Meranie v softvérovom projekte

JAKUB TEKEĽ

Slovenská technická univerzita
Fakulta informatiky a informačných technológií
Ilkovičova 3, 842 16 Bratislava
jakubtekel@gmail.com

Abstrakt. Najväčším problémom pri tvorbe softvérového produktu je jeho neviditeľnosť. Mnohé projekty sú zdanlivo takmer ukončené, ale potrebná práca presahuje množstvo tej vykonanej. Aby bolo možné priebeh tohto procesu kontrolovať, vznikajú rôzne spôsoby monitorovania. Výsledky meraní sú dôležité nie len pre kontrolu práce, ale hlavne pre riadenie a plánovanie softvérového procesu. Skutočnú cenu majú len aktuálne informácie, a preto musí byť ich zberanie v maximálnej možnej miere automatizované. Problém nastane, ak členovia tímu začnú namiesto práce produkovať len dobré výsledky meraní. Preto je veľmi dôležité zvoliť vhodné metriky a získané informácie správne spracovať. Táto práca sa zaoberá rôznymi spôsobmi monitorovania softvérového projektu, a vhodnosťou ich použitia v závislosti od vlastností projektu.

Úvod

Rozsiahle softvérové projekty vyžadujú veľký počet ľudí. Tento tím samozrejme treba riadiť a kontrolovať, inak by sa náklady na projekt neúmerne zvyšovali. Sleduje sa aj plán projektu, ktorý nemusí správne odhadnúť potrebné úsilie či rozloženie zdrojov a musia sa vykonávať operatívne zmeny. Každý dobrý plán obsahuje merateľné ciele, ohraničenia a dátumy. Nemožno kontrolovať to, čo sa nedá zmerať. Preto plnenie takéhoto plánu sa dá kontrolovať jedine vtedy, keď aj kontrola produkuje merateľné hodnoty. Problémom je teda zvoliť údaje, ktoré treba merať.

Najjednoduchšie sa projekt kontroluje po skončení práce, pretože stačí vyhodnotiť splnenie cieľov. Hoci je táto kontrola dôležitá, neprodukuje žiadne informácie potrebné pre riadenie projektu. Dáta spojené s časovými údajmi umožňujú zviditeľniť priebeh projektu, a odhaľovať súvislosti medzi samotnými meranými údajmi. Preto je užitočné používať nástroje a metódy, ktoré umožňujú priebežnú kontrolu. Túto prácu môže uľahčiť aj automatické zberanie meraných údajov.

Manažment v softvérovom inžinierstve, október 2007, s. 1-7.

Rozdelenie práce v softvérovom projekte

Takmer každý projekt sa skladá z viacerých častí a menších úloh. V jednoduchšom prípade sa vytvárajú časti za sebou, pričom každá úloha sa vypracuje až po ukončení predošlej. Veľké tímy samozrejme pracujú z časových dôvodov na viacerých úlohách paralelne. Obyčajný programátor pracuje na zadanej úlohe, sústreďujúc na ňu väčšinu svojej pozornosti. Zaujíma ho, ako vyriešiť technický problém. Naopak manažér vidí tieto úlohy z globálneho hľadiska, ale prehliada detaily. Pre manažéra je podstatné vnímať aktuálny stav a vidieť do budúcnosti, aby mohol jednotlivé tímy zosúladiť.

Programátor sa nachádza v jednoduchej pozícii, pretože jeho problém je jasne formulovaný. Naproti tomu manažér musí kontrolovať tvorbu softvéru, ktorý je neviditeľný a nestály [4]. Preto napredovanie projektu sleduje nepriamo. Zberom údajov ako počet riadkov kódu sa dá sledovať manuálna práca programátorov. Kontrolovaním špecifikácie je možné sledovať plnenie cieľov. Niektoré činnosti sú ťažko deliteľné, a aby bolo možné sledovať pokrok počas ich trvania, skladajú sa viaceré spôsoby monitorovania.

