POVSTANIE ZABUDNUTÉHO SYNA?

Pomaly ďalej zájdeš.

Marek Šurek

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

Abstrakt. *Plánovanie patrí medzi hlavné súčasti vývoja každého softvérového projektu. Poznáme rôzne modely vývoja softvéru, z ktorých vyplývajú isté špecifiká pri plánovaní. Ja sa venujem hlavne spôsobu plánovania v metodike Scrum. Zaoberám sa najmä jeho vhodnosťou pri nasadení v menších či väčších softvérových aplikáciách. Najväčšiu váhu pritom prikladám praktickej použiteľnosti tohto vývojového modelu pri plánovaní projektov. Aby som pokryl kompletné spektrum, vychádzam z odborných a praktických skúseností ľudí z komerčného prostredia. Keďže mám tiež skúsenosti z firemného prostredia, časť tejto eseje som venoval mojim priamym skúsenostiam s plánovaním v stredne veľkej firme. Použitie Scrum ako východiskového modelu vývoja pri tímovom projekte pokladám ako ďalšiu vhodnú oblasť, na ktorú je dobré sa zamerať a diskutovať o nej v primeranom rozsahu.*

Kľúčové slová: *plánovanie, Scrum, používateľské body, nevýhody Scrum, vhodná aplikácia Scrum, časový odhad*

Začíname

Plánovanie patrí medzi pravidelné činnosti ľudí. Každý z nás si niečo v živote plánuje a očakáva, že sa mu jeho predstava vyplní. Niečo naplánovať je zložitá úloha nakoľko je veľmi náročné počítať so všetkými faktormi, ktoré môžu do realizácie plánu zasiahnuť. Tak ako pri väčšine problémov aj v oblasti plánovania existuje viacero prístupov ako minimalizovať riziká jeho nenaplnenia. Dnes sa často hovorí o vývoji tzv. metodikou Scrum. S odlišným prístupom k vývoju softvéru súvisí aj odlišný spôsob plánovania

jednotlivých úloh. Výsledkom by mal byť proces, ktorý zlepší a skvalitní aj samotný proces plánovania. Teoreticky.

Scrum ako taký sa prvý krát spomína už pred viac ako 25 rokmi. Keď si vezmeme toto dlhé časové obdobie a spôsob, akým sa IT sektor rýchlo vyvíja, je to až nepochopiteľné, že myšlienky tohto typu vývoja sa presadzujú až dnes. Môžeme si domýšľať mnohé príčiny tohto stavu. Vynára sa preto otázka, čo je ten skutočný dôvod?

Trocha teórie nikdy nezaškodí

Plánovanie má veľmi špecifické miesto pri tvorbe softvéru. Na jednej strane neposkytuje zákazníkovi žiadnu priamu pridanú hodnotu požadovaného produktu. Zároveň by sme boli ale bez plánu úplne stratení a vývoj produktu by bol jeden veľký chaos. Dobrý plán dokáže zákazníkovi, ale aj samotným vývojárom, poskytnúť prehľad o veciach, ktoré sú už hotové, a ktoré len na svoje vykonanie čakajú. Z dobrého plánu teda profitujú obe strany. Zákazník vie, kedy má očakávať nasadenie požadovanej funkcionality a vývojár vidí požadované vlastnosti z väčšieho kontextu, čo mu pomôže lepšie pripraviť vhodnú architektúru.

Skúsenosti z praxe naznačujú, že prvotné plánovanie by sa malo začať o niečo skôr ako masívny vývoj [1]. Dalo by sa povedať, že by malo hrať rolu už pri tvorbe štúdie vhodnosti. Pomocou takejto štúdie vieme spraviť prvotný odhad plánov, ktoré sú dôležité pre nášho klienta. Medzi hlavné parametre, samozrejme, patrí odhad ceny a času. Zvyčajne sa štúdia vhodnosti nerealizuje rovnako kvalitne ako finálna dokumentácia. Dôležité je, aby boli poskytnuté hrubé črty výsledného produktu, na základe ktorých vieme stavať.

