KAMARÁTI DO DAŽĎA

Funguje to. Na sto percent. Jedine žeby nie.

Peter Dulačka

Slovenská technická univerzita
Fakulta informatiky a informačných technológií
Ilkovičova 3, 842 16 Bratislava
dulacka[at]gmail[dot]com

Abstrakt. *Vytvorenie tímu je náročná úloha, kedy je potrebné skombinovať nielen kvalitatívne, ale aj osobnostné charakteristiky jeho členov tak, aby sa navzájom doplňali. Fungovanie tímu pri použití metodológie Scrum je o to dôležitejšie, že prakticky nie je čas na zaváhania. V eseji diskutujem dôležitosť jednotlivca v tíme, problémy, ktoré jednotlivec môže spôsobiť a ako jednotlivec dokáže vplývať na kvalitu výsledku celého tímu. Opisujem, ako jednoduchý malý problém v sebe môže skrývať niekoľko väčších a skrytých problémov a snažím sa na základe vedomostí a skúseností navrhnúť, ako sa s takýmto problémom vysporiadať. Esej hodnotí ľudské charakteristiky a ich vplyv na členov tímu a najčastejšie problémy s charakteristikami spojené.*

Kľúčové slová: *manažment kvality, kvalita, SQA, softvér, zabezpečenie kvality, agile, scrum, komunikácia*

Nachádzate sa „TU“

Tvorba softvéru sa počas rokov stále upravuje v prospech požiadaviek trhu a neexistuje univerzálny spôsob vývoja, ktorý by sa dal označiť ako „najlepší“. Existujú ale metódy, ktoré sa k najlepšiemu spôsobu v konkrétnom čase približujú. Dnes sa presadzuje agilný vývoj a všetko sa orientuje na najčastejšie odovzdávanie funkcionality zákazníkovi. Kvalita sa zabezpečuje preventívnym určením rizík alebo spätnými metódami.

Nasledujúce strany sa pokúšajú odraziť autorov pohľad na agilný vývoj, pozíciu jednotlivcov v metodológií Scrum a metódy, ktoré by mohli na jednotlivca vplývať tak, aby kvalita softvéru bola na čo najvyššej úrovni. Keďže autor nemá jednoznačne vyhranený pohľad na to, čo jednotlivec pre tím znamená a čo na neho platí, môžu sa v texte miestami objaviť na oko schizofrenické tvrdenia. Ich cieľom je buď čitateľovi ponúknuť pohľad z oboch strán a nechať ho rozhodnúť sa alebo poukázať na občasnú

vysokú mieru nerozhodnosti autora. Názory na obe strany boli tvorené praktickými skúsenosťami vývoja softvéru v malom tíme ľudí.

Aj ja som „agilný“!

Hovorí sa, že práca v tíme dáva všetkým členom tímu veľkú porciu skúseností, priestoru na samostatnosť a možnosti sebarealizácie. Pri agilnom vývoji (obzvlášť pri metodike Scrum) by mala byť dôležitosť fungovania jednotlivca prinajmenšom rovnako veľká ako dôležitosť fungovania tímu ako celku. Prečo si myslím, že to tak je?

Pretože tím funguje z menších častí – jednotlivých ľudí a pri malom tíme sa každé zakopnutie môže prejaviť viacnásobne. Pretože moja skúsenosť ukazuje, že keď jednotlivci nepracujú spolu, efektivita vývoja klesá v dôsledku zhoršenej komunikácie (slabé množstvo komunikácie, komunikačné šumy a pod.) – v tomto názore ma podporujú aj kolegovia z Indie [2]. Pretože ak sú členovia tímu z iných končín sveta (stačí, keď je jeden z Bratislavy a druhý z „ďalekého vidieka“ – z Detvy), môžu vnímať veci inak. Pretože výmena človeka v tíme zmení fungovanie všetkých členov tímu – napr. príchod kolegu z odlišnej kultúry. A možno len nový kolega z Petržalky. Všetko predsa od niečoho závisí.

Existuje množstvo kníh, návodov, článkov a blogov (a iných užitočných zdrojov, ktorých množstvo a rozsah pokrývajú snáď všetko), ktoré hovoria ako z jednotlivca vyťažiť maximum. Čo si ale myslím je, že veľa ľudí si neuvedomuje ako dlho a ako postupne treba s človekom pracovať, aby sa dokázal zmeniť „v prospech projektu“ – a nakoniec; nezainteresovaný človek by mohol povedať, že celé to je o tom, ako sa podielnici snažia presvedčiť tvorcov projektu, aby pre dobro projektu (a v skutočnosti pre dobro podielnikovej výplaty) zahodili to, akí boli a ako fungovali a snažili sa zmeniť.