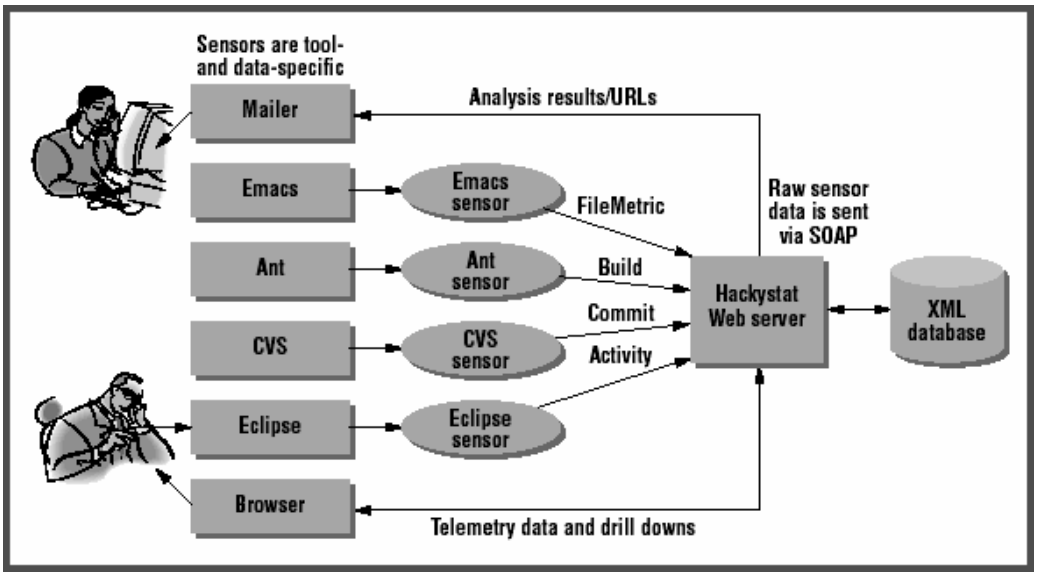
Sledovanie práce

Prvý spôsob monitorovania je priamočiary. Bude sa merať objem fyzicky vykonanej práce. Tieto údaje sa dajú ľahko získať pomocou softvérových nástrojov. Aj preto sa tieto metriky často v praxi používajú. Merať sa môže veľa údajov, medzi častejšími je to napríklad:

- Počet napísaných riadkov
- Počet opravených chýb
- Počet uskutočnených hovorov
- Čas strávený v kancelárii

Praktická realizácia týchto meraní sa robí pomocou prídavných programov, ktoré sledujú používateľa pri práci. Pre každý nástroj využívaný pracovníkom je vytvorený senzor, ktorý zbiera údaje a väčšinou ich posiela na ďalšie spracovanie (Obr. 1). Tento spôsob merania sa nazýva telemetria [1]. Základné vlastnosti sú:

- Zbieranie údajov prebieha automaticky, nie je ohraničené priestorom a vzdialenosťou
- Merané údaje sa skladajú aj z časového údaju, ktorý je dôležitý pre spracovanie
- Údaje je možné spracovať aj keď ich zber nebol zahájený na začiatku projektu.



Obr. 1. Architektúra monitorujúceho nástroju Hackystat [1]

Počet riadkov však nehovorí nič o tom, koľko funkčných vlastností daný systém poskytuje. Kontrola charakteristík zdrojového kódu ale môže byť užitočná z programátorského hľadiska, napríklad aby sa kontrolovala čitateľnosť kódu, správne pomenovanie objektov alebo úroveň previazania modulov.

Množstvo telefonátov a čas strávený v kancelárii môže byť dobrým signálom, že zamestnanec niečo robí. Monitorovanie týchto činností môže byť užitočné z operačného hľadiska, pretože manažér porovná frekvenciu a množstvo úkonov z predošlým obdobím, čím získa hrubý odhad vykonanej a potrebnej práce.

