tak sa môže spustiť reťazová reakcia, ktorá môže v extrémnych prípadoch viesť až k oddialeniu termínu dodania finálneho produktu.

Opačným prípadom je, že sa čas na úlohu precení a nastane situácia, keď pracovník skončí danú úlohu výrazne skôr. Toto má vážny dopad na efektivitu daného pracovníka, či celého tímu, pretože napriek tomu, že pracovník vykazuje prácu na úlohe v skutočnosti je nevyužitý a teda sa plytvá zdrojmi určenými na projekt.

Zaužívaným štandardom je, že ak odhadujeme čas na určitú úlohu, tak predpokladaný čas ešte navŕšime o 20 až 30% tohto času, aby nenastávala situácia, kedy musíme posúvať ukončenie danej úlohy, čo by nám mohlo spustiť reťazovú reakciu.

Preto ešte raz zdôrazňujem, že je veľmi dôležité, aby mal manažér plánovania minimálne teoretické skúsenosti s oblasťou, ktorú plánuje. Samozrejme zlým odhadom sa nedá vyhnúť a v každom projekte sa nájdu, netreba sa ich však báť, ale naopak treba sa z nich poučiť. Platí, že čím skúsenejší je manažér plánovania, tým sú aj odhady presnejšie.

Plnenie plánu a jeho aktualizácia, pretože dokonalý plán neexistuje

Poslednou dôležitou povinnosťou manažéra plánovania je dohliadať na plnenie plánu, ktorý vytvoril a v prípade hocijakých komplikácií aktualizoval plán, prípadne úlohu, po diskusii s daným členom tímu. V malých tímoch je to jednoduché, ide len o to, aby

manažér plánovania priebežne dohliadal na pokrok v jednotlivých úlohách a v prípade situácie, že na určitej úlohe sa nepracuje alebo dlhšie nenastal žiadny pokrok, tak kontaktuje člena tímu, ktorý za túto úlohu zodpovedá. V prípade potreby je možné posunúť dátum ukončenia úlohy, ale všeobecne sa to neodporúča, pretože takéto zásahy do úloh skreslujú celkový pohľad na výkonnosť tímu a príslušné štatistiky, či výstupné grafy.

Výhodou malých tímov, kde sa jednotliví členovia navzájom poznajú je aj fakt, že komunikácia v rámci tímu prebieha v reálnom čase. Ak sa stane, že úloha meška a treba ju aktualizovať, tak je jednoduché kontaktovať osobu zodpovednú za danú úlohu, a následne dohodnuté zmeny zaviesť do celkového plánovania s prihliadnutím aj na ostatné úlohy. Problémy však nastávajú vo veľkých projektoch, na ktorých spolupracujú medzinárodné tímy.

V týchto prípadoch sú totiž plánované časy pre jednotlivé tímy, a pri zlom odhade a následnom nedodržaní plánu určitého tímu nie je možné ihneď kontaktovať globálneho manažéra plánovania, či konkrétneho pracovníka, na ktorom projekt prípadne úloha stojí. Dôvody problematickejšej komunikácie sú úplne jednoduché. V prvom rade je to iné časové pásmo, kedy na prípadnú odpoveď musíme čakať, čo nie je nezanedbateľné. Ďalej to môže byť jazyková bariéra, aj keď v tejto dobe globalizácie tento prípad nastáva len veľmi zriedkavo, pretože manažér by mal komunikovať so zodpovednou osobou z daného tímu a nie priamo s daným pracovníkom. Dôsledkom tohto je spomalená komunikácia, čiže nie je možné určité problémy riešiť tak rýchlo ako v malých tímoch a k aktualizácii dochádza neskôr.

Agilný, či plánom riadený vývoj?

Plánovanie môžeme v podstate rozdeliť na dve vývojové vetvy. Pôvodnou a staršou vetvou je tradičné plánovanie - plánom riadený vývoj. Tradičné plánom riadené prístupy zdôrazňujú predvídateľnosť a stabilitu v projekte [3]. Je to aplikovanie vyššie spomenutých postupov na dlhodobý plán, pričom je pevne stanovený postup v pláne a jednotlivé stretnutia k postupu plánu.

