UŽITOČNOSŤ MONITOROVANIA A AGILNÝ SPÔSOB VÝVOJA SOFTVÉRU

Odmerať neviditeľné je ako vidieť nejestvujúce.

Adrián Kollár

Slovenská technická univerzita
Fakulta informatiky a informačných technológií
Ilkovičova 3, 842 16 Bratislava
xkollara2[zavináč]is[.]stuba[.]sk

Abstrakt. *Na uľahčenie plánovania a zlepšenie monitorovania procesu vývoja existujú rôzne metriky, ktoré sa snažia do istej miery merať, či už produkt, alebo proces samotný. Softvérové metriky však trpia niektorými problémami, ktoré vyplývajú z povahy softvéru. V eseji bude objasnená podstata týchto problémov a zodpovedaná otázka prínosu metrík pri procese vývoja softvéru. Dôležitou témou je aj agilný spôsob vývoja softvéru, ktorý v dnešnej dobe pri vývoji softvéru používa čoraz viac spoločností. V súlade s agilnou metodikou sa pri rozhodovaní spoliehajú hlavne na odborníkov a intuíciu. V poslednej dobe sa objavili rôzne modifikácie používaných metrík pre potreby agilnej metodiky vývoja. Aby tieto metriky mali výpovednú hodnotu, museli byť upravené za účelom zohľadnenia agilnej metodiky vývoja. Môžu byť tieto metriky prínosom? Je užitočné ich používať?*

Kľúčové slová: *monitorovanie, metriky, projekt, agilný vývoj*

Úvod

Najdôležitejším cieľom každého projektu je jeho úspešné zakončenie, ideálne podľa predpokladaných nákladov a zdrojov. Nevyhnutným predpokladom k naplneniu tohto cieľa je samozrejme vytvorený plán projektu. Ak sa však situácia nevyvíja pozitívne, môže sa stať, že plán projektu sa neplní podľa predstáv manažérov a plán treba následne prehodnotiť. Avšak softvérové projekty majú oproti iným projektom svoje špecifické vlastnosti. Na zistenie skutočnosti, že projekt neplní predpokladaný plán, je treba s istou

presnosťou vedieť, v akom stave sa projekt nachádza. A tu sa dostávame k problému softvérových projektov. Cieľom každého softvérového projektu je nepochybne vytvorený funkčný a kvalitný softvér. Preto pri monitorovaní softvérových projektov je dôležitou časťou monitorovanie samotného softvéru. Avšak softvér trpí takou zvláštnou vlastnosťou, ktorá sa nazýva neviditeľnosť. Je teda možné softvér vôbec monitorovať? Aké sú možnosti monitorovania softvéru? Líšia sa od spôsobov monitorovania, ktoré sú používané v agilnom spôsobe vývoja?

Dôvody monitorovania

Ak by sme dokázali softvér a jeho zložitosť dokonale odhadnúť, monitorovanie by sa stalo nepodstatnou záležitosťou. Hneď pri zahájení projektu by sme presne vedeli, koľko ľudí budeme potrebovať na realizáciu projektu, čas potrebný na dokončenie a aj jeho rozpočet. Avšak nežijeme v ideálnom svete a naše odhady sú väčšinou veľmi nepresné. Plány, ktoré si vytvoríme, ostávajú často nenaplnené. A aby sme vôbec zistili, že plán ostáva nenaplnený, je potrebné vedieť v akom stave sa práve nachádzame. Z toho dôvodu vstupuje do hry monitorovanie. Monitorovanie projektu dokáže odpovedať na otázku, či sa projekt vyvíja podľa projektového plánu. V prípade, že sa projekt nevyvíja podľa plánu a manažéri by nezakročili s patričnými opatreniami, projekt pravdepodobne prekročí stanovený rozpočet alebo čas dokončenia, v extrémnych prípadoch môže byť aj zrušený. Z uvedených dôvodov je potrebné monitorovať stav a priebeh projektu a výsledky priebežne porovnávať s predpokladaným plánom.