Základným problémom týchto metrík je, že pracovníci majú tendenciu (možno aj neúmyselne) zmeniť štýl práce tak, aby maximalizovali namerané hodnoty. Programátor, ktorý chce vyriešiť viac chýb, tak bude riešiť kozmetické úpravy, a kritickú chybu nechá v systéme. Druhým problémom je fakt, že perfektne napísaný program, ktorý neobsahuje ani jednu chybu, vôbec nemusí riešiť zadanú úlohu. Programátori nemali čas naprogramovať príslušnú funkcionalitu, ale merania vykazovali vysoké hodnoty, takže manažér si nič nevšimol. Navyše čas strávený pri riešení úlohy a vykonané úkony nehovoria nič o efektivite riešenia. Telemetria nedokáže odhaliť ani vynaložený čas a prostriedky na riešenie úlohy mimo pozorovaných bodov, napríklad pri rozhovore na chodbe.

Posledný problém je možno mierne neočakávaný. Pracovníci môžu mať problém so sledovaním pri práci, pričom nejde len o paranoju. Zamestnanci sa jednoducho obávajú, že budú porovnávaní s inými pracovníkmi, ktorí zvolili rozdielnu pracovnú metódu. Porovnanie údajov medzi takýmito pracovníkmi by mohlo viesť k domnienke, že jeden z nich pracuje menej. Okrem toho namerané hodnoty nie je vhodné skrývať pred samotnými zamestnancami. Každý má právo vedieť, že vykazuje podpriemerné

výsledky, a mal by zvýšiť úsilie, kým je ešte čas. Najlepším riešením je zverejnenie nameraných údajov. Vďaka tomu sa môžu všetci, pracovníci aj manažéri, naučiť správne vykladať tieto informácie, a odhadovať blízku budúcnosť projektu.

Sledovanie splnených cieľov

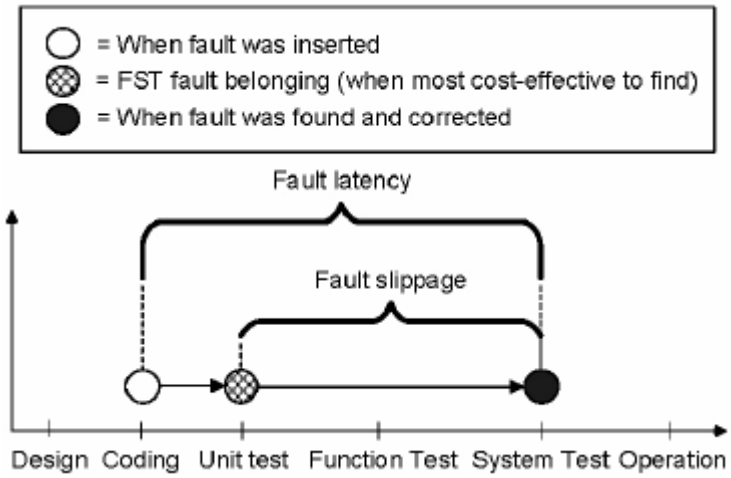
Z uvedených problémov vyplýva, že nestačí len zozbierať množstvo údajov. Manažéra nezaujíma ako je riešenie naprogramované, zaujíma ho plnenie cieľov. V prvom rade si treba uvedomiť, že nie je správne najprv údaje zbierať a neskôr sa zamýšľať nad tým, ako ich využiť. Naopak, na základe cieľov treba navrhnúť merania tak, aby bolo možné kontrolovať plnenie práve týchto cieľov. Napríklad pri projekte s používateľským rozhraním sa nebude kontrolovať množstvo funkcií v zdrojovom kóde, ale množstvo funkcií prístupných používateľovi. Tieto funkcie samozrejme majú rôznu dôležitosť, takže nestačí triviálne spočítanie tlačidiel na obrazovke. Problém je, že s jednoduchými metrikami uvedenými v predošlej kapitole sa takáto úloha nedá realizovať.

Znovu platí: Čo sa nedá zmerať, nedá sa kontrolovať. Preto treba najprv zadefinovať každý cieľ, a následne spôsob jeho kontroly. Fráza „prehľadné ovládanie“ sa zmení na „tlačidlo každého príkazu sa nachádza na úvodnej obrazovke“. V spomínanom príklade s funkciami sa najprv určí dôležitosť jednotlivých funkcií číselne, a potom sa hodnotí ich splnenie podľa priloženej tabuľky.