Moderná doba si vyžaduje moderné prístupy, a preto sa v softvéroch čo raz viac začína skloňovať slovo paralelizácia. Princíp je predsa jednoduchý. Treba nájsť rozumnú mieru granularity nepriamo súvisiacich problémov a tie následne paralelizovať. Rovnakým spôsobom sa snaží riešiť problémy aj Scrum. Dôležite je si prioritizovať a preplánovať úlohy a v krátkych časových intervaloch (šprintoch) dodávať rýchlo nové verzie softvéru. Tento stav je možné brať aj ako postupnosť inštrukcií v prioritnom rade, ktoré bežia na viacjadrových procesoroch simultánne. Ak sa zmení špecifikácia niektorej úlohy, nastáva synchronizácia a preplánovanie. Následne môžeme pokračovať ďalej v upravenom pláne. Scrum vo svojej podstate nezavrhuje nutnosť tvorby akéhosi prvotného neagilného plánovania. V podstate uznáva, že toto plánovanie je nutné. Celý proces sa snaží čo najviac zefektívniť v nasledujúcich fázach použitím agilného spôsobu plánovania.

Z tohto pohľadu sa môže javiť, že plánovanie v Scrum spĺňa všetky atribúty moderného softvéru. Prax ale mnoho krát preukázala, že teoretické predpoklady sa môžu líšiť od praktickej realizácie.

Prax, ako je to s tebou?

Už spočiatku musím povedať, že svet nehodno nikdy vidieť čierno-bielo. Veci nebývajú čisto zlé alebo úplne dokonalé. Skôr môžeme hovoriť o väčšej či menšej vhodnosti pre danú oblasť. Bez akýchkoľvek emócií je preto niekedy vhodné vypočuť si názor iného

človeka, ktorý danú problematiku dobre pozná. Na takéto účely sú vhodné výskumy, ktoré reprezentujú profesionálny pohľad na vec bez subjektívnych pohľadov.

Jeden taký výskum sa zaoberá penetráciou vývojových modelov vo firmách [1]. Keď si vezmeme nepopierateľné teoretické prínosy a vek Scrum, mal by byť tento model používaný všade. Z článku sa ale dozvedáme, že to úplne nie je pravda. Zo 14 firiem, ktoré sa zapojili do výskumu, vývoj pomocou Scrum aplikuje presná polovica. Treba sa zamyslieť, či je možné vynášať súdy na základe vzorky 14 firiem. Nakoľko firmy boli vyberané náhodne, je možné každopádne usúdiť, že pre isté dôvody pre polovicu firiem nie je Scrum vhodný, resp. nepoužívajú ho z iného dôvodu. Tento fakt považujem za dôležitý indikátor, ktorý poukazuje na možnú neaplikovateľnosť/nevhodnosť plánovania a používania Scrum pre každú firmu či typ produktu.

Dobrým zdrojom informácií sú aj softvéroví veteráni, ktorí majú veľké skúsenosti s vývojom systému a vo svojej praxi sa stretli s viacerými spôsobmi plánovania. Aj u nás v školskom prostredí sa stretávame s učiteľmi, ktorí kedysi pôsobili v komerčnom sektore. Tí poukazujú na zaujímavý fakt, že v realite nemajú veľkí zákazníci alebo štát až taký veľký záujem na tom, aby každú chvíľu (šprint) hodnotili úplne minimálne zmeny v softvéri. Ak by sme si zobrali viacero paralelných šprintov tak situácia pre zákazníka by bola príliš zaťažujúca. Takéto tvrdenie je ale veľmi všeobecné. Daní učitelia pracovali prevažne pre veľké firmy, ktoré sú mnohokrát priveľmi zbyrokratizované a preto nemajú čas na to najdôležitejšie a tou je intenzívna starostlivosť o samotný vývoj produktu. Tieto firmy vykonávajú len pravidelné kontroly v ucelenejších verziách. Nezasahujú ale tak často (a zavčas) do vývoja samotného produktu ako to Scrum metodika očakáva. Túto informáciu musíme brať ako fakt, ktorý hovorí o tom, že Scrum pre veľké podniky nie je vždy žiaduci a niektoré firmy si viac zakladajú na tradičných spôsoboch plánovania.