Problémy metrík

Doterajší výskum v softvérovom inžinierstve priniesol viac ako 200 rôznych softvérových metrík, avšak miera využívania týchto metrík pri vývoji softvéru je na minimálnej úrovni. Väčšina používaných metrík je založená na starých metrikách ako cyklomatická zložitosť, počet riadkov zdrojového kódu, alebo funkčných bodoch. Všetky spomínané metriky pochádzajú ešte zo sedemdesiatych rokov minulého storočia. [4] Na druhej strane sú tieto metriky empiricky overené a sú oproti novším metrikám jednoduchšie a intuitívnejšie. Mnohým novým metrikám chýbajú spomínané vlastnosti alebo teoretický základ. To odrádza používateľov od zavádzania nových, často komplexných a nedostatočne preverených metrík.

Problémom jednotlivých metrík môže byť aj rôzna motivácia. Autori metrík sú často z akademickej pôdy a ich cieľom je kvalitnejší vývoj softvéru a lepšie riadenie softvérového procesu. V softvérovom priemysle je použitie metrík často motivované túžbou prejsť určitou certifikáciou.

Metódy monitorovania agilných projektov

Primárna metrika v agilnej metodike, ale aj v celom softvérovom inžinierstve je fungujúci program. Agilný spôsob vývoja softvéru sa tomu prispôsobil a snaží sa o čo najrýchlejšie dodanie fungujúcich prototypov programu. Stavia teda na rýchlosti a pružnom reagovaní na zmeny. Rýchlosť súvisí aj s rýchlym dodaním produktu zákazníkovi, ale aj s krátkymi

vývojovými cyklami, ktorých výsledkom je ďalšia iterácia produktu s pridanou funkcionalitou. Dôležitou metrikou pri agilnom spôsobe vývoja je teda samotná rýchlosť vytvárania jednotlivých iterácií produktu. Rýchlosť samotná ale nedokáže poskytnúť odhad o stave riešenia projektu a očakávaného času dokončenia. Aby bola rýchlosť plnohodnotným ukazovateľom, jednotlivé iterácie by museli byť rovnocenné z hľadiska vykonanej práce a bolo by potrebné poznať, koľko práce ešte ostáva. Preto sa v agilnom spôsobe vývoja odhaduje zložitosť ostávajúcej práce. Táto odhadnutá zložitosť ostávajúcej práce sa spolu so skutočne vykonanou prácou zobrazuje v grafe nazývanom burn-down chart [2].

Avšak dokáže burn-down chart odhaliť odchýlenie od plánu dostatočne skoro a s jednoznačným výsledkom? Podľa môjho názoru je burn-down chart dobrým prostriedkom na určovanie približného hrubého odhadu nedokončenej práce v aktuálnom šprinte. Avšak keď sa krivka ostávajúcej práce začne odchyľovať od krivky ideálnej hodnoty, nedozvieme sa, či ide len o lokálne vychýlenie, alebo je to predzvesť niečoho horšieho. Je možné, že krivka sa čoskoro opäť priblíži plánu. Avšak, čo ak sa tak nestane? Ovplyvní to len aktuálny šprint? Alebo je to už signál, že projekt nestihne byť dokončený v naplánovaný čas? Rozhodne je to signál, ktorý hovorí, že projekt sa neuberá správnym smerom. Avšak zo svojej podstaty, burn-down chart stojí najmä na odhadoch a tie, aj keď sa ich snažíme vytvoriť čo najpresnejšie, sa často od reality veľmi odlišujú. Z toho dôvodu je veľmi problematické vyvodzovať nejaké závery na základe prostriedku, ktorý stojí a padá na odhadoch. Ale nie je možné ho brať ani na ľahkú váhu. Vhodným riešením by mohla byť kombinácia viacerých kritérií. Väčšinou sa však rôzne metriky venujú aj odlišným stránkam samotného problému. V prípade, že existuje viacero metrík na meranie alebo predpovedanie toho istého typu problému, väčšinou stoja na tých istých predpokladoch alebo ich závery nemajú pridanú výpovednú hodnotu. Asi aj z toho dôvodu, pri skúmaní nových metrík pre agilné prostredie vývoja, sa autori [2] a [3] zamerali hlavne na metriky z nových oblastí monitorovania softvérových projektov. Napríklad, ako najdôležitejšiu metriku uvádzajú v [2], viditeľnosť biznis hodnoty. Ako som spomínal, je to metrika, ktorá sa venuje inej oblasti, ako bežne používané metriky. Osobne sa mi práve táto metrika pre agilné projekty nezdá až tak dôležitá, keďže projekty sa vyvíjajú hlavne na mieru zákazníka, ktorý si sám určuje požiadavky, ktoré sú pre neho dôležité. Táto skutočnosť môže vypovedať aj o tom, že používatelia súčasných metrík sú s nimi do istej miery spokojní a neočakávajú objav nejakej novej prevratnej metriky.