Dôležitosť funkcie ale často krát nie je zhodná s jej zložitosťou. Manažér chce vedieť, ako ďaleko postúpila práca. Preto sa v pláne nachádzajú míľniky, ktoré delia úlohu na menšie časové úlohy. V týchto bodoch sú potom jednoznačne definované metódy overenia, či sa podarilo tento míľnik splniť. Stav práce medzi dvoma míľnikmi sa musí určovať inými spôsobmi – napríklad automatizovanými metrikami.

Ciele však nemusia byť obmedzené len na koncový produkt. Manažér si môže stanoviť za cieľ nízky počet chýb v programe. Nutnou podmienkou je možnosť kontroly. Pri tomto cieli sa však nemôže v pláne nachádzať zmienka o tom, koľko chýb sa má v projekte odhaliť alebo opraviť. Navyše takáto informácia nikomu nepomôže, pretože neposkytuje odhad o tom, koľko chýb sa ešte v programe nachádza. Chybovosť produktu sa ale sledovať dá.

Riešením je správne definovať kvalitu práce. V uvedenom probléme chybovosť neurčuje počet odhalených chýb. Dobrým prístupom je napríklad sledovať čas oneskorenia odhalenia chýb [2]. Pre každú nájdenú chybu sa určí, kedy vznikla príslušná časť kódu a kedy mala byť chyba odhalená v ideálnom prípade (Obr. 2). Chyba, ktorá bola odhalená pri prvom testovaní danej funkcie, nepredstavuje vôbec žiaden problém. Sledovať sa bude pomer neskoro odhalených chýb a celkového počtu chýb. Ak je výsledok malý, je možné konštatovať, že počet pretrvávajúcich chýb je rovnako malý. Z tohto je možné utvoriť záver, že výsledný program obsahuje malý počet chýb.



Obr. 2. Metóda fault slippage sleduje oneskorenie odhalenia chyby [2]

Tento spôsob sa dá zovšeobecniť aj na sledovanie iných vlastností vytváraného softvérového produktu. Dôležitou skutočnosťou je, že sa nesleduje len jeden elementárny údaj. Sledovaním podielu vhodných vstupov získame percentuálne údaje, ktoré je možné vyhodnocovať aj keď projekt nie je ukončený, a to dokonca aj vtedy, keď nie je známy aktuálny stav projektu. Navyše je možné porovnávať aj projekty navzájom, a to aj za predpokladu, že nie sú rovnako veľké.

Vyhodnotenie meraní, riadenie

Hlavnou motiváciou merania a kontrolovania softvérového projektu je zviditeľnenie aktuálneho stavu projektu. Ďalej dáva manažérovi možnosť predpovedať pokrok a problémy v projekte. Aby bolo toto všetko možné, musia sa splniť dve nutné veci: Treba monitorovať správne zvolené údaje a je nutné ich správne vyhodnotiť.

Potreba vybrať vhodné údaje na monitorovanie nemusí na prvý pohľad vyzerať zložito. V praxi sa ale stáva, že sa sledujú aj úplne zbytočné údaje len preto, že sa sledujú jednoduchšie ako ostatné. Napríklad v [1] pracovný tím prekvapivo zistil, že pravdepodobnosť kompilačnej chyby nezávisí od dĺžky pripísaného zdrojového kódu. Zdôvodniť sa to dá tak, že pracovník venuje kontrole dlhšieho kódu aj viac času.

Po úspešnom zmeraní údajov sa pristúpi k ich vyhodnoteniu. Najčastejší spôsob analýzy údajov je prirovnávať ich k už existujúcim dátam z ukončených procesov. Tento postup prináša kvalitné výsledky aj vďaka svojej nízkej náročnosti. Skúsenosti manažéra samozrejme môžu zlepšiť výsledok.

Údaje sa môžu použiť na vytvorenie matematického modelu. Používané sú napríklad plány v tvare stromov a orientovaných grafov. Použiť sa dá fuzzy logika [3]. Vďaka tomu je možné vypočítať kritické dátumy plánu aj keď sú niektoré údaje nepresné alebo chýbajúce.

Posledný často používaný spôsob tvoria rôzne heuristiky a takzvané dobré odporúčania. Hoci tieto pravidlá nie sú presné, ich výhodou je jednoduchá aplikácia.