Zákazníci nemajú potrebu stále niečo konzultovať. Veď prečo by aj, keď všetko podstatné si zakotvili v zmluve. A to je ďalší problém vývoja v Scrum – zazmluvnenie. Plán môžeme považovať ako jeden z hlavných komponentov zmluvy, kde je jasne definovane čo, kto a kedy. Ak používateľ stále mení požadovaný výsledok nejakého šprintu a mení požiadavky, ako nastaviť zmluvu tak aby pokryla náklady na tvorbu softvéru a súčasne sa dodržali stanovené termíny? Nazačiatku si jednoznačne treba stanoviť podmienky, čo sa má spraviť a čo na druhej strane nie. Sám som robil na menších projektoch, kde síce základné črty boli definované a na ich základe bola stanovená aj cena, ale zákazník stále vymýšľal a menil pôvodné črty až sa to nakoniec pre mňa stalo veľmi nerentabilné. Čo som ale mohol spraviť? Črty boli definované natoľko abstraktne, že som musel vyhovieť zákazníkovi a niekoľkokrát prerábať isté časti no zisk mi z toho už neplynul. Ak by bola nazačiatku vytvorená presná nemenná špecifikácia a k nej príslušný plán činností, na základe ktorej by bol vytvorený rozpočet, bolo by všetko tak ako má byť. Fixne stanovené veci sú ale presne to, proti čomu sa Scrum snaží v čo najväčšej miere bojovať. Aj z tohto pohľadu je preto nutné aby boli zmluvy písané novým spôsobom plánovania - veľmi častými dodatkami k zmluvám, čo nie je veľmi populárne.

Zdalo by sa, že Scrum by mohol mať na problém častých prezentácií pred zákazníkom adekvátne riešenie. Scrum predsa veľmi vágne definuje dĺžku šprintu. Tým sa stáva tento spôsob veľmi modulárny nakoľko Scrum šprint môže mať až jeden mesiac. Problém to ale nerieši a súčasne zhoršuje. Princípy agilného vývoja reprezentujú rýchle prinášanie nových funkčných verzií, za účelom včasného odhalenia problémov

a nedorozumení medzi zákazníkom a vývojovým tímom. Predlžovanie šprintu automaticky oslabuje nesporné výhody, ktoré agilné programovanie prináša, pretože nezrovnalosti sa odhalia neskôr a tento spôsob sa stáva viac tradičným iteratívnym modelom [2].

Ako som spomínal úvodom, nič nie je čierno-biele. Existujú aj spoločnosti, ktoré práve veľmi dbajú na časté pripomienkovanie k funkcionalite produktu. Ide hlavne o menšie spoločnosti, kde je osobná zainteresovanosť jednotlivých členov vyššia ako v obrovskom korporátnom molochu. Menšie celky sú lepšie organizované a preto aj plánovanie v metodike Scrum je zvládnuteľnejšie. Ďalším aspektom pri malých firmách sú aj financie. Kým veľké spoločnosti bývajú prevažne dostatočne bohaté, mladé začínajúce firmy sa musia veľmi obracať aby obstáli na trhu. Pre menšie spoločnosti môže byť nesplnenie požiadaviek doslova fatálne.

Napriek tomu, že spôsob agilného vývoja prináša svoje ovocie a dokáže výrazne zefektívniť činnosť jednotlivých procesov prináša väčšiu záťaž na samotných pracovníkov [5]. Každý pracovník je kontrolovaný denne na denných Scrum stretnutiach ako aj na šprintových stretnutiach. To vyvíja na jednotlivcov tlak, kedy musia neustále preukazovať nové a nové výsledky a súčasne vysokú mieru sebadisciplíny. Disciplína je to, čo dokáže zvýšiť šancu na úspešne splnený plán. Každý určite uzná, že disciplína a vysoký stupeň kontroly práce nie sú populárne nástroje medzi pracovníkmi. Keď si k tomu všetkému pripočítame niektoré vymoženosti ako napríklad práca z domu či flexibilita pracovnej doby, ktoré programátori získali v priebehu existencie svojej profesie, je jasné, že sa im ťažšie zvyká na disciplinovanejší spôsob práce ako Scrum vyžaduje.