Užitočnosť metrík

Dôležitou otázkou je užitočnosť metrík samotných. Prekvapil ma obrovský počet rôznych metrík, na ktoré som narazil pri hľadaní zdrojov k tejto eseji. Jedná sa napríklad o metriky KPI, Earned Value Indicator, Bayesian, LOC, cyklomatická zložitosť, Haelsteadova zložitosť a rôzne iné. [4] K zaujímavému pozorovaniu prispel aj Oram [1], ktorý sa venoval potrebe ďalších metrík. Pri skúmaní metrík na odhadovanie zložitosti zdrojových kódov dospel k záveru, že všetky rôznorodé metriky ktoré skúmal korelujú s metrikou známou ako LOC, teda počet riadkov zdrojového kódu. Venoval sa najmä metrikám na posúdenie zložitosti zdrojového kódu. Toto zistenie korešponduje s prechádzajúcou hypotézou, že

metriky zamerané na rovnakú problémovú oblasť monitorovania dávajú približne rovnaké výsledky a neprinášajú teda dodatočnú pridanú hodnotu.

Očakávaný výsledok pri používaní softvérových metrík je zlepšenie kvality projektu, udržiavateľnosti, alebo zrýchlenie celkového vývoja softvéru, čo znamená aj výslednú nižšiu cenu softvéru ako takého. Kvalitný softvérový produkt by rovnako mal byť ľahko porozumiteľný, modifikovateľný a jednoducho rozšíriteľný. V perfektnom svete by nám k tomuto požadovanému stavu mali dopomôcť softvérové metriky. Po odmeraní softvéru vhodnou metrikou by sme dostali presné číslo, na základe ktorého by sme vedeli vyhodnotiť aktuálny stav. Po prekročení určitej hranice by sme následne strávili určitý čas na vylepšenie návrhu softvéru. Keby takáto metrika existovala, drasticky by sa tým zredukovali nároky na údržbu softvéru. Napriek tomu, že takáto metrika neexistuje, dokážu metriky poslúžiť ako užitočné ukazovatele zložitosti softvéru a ako také sú cenným nástrojom v rámci vývoja softvéru a pri procese refaktorovania.

Ich nedostatky vyplývajú z toho, že sa musia používať spätne pri určovaní kvality softvéru. Najprv je potrebné zdrojový kód vytvoriť a až následne nám vedia tieto metriky poskytnúť informáciu o zložitosti tohto kódu. To je užitočné, najmä ak ste manažérom veľkého tímu, ktorý dokáže na základe metrík posúdiť kvalitu kódu, prichádzajúceho od mnohých vývojárov. Metriky sú menej užitočné, keď potrebujeme zabrániť vzniku nadmernej zložitosti pri navrhovaní systému od základov. Je možné zníženie zložitosti spätne pomocou refaktorovania, ale v konečnom dôsledku to bude drahšie, ako keby sme danej zložitosti vedeli predísť pomocou preventívnych opatrení.

K dnešnému dňu existuje viac ako 200 zdokumentovaných softvérových metrík určených na meranie a hodnotenie rôznych aspektov softvérového systému. V [4] sa uvádza, že dôvod pre meranie softvéru metrikami je jeden z nasledujúcich:

1. Posúdenie alebo predpovedanie kvality softvérových produktov.
2. Posúdenie alebo predpovedanie zdrojov a nákladov na vývoj.

Softvérová kvalita je ovplyvnená množstvom premenných a jej posúdenie nemôže byť vykonané pomocou jednej metriky. So softvérovou kvalitou nepochybne súvisí zložitosť samotného systému. Čím je softvér zložitejší, tým je ťažšie testovateľný, rozšíriteľný aj udržiavateľný. Zložitosť zdrojového kódu zase ovplyvňuje jeho zrozumiteľnosť a je kľúčovým atribútom pri vývoji softvéru kvôli iteratívnej povahe vývoja. Proces vývoja softvéru je cyklický, pričom sa vo viacerých iteráciách striedajú fázy údržby a rozšírenia zdrojového kódu. Existuje teda priamo úmerný vzťah medzi zrozumiteľnosťou kódu a nákladmi týchto cyklov.