A prečo si zároveň myslím, že ten jednotlivec nakoniec nemusí byť až tak dôležitý? Aj v tíme (rovnako ako v akejkoľvek inej skupine ľudí – spolužiaci, spolucestujúci) pravdepodobne funguje istá psychológia, podľa ktorej sa ľudia môžu (možno vedome, častejšie však iba podvedome) správať a navzájom ovplyvňovať. Jednotlivec nemusí byť dôležitý, pretože navonok sa tím prezentuje ako jeden objekt – nenadáva sa na konkrétnych poslancov, nadáva sa na parlament ako celok – a práve tento objekt je terčom kritiky. Všetci členovia tímu sú z pohľadu zákazníka vinení rovnako, aj keď nemusia mať na probléme žiadnu účasť. A väčšinou sa členovia tímu aj tak prispôsobia jednému či dvom starším, skúsenejším, presvedčivejším členom (volajte to ako chcete, všetci to z jedného pohľadu určite poznáme). Pretože ako sa hovorí, každý je nahraditeľný. Pretože v praxi nakoniec vidím, že aj keď je člen tímu na druhej strane sveta (na ďalekom východe v Snine), pri správnom nastavení a dodržiavaní pravidiel vývoj funguje bez problémov.

Kameň, papier, nožnice

Ako sa ukázalo, nie som v otázke dôležitosti jednotlivca priklonený ani na jednu konkrétnu stranu. Tím by mal fungovať zaužívaným spôsobom. Tak, ako mu vyhovuje najviac. Aby niečo také vôbec bolo možné, musia existovať pravidlá. A aby mohli existovať pravidlá, mal by byť v tíme niekto, kto ich navrhne a zároveň by mali v tíme pracovať ľudia, ktorí sú dostatočne uvedomelí tieto pravidlá neskôr dodržiavať. Čo je ale zaujímavé, je to, že na základe vyššie spomenutého by malo platiť, že je úplne jedno, kto sa ujme roly

navrhovateľa. Môže to byť ľubovoľný člen (alebo členovia) tímu. Znamená to, že tím sa organizuje sám [3]? A ak sa organizuje sám, potrebuje mať niekoho, kto naň bude dohliadať? Ak je potrebné učiniť rozhodnutie, bude ten, kto ho učiní, aj zodpovedný?

Správne odpovedať na všetky tieto otázky pravdepodobne nie je ľahké (v niektorých prípadoch by pomohol iba hod mincou), ale stále nutné. Nesprávne, alebo v horšom prípade žiadne odpovede by totiž viedli k nespokojným členom tímu – a ak nie nespokojným tak určite nie motivovaným. Myslím, že to stačí na to, aby sa vývoj spomalil, vznikol priestor na chyby a aby sa to spolu odrazilo na kvalite dodávaného produktu.

Vezmime si jednoduchý prípad nespokojnej členky tímu (nazvime ju Lujza), kedy iný člen tímu (nazvime ho René) prebral jej zadanie kvôli Lujzinej neschopnosti dokončiť zadanie rýchlo a efektívne [1]. René zadanie spravil aj bez ďalšej konzultácie, avšak s dvoma chybami, ktoré Lujza zbadala ihneď pri analýze. Prípad, kde sa ukazuje, že každý je nahraditeľný, ale keďže len do určitej miery, zostáva stále dôležitý.

Príbeh jedného projektu

Ale pekne poporiadku. Bol raz jeden tím. V tom tíme boli schopní, talentovaní ľudia pracujúci na jednom rozsiahlom projekte. Do tímu prišla Lujza. Neskúsená, ale o to viac zanietená programátorka, ktorá chcela veľa vedieť. Dostala svoju prvú úlohu. Ako sa ale ukázalo, Lujza mala problém, ktorý nekomunikovala dostatočne skoro a snažila sa nájsť riešenie sama – moment, ktorý mal byť odbúraný každodennými rannými mítingami odporúčanými v metodológií Scrum. Ako sa ukazuje, stretnutia nemusia vôbec pomôcť, ak členovia tímu nedokážu prezentovať svoje problémy (prípadne dokážu, ale nie vhodným spôsobom). René, ktorý keď sa problém odhalil prebral Lujzinu úlohu, mal naopak pocit, že problémovú oblasť ovláda a Lujzu do riešenia nezahrnul. Nielen že jej tak znemožnil naučiť sa a pochopiť danú oblasť a spôsob ako problémy riešiť, ale z môjho pohľadu svojím konaním Lujze naznačil, že je nepotrebná, čo mohlo viesť k demotivácií.