Vlastná skúsenosť poukazuje na problémy

Najväčší problém spočíva v schopnosti relatívne správne odhadovať náročnosť úloh [5]. Niekto by mohol povedať, veď na odhady sú tu predsa Scrum kartičky. V dokonalom svete by to takto určite fungovalo. V tíme, ktorý je skladaný dynamicky je to veľmi problematické. Na ilustráciu poskytnem konkrétny prípad, s ktorým sa veľmi často stretávam vo firme, kde pracujem. Človek, ktorý robí s jedným nástrojom/problematikou najdlhšie má automaticky pridelenú úlohu. Ostatní členovia tímu nemajú ani právo, ani vedomosti mu do toho zasahovať nakoľko oni by čisto len „strieľali“ nejaké hodiny. Ak by to právo aj tak dostali, z kartičiek môže vyplynúť taký paradox, že ostatní členovia pridelia na danú úlohu vyššie časy ako ten, kto to bude v konečnom dôsledku robiť. Je prirodzenou ľudskou črtou, že on sa predsa nebude brániť. Veď prečo by mal? Takto môže pracovať v pokoji s tým, že má dostatočne veľkú rezervu. S odhadmi je to proste ťažké a ani Scrum nedokáže všetky jeho nástrahy eliminovať.

Ako je teda zrejmé, firma, v ktorej pracujem nepoužíva Scrum pri vývoji produktu a plánovaní. Hlavným problémom pri takomto plánovaní je distribuovanie úloh medzi jednotlivcov a stále distribuovanie úloh naprieč firmou. Takéto plánovanie teda vychádza skorej z definovania si úloh a následne dynamické vytváranie tímu, kde sa isté množstvo vývojárov neustále mení. Spájajú sa jednotlivé komponenty z iných aplikácií. Dáta z iných aplikácií a pod. Ako je možné vytvárať šprinty, keď nemáte pevne stanovený tím?

Časť argumentov, ktoré som použil v predchádzajúcich odsekoch by bolo možné aplikovať aj na iné vývojové modely. Čo ale považujem za úplne nepoužiteľný koncept

v Scrum sú používateľské body. Ich zámer je odbremeniť sa a neupínať sa na odhady hodín ako na presnú hodnotu a to poskytnutím istej úrovne abstrakcie. Čo nám ale takáto abstrakcia prináša? Z môjho pohľadu ide o zbytočné zahmlievanie vývoja produktu. Súhlasím s myšlienkou zbytočného upnutia sa na odhady a branie ich ako reálne. Použitím používateľských bodov dokážeme mierne zahmliť realitu, ale v konečnom dôsledku je to taktiež len odhad, ktorý nikomu neprináša väčšiu hodnotu. Ľudia prirodzene odhadujú na základe svojich skúseností z minulosti. Pri takomto porovnaní sa ale zameriavajú hlavne na hodiny, ktoré nad projektom strávili. Preto sa stáva, že používateľské body sú v konečnom dôsledku aj tak používané ako odhady hodín, aj keď sa tvária, že je to niečo iné, tajomné, trendové. Túto vlastnosť asi ako jedinú považujem za čiernu a teda nehodnú používania. Neviditeľnosť softvéru proste neoklameme sebe lepší mechanizmus a ja si myslím, že Scrum používateľské body sú skôr len na príťaž.

Tím náš školský

Scrum metodiku si máme možnosť reálne odskúšať na tímovom projekte. Z predchádzajúcich zistení sa domnievam, že pre náš menší tím by mohla byť Scrum metóda po miernych úpravách vhodná. Rovnaký názor na to má aj výskum na jednej zo škôl, kde takisto posudzovali uplatnenie Scrum v menších tímových projektoch [3].

Študenti prezentovali, že vývoj a plánovanie bolo na začiatku náročné, ale postupne si prijali túto metodiku za svoju. Miernu skepsu si ale stále ponechávam. Predsa len ide o vyjadrenia študentov, ktorý nemajú veľké skúsenosti ani s inými vývojovými modelmi. Nie je preto jasné či dokázali správne porovnať a vyhodnotiť tento typ plánovania.