Existuje viacero atribútov, ktoré prispievajú k zložitosť systému. V oblasti vývoja softvéru medzi ne patrí systémový návrh, funkčné obsah a jasnosť zdrojového kódu. Ak chceme zistiť, či nám metriky môžu pomôcť zlepšiť systémy ktoré vyvíjame, musíme lepšie porozumieť softvérovej zložitosti. Dokážu nám metriky povedať niečo o jej povahe?

Ďalším názorom je, že úspech metriky pri vývoji softvéru nespočíva v ich schopnosti merať konkrétne softvérové entity. Miesto toho majú metriky poskytnúť nástroj na objektívne zdôvodňovanie rozhodnutí pri procese vývoja softvéru. [4] Metriky sú nástrojom, vďaka ktorému sme schopní pozorovať a rozhodovať aspoň o niektorých aspektoch softvéru. To nám zase umožňuje overenie a vyhodnotenie procesov.

Záver

Výhoda agilného spôsobu vývoja je v rýchlom a postupnom vývoji softvéru po malých častiach, vďaka čomu sú výsledky viditeľné postupne po jednotlivých šprintoch. Tento fakt sám o sebe napomáha k čiastočnému odstráneniu problému neviditeľnosti softvéru. Rovnako napomáha aj použitiu metrík, keďže metriky lepšie fungujú pri spätnom vyhodnocovaní výsledku ako pri predpovediach. Tento fakt je pravdepodobne dôvodom úspešnosti a populárnosti agilných spôsobov vývoja.

Na plánovanie a kontrolu postupu prác je podľa môjho názoru vhodné používať bežné spôsoby monitorovania agilných projektov, ako napríklad burn-down chart, burn-up chart, alebo rýchlosť. Avšak dôležitým cieľom všetkých projektov by malo byť udržanie zdrojového kódu na nízkej úrovni zložitosti, čo zabezpečí lepšiu udržiavateľnosť, jednoduchšie pochopenie a rozšíriteľnosť zdrojového kódu. Preto je vhodné používať aj metriku na meranie zložitosti kódu, napríklad cyklomatickú zložitosť.

Pri monitorovaní softvérových projektov nie je rozhodujúca zvolená metrika, a rozhodne nie je prínosom využívanie veľkého množstva rôznych metrík na tú istú problémovú doménu. Dôležité je, aby vôbec nejaká metrika pri monitorovaní projektov bola použitá, a na jej výsledky bolo prihliadané pri vykonávaní rozhodnutí. Netreba však zabúdať, že metriky sa snažia odhadnúť zložitosť neviditeľného problému a preto im netreba slepo dôverovať, ale treba akceptovať aj vedomosti a intuíciu ľudí, ktorí sú odborníkmi v danej oblasti.

Použitá literatúra

1. Oram, Andy. 2010. *Do we need more software complexity metrics?*. http://answers.oreilly.com/topic/2258-do-we-need-more-software-complexity-metrics/
2. Ktata, O., Lévesque, G.: *Designing and Implementing a Measurement Program for Scrum Teams: What do agile developers really need and want?*. In: Proceedings of the Third C* Conference on Computer Science and Software Engineering, 2010, pp. 101-107.
3. Cheng, T-J., Jansen, S., Remmers, M.: *Controlling and Monitoring Agile Software Development in Three Dutch Product Software Companies*. In: Proceedings of the 2009 ICSE Workshop on Software Development Governance, 2009, pp. 29-35.
4. Fenton, N., Neil, M.: *Software Metrics: Roadmap.* In: ICSE – Future of SE Track, 2000, pp. 357-370.

Annotation

Usefulness of monitoring and agile software development

To facilitate planning and improvement of monitoring of the development process, there are various metrics that try to measure a product or process itself. Software metrics, however, suffer from some problems which arise from the nature of the software itself. The essay will explain the nature of these problems and it will try to answer the question about contribution of metrics to software development process. An important issue is the agile software development, which is used by more

and more companies these days. In accordance with the agile methodology, decisions are made by professionals and rely mainly on intuition. Recently, there were various modifications of metrics used for the needs of agile development process. If these metrics have to be meaningful, they have to be modified to reflect the agile development methodology. Can these metrics be beneficial? Is it useful to use them?