Samotný fakt, že Lujza mala problém s vytvorením funkcionality do niečoho existujúceho vo mne evokuje, že projekt (a celkové nastavenie procesov) pravdepodobne neboli dostatočne prehľadné a že v tíme chýba (aspoň náhodná) kontrola kódu a zavedenie základných procesov – podľa môjho názoru kľúčová vec pre tím, kde sú členovia na rôznej úrovni skúseností. Číselné metódy vyhodnocovania kvality mohli ukazovať, že Reného kód bol „kvalitnejší“, čo mohol byť jeden z argumentov, prečo Reného rozhodnutie malo byť správne. Avšak byť na Lujzinom mieste, nastane znovu demotivácia – čísla naznačujú, že nerobí dobrú prácu a ešte ju musí po nej niekto preberať. Navyše Lujza o svojich problémoch rozprávala do vedeckého článku (čitateľ nech berie posledný argument s veľkou rezervou). Myslím, že ak má tím fungovať správne, mal by si vedieť problémové veci vyriešiť interne. Žiadna z týchto spomenutých vecí asi nepomáha tak kvalite produktu, ako aj atmosfére v tíme.

Na druhej strane si myslím, že v prípade neskúseného člena tímu by mohlo pomôcť párové programovanie, kedy sa zabezpečí vysoká kvalita kódu dozeraním skúseného programátora a zároveň „zaškolenie“ menej skúseného. Lujza aj vo vedeckom článku [1] spomína, že pri párovom programovaní mala pocit vyššej produktivity – avšak bez vyčíslenia ktoroukoľvek zo známych metód. Iba empiricky. A ak by aj nie, nestačí že z toho mala lepší pocit? Že ju to viac motivovalo?

Nie vždy musí párové programovanie pomôcť (ak si ľudia nesadnú, ak je problém komplexný). Vtedy by sa René mohol venovať implementácií a testy by mohol nechať na Lujzu. Písanie testov nie je až také náročné, ale pre ich správne napísanie je dôležité, aby Lujza pochopila fungovanie systému. Testovaním by sa zabezpečila ochrana projektu pred chybami, kód by bol kvalitný (od Reného) a Lujza by získala prehľad v konkrétnej časti projektu. Vyhrali by všetci.

Ponaučenie z jedného projektu

Menšie okienko do teórie. Kvalita je miera splnenia požiadaviek. Kvalita sa dá merať. Čiastočne. Tým koľko funkcionality sa dodalo a aj tým, koľko chýb (komentárov, tried, metód) kód obsahuje. Kvalita môže aj nemusí mať výpovednú hodnotu a záleží na použitých metrikách a interpretácií výsledkov. Ako teda René a Lujza a ich akcie vplývali na ostatných členov tímu a kvalitu projektu?

Lujzina uzavretosť a podcenenie zadania spôsobili, že sa problém neodhalil včas. Slabý záujem ostatných členov tímu o Lujzu - v zmysle novej členky tímu, samozrejme - zas spôsobil Lujzin problém a ešte viac prehĺbil jej uzavretosť. Reného tvrdohlavosť vyprodukovať kvalitný kód v čo najkratšom čase spôsobila ďalšie dve chyby a prehĺbila frustráciu Lujzy. Praktický modelový príklad toho, ako to nerobiť.

Pritom mohlo stačiť jediné: aby sa René (ako člen tímu, ktorý pozná danú časť projektu najlepšie) čas od času spýtal Lujzy, či nepotrebuje pomôcť. Zvýšila by sa morálka v tíme, Lujza by svoju úlohu dokončila, René by neurobil jeho dve chyby a ešte by stihol dokončiť aj svoje úlohy. Teda miera splnenia požiadaviek by bola omnoho vyššia a členovia tímu by boli spokojnejší. A nechýbalo veľa. Stačilo sa spýtať. Teda na kvalitu (v tomto prípade aj projektu, aj tímu a atmosféry v ňom) z môjho pohľadu negatívne vplývajú uzavretosť – introverzia, čiastočná ľahostajnosť a tvrdohlavosť. A na ich elimináciu by malo stačiť naordinovať trochu otvorenosti a transparentnosti.