Agilný vývoj kladie menší dôraz na dlhodobé plány a prísne kontroly postupu stanoveného plánu. Viac sa zameriava na mechanizmy riadenia zmien, tvorbu krátkodobých plánov s možnosťou aktualizácie. V tomto postupe ide viac o ľudí a ich tvorivosť ako formalizované postupy dodržania plánu [3]. Ďalšími charakteristikami agilného vývoja je vodcovstvo, spolupráca, neformálna komunikácia a motivovanie tímov cez spoločenské akcie [4].

Na základe mojich skúseností si myslím, že oba typy plánovania majú svoje výhody a nevýhody. Pre plánovanie projektu je však veľmi dôležité si zvoliť správnu stratégiu vývoju. Neodporúčal by som riešiť všetky projekty bezhlavo jednou stratégiou plánovania a vývoju. Sú projekty, pre ktoré je lepší klasický vodopádový model plánovania a na druhej strane sú projekty, kde je lepšie použiť agilné metodiky. K tomuto tvrdenie som prišiel počas práce na tímovom projekte, kde som si vytvoril názor, že nám určená stratégia Scrum nie je tou správnou voľbou. Toto tvrdím aj napriek tomu, že Scrum ako stratégia vývoju a plánovania sa mi veľmi pozdáva.

Zastávam názor, že nikdy nie je neskoro zmeniť stratégiu plánovania, špeciálne v prípadoch, ak by to zlepšilo prácu v tíme. Jednou z možností ako plánovať, je aj kombinácia oboch vetiev plánovania. Toto kombinovanie je však možné len pri väčších projektoch, kde je možné spraviť celkový vývoj projektu vodopádovo, a jednotlivé tímy, ktoré pracujú na určitých menších častiach môžu postupovať agilne, napríklad Scrumom alebo inou agilnou metodológiou. Takýto postup môžeme nájsť vo viacerých veľkých firmách ako napríklad IBM.

Agilné plánovanie

Keďže momentálne poznáme pomerne veľké množstvo agilných metodík vývoja softvéru, je veľmi ťažké definovať základné aspekty agilného plánovania. Ako však hovoria autori agilného prístupu k vývoju a plánovaniu vo svojom manifeste – Jednotlivci a iterácie sú nad procesmi a nástrojmi. Funkčný softvér je viac ako komplexná dokumentácia. Spolupráca so zákazníkmi nad vytvorenou zmluvou a reakcia na zmeny je dôležitejšia ako striktné dodržanie plánu [1]. Z týchto bodov sa priamo plánovania týka tretí a štvrtý bod manifestu, ktoré sa snažia povedať, že aj napriek tomu, že vytvoríme nejaký dlhodobý plán stále je dôležité priebežne komunikovať so zákazníkom, prezentovať mu dosiahnutý postup na produkte a prípadne pripomienky zákazníka ihneď zapracovať do plánu. Totiž často krát sa stáva, že až keď vidí zákazník ako produkt pracuje, tak si uvedomí, že dačo chcel inak, čo výrazne ovplyvní vytvorený plán.

Čas na Scrum

V univerzitných projektoch, ktoré môžeme považovať za malé projekty, ktoré sa odohrávajú v priebehu jedného až dvoch semestrov, je plánovanie Scrumu trochu rozdielne ako v skutočných veľkých projektoch. Na nemeckej univerzite v Duisburgu použili plánovanie rozdelené na ABC-šprinty.

Vývojovú fázu rozdelili do troch iterácií – Alpha, Beta a Dokončujúceho (Completion) šprintu. Počas týchto šprintov mal každý tím pracovať samostatne a organizovať vlastnú činnosť. Z tohto dôvodu si musel každý tím veľmi precízne vybrať aktivity do nadchádzajúceho šprintu. V každom tíme pôsobil jeden Scrum majster, ktorý pravidelne dopĺňal backlog a podával pravidelné správy inštruktorovi, ktorý pôsobil ako zákazník. Na konci každého šprintu prezentoval tím dosiahnuté výsledky inštruktorovi. Šprinty boli časovo rozdelené nasledovne A-šprint a B-šprint trvali 4 týždne, zatiaľ čo dokončujúci šprint trval, len dva týždne a slúžil na ladenie a leštenie výsledného produktu [5].