Blížime sa k cieľu

Dobrých a racionálnych myšlienok pri plánovaní v Scrum je viacero. Hlavnou z nich je rozkladanie si problému na menšie časti s tým, že zákazník vidí produkt častejšie. Takýto proaktívny prístup k vývoju softvéru má určite predpoklady na zníženie rizika pri vývoji. Komerčné nasadenie vo veľkej firme poukazuje, že ani byť lepší neznamená byť plne aplikovateľný. Pri jasne stanovenom pláne sa dá jednoznačne definovať cena produktu. Ak sú zle špecifikované požiadavky resp. je možné ich meniť tak ako to Scrum dovoľuje ťažšie sa vyrokúvajú zmluvné podmienky. Nikto sa nemá na koho vyhovárať. Nazačiatku sa celý proces naplánuje a ktorá strana daný plán potrebuje zmeniť je automaticky vinníkom a je možné stanoviť aj sankcie.

Problémom pri väčších firmách sú samozrejme aj samotní ľudia. Tí nie sú do vývoja osobne zainteresovaní a preto sa stáva, že im ani nezáleží na výslednej kvalite produktu. O tomto probléme hovorí aj samotný manifest Scrum: „Pracujte na projekte s motivovanými jednotlivcami“[4]. Scrum prináša ľuďom nutnosť väčšej disciplíny a súčasne aj väčšej pracovnej záťaže nakoľko by jednotlivé etapy vývoja boli častejšie hodnotené, čo by im zaberalo viac času. To je tiež jeden z dôvodov, prečo sa vo veľkej miere Scrum nepresadzuje a nepoužíva vo všetkých spoločnostiach a typov projektov [1].

Každý model plánovania sa pri ideálnych podmienkach prejavuje výbornými výsledkami. Dôležité je nájsť model, ktorý podáva výborné výsledky aj v neideálnom svete. Tu je nutné povedať, že aj keď s mnohými vecami nesúhlasím alebo vidím v nich

problém, považujem Scrum za vývojový model, ktorý dokáže výrazne eliminovať možné problémy. Malé tímy a malé projekty majú veľké predpoklady profitovať z tohto modelu. Väčšie firmy s jasne stanovenými procesmi budú práve možno bojovať so štruktúrou, akú im tento model poskytuje. Tak ako aj autori článku [3] prezentujú, že plánovanie a celkovo Scrum je len otázkou zvyku, a keď doň človek prenikne, môže dosahovať lepšie výsledky.

Použitá literatúra

1. Greer, D., Conradi, R.: *Software project initiation and planning – an empirical study*. IET Software, Vol. 3, No. 5 (2009) 356-368.
2. Motoyoshi, Y., Otsuki, S.: An incremental project plan: introducing cleanroom method and object-oriented development method. *Software Engineering, 1998. Proceedings of the 1988 International Conference on,* Vol., No. (1998) 430-433.
3. K. Beck, M. Beedle, A. van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J. Highsmith, A. Hunt, R. Jeffries, J. Kern, B. Marick, R.C. Martin, S. Mellor, K. Schwaber, J. Sutherland, and D. Thomas, *Manifesto for Agile Software Development*, 2001.
4. (Overhage, S.; Schlauderer, S.; , 2012), "Investigating the Long-Term Acceptance of Agile Methodologies: An Empirical Study of Developer Perceptions in Scrum Projects," *System Science (HICSS), 2012 45th Hawaii International Conference on*, pp.5452-5461, 4-7 Jan. 2012
5. Werner, L., Arcamone, D., Ross,B.: Using Scrum in a quater-lenght undergraduate software engineering course. *Software Engineering, 1998. Proceedings of the 1988 International Conference on*, Vol. 27, No. 4(2012) 140-150.

Annotation

Rise of forgotten son?

Planning is one of the main components of development in any software product. We know different development models, which have its specifics. I'm interested in planning in Scrum methodology. My concerns belong to suitability of deployment in smaller and bigger software products. The highest importance is given to practical experience and usability in project planning. To cover complete area of Scrum, I based my opinion on practical and expert experience of people from business. I have also practical experience from middle size company which I'm going to share. Using Scrum in team project I consider as next area which deserves to discuss in adequate range.