Ja si myslím, že práve vyhodnotenie informácii je tá časť projektu, kde najviac vyniknú schopnosti manažéra. Spôsob monitorovania sa dá používať v každom projekte firmy rovnako, a predsa nebudú výsledky zhodné. Je tomu tak preto, že manažér má očakávať aj nečakané. Riadenie sa preto vôbec nedá naučiť jednoducho. Môj názor je, že bez monitorovania softvérového projektu sa plytvá zdrojmi už aj pri dvojčlennom tíme. Z vlastných skúseností poznám situácie, keď sa programátor presvedčený o svojom dobrom postupe márne skúša naprogramovať zlým postupom niečo, čo jeho kolega zvládne za desať minút. Takýto človek bude stále tvrdiť, že je s prácou už takmer hotový, a potrebuje najviac päť minút.

Záver

Riadenie softvérového tímu je náročný proces. Aj s najkvalitnejšie navrhnutým plánom a najlepšími pracovníkmi sa dajú očakávať odchýlenia od vytýčeného rozvrhu. Aby manažér projektu dokázal odhaliť tieto odchýlenia, nutne potrebuje prehľad nad stavom a smerovaním práce. Vzhľadom na zložitosť, nestálosť a neviditeľnosť softvéru by sa bez vhodných prostriedkov na kontrolu takýto cieľ nikdy nepodarilo dosiahnuť. Riešením je meranie práce a kontrola splnenia cieľov.

Monitorovanie fyzickej práce s použitím programov je jednoduchým a rýchlym spôsobom, ako získať základné informácie o napredovaní projektu. Hlavnou výhodou je, že takéto údaje sú vyčísliteľné. Keďže však umožňujú len meranie jednoduchých veličín, väčšinou sa nedajú použiť na kontrolu splnenia cieľov.

Na to, aby bolo možné kontrolovať funkčnosť produktu, musia sa najprv jeho vlastnosti vhodne formulovať. Každá vlastnosť totiž musí byť merateľná. Táto podmienka zvyčajne vylučuje použitie automatického vyhodnocovania. Aby sa produkovali aj priebežné výsledky, je vhodné prácu deliť na menšie úlohy, ktoré sa dajú vyhodnocovať samostatne.

Kontrola a riadenie projektu je hlavne umenie dobre zvoliť čo treba sledovať, a ako to sledovať. Zložitosť dnešných systémov vyžaduje, aby bolo toto monitorovanie čo možno najefektívnejšie, inak sa energia pri práci vydáva nadarmo. Dnes sa už žiaden manažér bez monitorovania nezaobíde.

Použitá literatúra

1. Johnson, P. M., Kou, H., Paulding, M., Zhang, Q., Kagawa, A., Yamashita, T.: Improving Software Development Management through Software Project Telemetry, *IEEE Software*, Vol. 22, No. 4 (2005), pp. 76-85.
2. Damm, L. and Lundberg, L. 2006. Using fault slippage measurement for monitoring software process quality during development. In *Proceedings of the 2006 international Workshop on Software Quality* (Shanghai, China, May 21 - 21, 2006). WoSQ '06. ACM Press, New York, NY, 15-20

3. Cîmpan, S. and Oquendo, F. 2000. Dealing with software process deviations using fuzzy logic based monitoring. *SIGAPP Appl. Comput. Rev.* 8, 2 (Dec. 2000), 3-13
4. Bieliková, M.: Softvérové inžinierstvo. Princípy a manažment. Vydavateľstvo STU Bratislava, 2000.

Annotation

Measurement in software project

The biggest problem of software product development is its invisibility. Many projects are seemingly near completion, but required work exceeds work done. Different ways of monitoring have been developed in order to control the progress of this process. Results of measurements are not only essential for control of work, but also for management and planning of software process. Only actual information is really valuable, and because of that gathering of information must be automatic in most possible way. Problem arises when members of team begin production of good measurement results instead of doing real work. Because of that, it is important to choose a good metrics and to analyze collected information in correct way. Work presented in this paper deals with different methods of software process monitoring, and their suitability depending on properties of project.