V rámci projektu, ktorého som zúčastnil na univerzite, sme tiež postupovali Scrumom a plánovanie vyzeralo podobne ako v ABC-šprintoch. Podstatný rozdiel bol v dĺžke jednotlivých šprintov, ktoré boli pevne stanovené na dva týždne. Toto rozhodnutie nepovažujem za najlepšie a na vývojovú fázu šprintov by som odporúčal skôr štyri týždne ako to spravili v Duisburgu. Dva týždne pre jeden šprint je pomerne málo času pre univerzitné projekty, pretože týmto projektom sa nevenuje 40 hodín týždenne, ale maximálne 8 hodín týždenne na jedného člena. Tým pádom sa za jeden šprint nestíha spraviť žiadna väčšia časť aplikácie, ktorá by sa dala reálne prezentovať zákazníkovi. Musím spomenúť, že úlohy, na ktoré sa v normálnych projektoch vyčlení zopár hodín tu trvajú aj niekoľko dní a veľa času sa strávi komunikáciou v tíme.

Z nášho projektu by som vyzdvihol z pohľadu plánovania proces výberu úloh na nadchádzajúci šprint. Kedy sa prezrie celý backlog, čo je zoznam požiadaviek zákazníka, ktorý tvorí zákazník a môže ho dopĺňať počas projektu, následne pomocou kartičiek hodnotíme obtiažnosť jednotlivých požiadaviek. Na základe tohto hodnotenia vie manažér plánovanie lepšie odhadnúť, ktorá úloha bude koľko trvať, keďže má k jednotlivým úlohám vyjadrenia členov tímu.

Pre efektívne využitie každého šprintu je dobré, ak manažér plánovania, v tomto prípade Scrum majster, naplánuje do šprintu aj úlohy, s ktorými sa dopredu počíta, že sa nestihnú dokončiť v aktuálnom šprinte a presunú sa do ďalšieho. Myslím, že je to správny postup, lebo týmto spôsobom sa vyhneme situácii, že ukončíme plán šprintu pred jeho koncom. Keďže plány jednotlivých šprintov sú pevné a na začiatku každého šprintu vie každý člen tímu, čo bude nasledujúce dva až štyri týždne robiť.

Toto sa môže zdať mätúce s manifestom agilného prístupu, konkrétne s bodom - reakcia na zmeny je dôležitejšia ako striktné dodržanie plánu [1]. Tento postup však plne dodržuje tento manifest. Pretože ak počas šprintu dokončíme časť aplikácie, tak ju prezentujeme zákazníkovi. Ak zákazník zistí, že niečo by chcel zmeniť, tak v prípade malej zmeny to môžeme vykonať ešte v aktuálnom šprinte, pod pôvodnou úlohou. Ak sú to väčšie zmeny, tak tie sa napíšu do backlogu ako zákazníkova požiadavka a spracujú sa v niektorom z nasledujúcich šprintov, ktoré sa ešte len budú plánovať.

V konečnom dôsledku súhlasím s názorom, že Scrum pomohol úspešne definovať jednotlivé čiastkové ciele v šprintoch, čoho priamym výsledkom je funkčný prototyp so základnou funkcionalitou v začiatočných fázach projektu, ktorý je kľúčový pre komunikáciu so zákazníkom [5].

Všetko má svoje ale!

Aj agilné metodiky, v tomto prípade Scrum, majú svoje nevýhody. Ako už bolo spomenuté, tak sú projekty, ktoré sa lepšie riešia vodopádovým plánovaním, keď je naozaj výhodnejšie mať celkovú analýzu hotovú pred začatím implementácie. Je to napríklad tvorba webových aplikácií, kde je dobré si dopredu zadefinovať, čo bude ako vyzerať. Tým získa vývojový tím prehľad, čo sa dá viackrát využiť a môže písať univerzálnejšie zdrojové kódy.