Jeden za všetkých, všetci za všetkých

Kvalita kódu, rozdielne vedomosti, množstvo chýb, stabilita softvéru. Takto sa kolegovia v [2] rozhodli definovať faktory vplývajúce na celkovú kvalitu. Ako na ne vplývajú ľudia ako tieto faktory vplývajú na ľudí?

Kvalita kódu (ako sa ukázalo aj v predchádzajúcej časti) je výrazným faktorom hlavne pri nových členoch tímu alebo pri práci so starým kódom. Rôzne publikácie diktujúce „ako písať kvalitný kód“ (typu „*Čistý Kód*“), napriek tomu aké sú kvalitné, by z môjho pohľadu nemali byť brané doslova a čitateľ by si z nich mal zobrať len odkaz, ktorý chceli zanechať. Pri ich čítaní som narazil na celkom radikálne a miestami príliš reštriktívne myšlienky (komentáre sú zlo, trieda by nemala mať viac ako „n“ riadkov), ktoré sú z môjho pohľadu hlúposť. Ale keď sa nad tým človek naozaj dobre zamyslí, tak odkaz tejto myšlienky je písať tak dobrý kód, aby ten komentár nepotreboval. Myšlienka o počte riadkov podľa mňa hovorí iba o prehľadnosti kódu – a metriky, ktoré sa používajú slúžia len ako zrkadlo programátorovi, ako približne prehľadný ten kód je.

Na základe takto získaných vedomostí by tím mal mať stanovené štandardy, podľa ktorých sa pri programovaní riadi (názvy funkcií, komentovanie a i.). Tieto, ako sa

z mojich skúseností doteraz osvedčilo, musia prejsť spoločnou diskusiou, aby každý člen tímu bol aspoň do určitej miery spokojný s výsledkom a dospelo sa ku kompromisu. Ľudia musia byť schopní vyjadriť, čo chcú a musia byť schopní počúvať argumenty ostatných.

Rozdielne vedomosti členov tímu nemusia byť až tak závažným problémom. Starší členovia oprášia starý kód a zamrmlú popod fúz ako hrozne programovali pred niekoľkými mesiacmi, mladší zas majú možnosť prebrať od starších získané vedomosti. Myslím si, že predávanie vedomostí a zmazávanie rozdielov medzi členmi tímu (čítaj „generačných rozdielov“) sú kľúčové pre rast tímu a je dôležité aby nenastávala obdoba generačnej segregácie. Rast tímu - odborne aj ľudsky – prispieva aj k rastu projektu.

Množstvo chýb vplýva na kvalitu najväčšou váhou. Práve chyby sú totiž popri dodanej funkcionalite faktor, ktorý zákazník dokáže vidieť. Existuje množstvo spôsobov ako sa chybám vyvarovať. Osobne som za integračné testovanie (ľubovoľným spôsobom) a v prípade priestoru aj jednotkové testy. Tu sa presúva zodpovednosť za funkčnosť na jednotlivých členov tímu a oni by si mali uvedomiť, ako veľmi ich ľahostajnosť môže vo výsledku ovplyvniť ostatných. Taktiež sa mi osvedčilo, keď vynucovanie dodržiavania (ne)testovania je brané zábavnou formou, ktorá tím združí - pivo pre ostatných, ak niekto niečo kritické nedodrží – trest formou združovania. Hlavne tak, aby to niekto nevzal príliš osobne a nepokazila sa tímová atmosféra. Každý tím je iný a mal by si „to svoje“.

Stabilita softvéru je faktor, ktorý z môjho pohľadu na tím vplýva najviac. Frustrácia, ktorú môže nestabilný softvér spôsobiť a neustále hľadanie chýb a ladenie pravdepodobne nedodávajú tímu dobrú náladu. V tomto prípade by podľa mňa mohli pomôcť noví členovia v tíme, ktorí svojim entuziazmom a zanietenosťou (pravdepodobne spôsobenou skúšobnou lehotou) dokážu nakopnúť ostatných k náprave, prípadne iba pohľad nezávislej osoby, ktorá dokáže veľmi ľahko vidieť, čo v tíme funguje a čo nie.