V projektoch vedených Scrumom sa vo viacerých štúdiách do popredia dostáva problém stresu a vysokého pracovného zaťaženia. Scrum vytvára väčší tlak na programátorov, aby poskytli funkčnú funkcionalitu načas ako plánom riadená vývoj [3]. Pretože každý jeden šprint by mal končiť úspešným a hlavne finálnym odovzdaním časti produktu. Podľa komentárov a hodnotenia štatistík je pracovné zaťaženie pomerne vysoké, minimálne v univerzitných projektoch, čo vedie programátorov k tomu, že musia na projekt obetovať aj časť svojho vlastného času [5]. Táto štúdia však bola vytvorená pre herný priemysel, čo jemne skresluje pohľad na pracovné zaťažene. Nemyslím si, že pracovné zaťaženie je nezvládnuteľné, no určite je vyššie ako vo vývoji riadenom plánom, pretože sa pohybujeme v cykloch a každé dva až štyri týždne sa opakuje konečný termín šprintu, v ktorom musia byť úlohy hotové. Toto drží celý tím v napätí a v pracovnej atmosfére. Pričom v plánom riadnom vývoji sa pracovné zaťaženie zvyšuje len ku koncu projektu, avšak mnohokrát výraznejšie ako oproti tomuto čiastkovému v Scrume.

Záver

Plánovanie je veľmi dôležité pre efektívny a úspešný vývoj nielen softvérových produktov. Je intenzívne najmä v začiatočných etapách projektu, kedy sa vytvára plán projektu. Neskôr v priebehu vykonávania a riadenia projektu sa plány podľa potreby upravujú [2]. To je klasický plánom riadený vývoj.

Z vlastnej skúsenosti viem, že v agilnej metodike Scrum je plánovanie intenzívne vždy na začiatku šprintu, keď je potrebné vytvoriť plán daného šprintu. Toto plánovanie je veľmi dôležité, pretože ovplyvňuje celý priebeh prác na projekte.

V tejto eseji som sa snažil upozorniť na základné rozdiely medzi agilnou metodikou a vodopádovým prístup k vývoju a plánovaniu. Je dôležité si uvedomiť, že neexistuje dokonalý plán a preto treba správne a citlivo voliť prístup k vývoju a plánovaniu. Netreba sa báť zmien aj počas projektu a prejsť z jednej metodológie k druhej. Veľký potenciál vidím v hybridnom plánovaní, kde využívame oba momentálne zaužívané postupy plánovania.

Hlavne nikdy nezabudnite na dobrú komunikáciu s tímom, ktorá je zdravým základom správneho plánu. Častou komunikáciou so zákazníkom predchádzame opravám aplikácie pri ukončení plánu, keď sa na to musí vynaložiť viac úsilia ako v momente programovania danej funkcionality.

Použitá literatúra

1. Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, C.R., Mellor, S., Schwaber, K., Sutherland, J., Thomas, D.: *Manifesto for Agile Software Development*, 2001, http://agilemanifesto.org/
2. Bieliková, M.: Inicializácia a plánovanie softvérového projektu, 2009, http://www2.fiit.stuba.sk/~bielik/courses/msi-slov/prednasky/msi03-2009.pdf
3. Jingyue, L., Nils, B. M., Tore, D.: *Transition from a Plan-Driven Process to Scrum – A Longitudinal Case Study on Software Quality*, 2010, ACM, http://portal.acm.org/citation.cfm?id=1852786.1852804
4. Nerur, S., Mahapatra, R., Mangalaraj, G.: *Challenges of Migrating to Agile Methodologies. Communication of the ACM*, 2005, 72-78, http://doi.acm.org/10.1145/1060710.1060712
5. Schild, J., Walter, R., Masuch, M.: *ABC-Sprints: Adapting Scrum to Academic Game Development Courses*,2010, ACM, http://portal.acm.org/citation.cfm?id=1822348.1822373

Annotation

SCRUM: PLANNING, CHANGES AND TASKS DISTRIBUTION IN THE SOFTWARE PROJECT

The methods of development, which we choose in a software projects, play a important role in the project life cycle. Scrum as a development strategy significantly affects the way of the project planning, allocating task to he individual members of the team and updating changes into the

project plan. However Scrum itself is divided into the sprints, for which are created the more specific plans for a predetermined short period of two to four weeks, and also project planning have to adapt to this strategy. Tasks are divided between the team members with regard to the previous sprints. The plan of the running sprint can not be change, only some of the task can be unfinished at the end of the sprint. The result is a short-term planning and frequent review of the plan at the time of the end of one of the sprints. This new approach to software development asks following questions. How best could we adapt planning to the Scrum? Does Scrum bring more advantages or disadvantages into the planning and the project management software? Can we make better distribution of the tasks between the team members by using Scrum?