„Stratení v kóde“

Čo sa ale stane, keď sa spoja ľudia, ktorých v princípe dopad ich práce na celkovú kvalitu nezaujíma resp. očakávajú, že sa v obrovskom dave stratí [4]? Táto časť je o tom, ako sa obrovské množstvo programátorov rôznej kvality spojilo pri otvorenom vývoji Linuxu a nedopadlo to dobre (čitateľ nech si všíma znepokojivú podobnosť naštrbenej kvality vo vývoji softvéru so súčasnosťou).

Začiatkom storočia bol v USA obrovský rozmach programátorov – programoval snáď každý, kto mal klávesnicu. Linux sa stával hitom a tak programátori dopĺňali stále nové veci. Avšak veľká časť z nich nerozumela čo robí a iba kopírovala a upravovala existujúce časti kódu. Vznikalo veľké množstvo pomalého kódu s priveľkým počtom závislostí na iné balíky, ktoré sa reálne ani nepoužívali. Len kvôli kopírovaniu. Programátori z pocitu malosti v dave a žiadnej kontroly nemali potrebu vytvárať niečo naozaj dobré a nejaká chybička ich moc netrápila. A Linux sa z toho chvíľu spamätával.

Aké si z toho treba zobrať ponaučenie? To by mal posúdiť každý po prečítaní knihy *„Generation Lost in Bazaar“*, ale z môjho pohľadu je podstatných niekoľko vecí, ktoré sa dajú uplatniť aj na menšie projekty. Z individuálneho hľadiska je nevyhnutné neskúsenému členovi tímu pomôcť pochopiť princíp vecí (ako fungujú, prečo práve takto a nie inak) a radiť mu pri návrhových rozhodnutiach. Zo všeobecného hľadiska musí existovať kontrola kvality. V ľubovoľnej forme. Pretože ľudia, keď nemusia, nebudú.

A keď nik nekontroluje chybovosť/čitateľnosť/návrh, môže sa ľahko stať že projekt bude nielen stagnovať, ale upadať. A na záver, ak to je možné, nezamestnať programátorov - kopírovačov. Dobehne to každého.

Vyskúšame, uvidíme

Z eseje teda vyplynulo niekoľko skutočností, ktoré sú mojim osobným názorom na zabezpečovanie kvality v projekte pomocou práce s členmi tímu.

Tím by podľa mňa mal fungovať ako celok, jednotne, avšak stále by mal byť veľký dôraz aj na kvalitu a spokojnosť jednotlivca. Tím by mal mať zavedené svoje pravidlá, ktoré fungujú – a ak nefungujú, mal by ich zmeniť, na každého vplývajú rovnaké veci inak. Členovia by sa sami mali chcieť chopiť zodpovednosti a napriek možnej demotivácií, mali by sa navzájom kontrolovať. Tím by sa mal snažiť vyvarovať uzavretosti, tvrdohlavosti a podceňovania problémov. Naopak, mal by sa snažiť čo najviac komunikovať a predávať si navzájom vedomosti – aby všetci členovia rástli. Ľudia by mali byť sami nastavení tak, že chcú dodávať produkt zákazníkovi a chcú ho dodávať kvalitne a kritiku ostatných nebrať negatívne, ale konštruktívne. A hlavne, mať stále potrebu zlepšovať sa.

Použitá literatúra

1. Berenson, S.B., Slaten, K.M., Williams, L., Ho, C.W.: Voices of women in a software engineering course: reflections on collaboration. *Journal on Educational Resources in Computing (JERIC)*, Article 3, 2004.
2. Bhasin, S.: Quality Assurance in Agile: A Study towards Achieving Excellence. *2012 Agile India* (2012) 64-67.
3. Cockburn, A.; Highsmith, J.: Agile software development, the people factor. *Computer*, vol.34, no.11, pp.131-133, Nov 2001
4. Kamp, P.H..: A Generation Lost in the Bazaar. *The Bikeshed* (2010).

Annotation

With a Little Help from my Friends

Team creating is challenging task when one needs to combine not just quality characteristics of a team member, but personal characteristics to supplement other team members as well. Operation of the team using Scrum is even more important because of practically no time for hesitance. In this essay I discuss importance of team member in a team, problems one can cause and impact of individual on quality of whole team work. I explain the case when a small problem can contain more complex and hidden problems and according to my experience I try to propose methods how to deal with such a problems. Essay judges one's characteristics and their influence on team members, work progress and final result, options of team members present when a problem occurs and most common problems connected with a person's characteristics.