SLEDOVANIE ÚLOH V PROJEKTE A PROBLÉMY S TÝM SPOJENÉ

Keď som sa dnes ráno pozrel do zrkadla,
videl som, o koľkej pôjdem zajtra večer spať.

Ján Zdechovan

Slovenská technická univerzita
Fakulta informatiky a informačných technológií
Ilkovičova 3, 842 16 Bratislava
jan[.]zdechovan[zavináč]ynet[.]sk

Abstrakt. *V eseji sa venujem problematike sledovania priebehu úloh v softvérovom projekte. Zameriavam sa na problémy, ktoré vznikajú v dôsledku zanedbaného alebo nedostatočného reportovania stavu úloh. Opisujem následky, ktoré môžu v projekte vzniknúť z tohto nedostatku a rozoberám, ako ovplyvňujú priebeh celého projektu z pohľadu ďalšieho plánovania a riadenia, ale aj to, ako by sa dali informácie získané sledovaním stavu úloh využiť či už v prospech celého projektu, alebo na podporu osobného manažmentu jeho členov. Ďalej sa zaoberám možnými riešeniami, ktoré by mohli pomôcť tento problém eliminovať. Predstavujem tu tiež v krátkosti niektoré existujúce nástroje na sledovanie úloh, ktoré boli navrhnuté na riešenie takýchto problémov.*

Kľúčové slová: *sledovanie, riadenie, úloha, vývoj, reportovanie, monitorovanie, tím, softvérový projekt*

Sledovanie úloh v softvérovom projekte

Softvérový projekt často nie je triviálnou záležitosťou. Najmä pri rozsiahlejších projektoch, keď je potrebné zapojiť väčšie množstvo zdrojov a riešiť viacero zložitých úloh, je potrebné zabezpečiť kvalitné riadenie projektu. To zahŕňa plánovanie a vytváranie rozvrhu prác, následné monitorovanie vykonávania úloh, z ktorého zvyčajne vyplýva ďalšie plánovanie, prípadne iné zásahy zo strany manažéra projektu.

Proces riadenia vývoja

Proces vývoja prebieha zvyčajne tak, že projektový manažér, zvyčajne za pomoci skúseného vývojára vytvorí zoznam úloh, ktoré treba realizovať, pridelí ich jednotlivým pracovníkom a dohliada na ich vykonanie. Počas práce očakáva priebežné správy o stave jednotlivých úloh. Pre manažéra sú tieto informácie dôležité, pretože z nich môže vidieť, v akom štádiu sa momentálne celý projekt nachádza. Vie zistiť, či nenaberá meškanie, či nenastali problémy s implementáciou nejakých funkcií a podobne, čo mu umožňuje aj včas reagovať na vzniknutú situáciu.

Softvérové inžinierstvo sa však od iných inžinierskych disciplín líši vo viacerých aspektoch [6]. Jedným z nich je aj to, že softvér je nehmatateľný, a teda problémy pri jeho vývoji sú navonok nespozorovateľné. V iných disciplínach, napríklad v stavebníctve tomu tak nie je – ak stavba naberá sklz, je jasne vidieť, čo ešte nie je postavené. Na druhej strane v softvérovom inžinierstve je projektový manažér závislý od informácií, ktoré mu poskytnú iní. Pokiaľ sa k nemu nedostanú alebo sa dostanú príliš neskoro, môže sa stať tento problém kritickým, pretože to povedie k neočakávaným udalostiam a môže sa nepriaznivo prejaviť v podobe prekročenia rozpočtu, termínu odovzdania alebo nedostatočnej funkcionality produktu.

Vývojári

Vývojári pracujú na svojich úlohách v priebehu dňa. Okrem písania kódu, ktoré zaberá zhruba polovicu ich pracovného času [3], popisujú svoju prácu rôznymi spôsobmi. Niektoré z možných foriem sú [5]:

- *Vlastné poznámky* – v elektronickej alebo papierovej podobe, obsahujú najmä poznámky dôležité pri nadväzovaní prerušenej úlohy
- *Denník* – miesto, kde sa zapisuje čas strávený nad jednotlivými aktivitami počas dňa
- *Sociálne prostriedky* – slúžia najmä na zdieľanie informácii o aktivitách medzi spolupracovníkmi, patria sem napr. fóra, maily, prípadne blogy
- *Odovzdávacie správy (commit messages)* – komentovanie aktivít po úpravách v kóde
- *Záznamy zo systému na sledovanie úloh* – napr. Trac[1], JIRA[2], Bugzilla[3]

Každá forma môže byť užitočná pre iný účel. Napríklad manažéra projektu nemusí zaujímať, ako sa vývojárovi podarilo vyriešiť dlhodobý problém a jeho implementačné detaily, o ktoré sa delí so spolupracovníkmi v databáze znalostí. Bude ho však zaujímať, koľko mu táto úloha trvala a či ňou neohrozil aktuálny rozvrh.

Z pohľadu manažéra projektu je teda na ňom, ktorý zdroj informácií použije. V ideálnom prípade by však nemal mať problém s nedostatkom informácií o projekte.

---

[1] http://trac.edgewall.org/
[2] http://www.atlassian.com/software/jira/
[3] http://www.bugzilla.org/

Kde je problém?

Práve na základe vyššie uvedeného dôvodu si myslím, že táto téma si vyžaduje pozornosť zo strany všetkých zúčastnených a že je naozaj vhodné, aby vývojári poskytovali relevantné informácie o postupe ich práce v primeraných intervaloch, čo možno najpresnejšie a s najnižším možným oneskorením, a to najmä v prípade vzniknutého problému. V ideálnom svete to problém byť nemusí, avšak reálne je situácia iná.

Existuje niekoľko problémov, ktoré môžu spôsobiť „zrútenie" rozvrhu [4]. Jeden z nich, ktorý pramení práve z nedostatočnej informovanosti projektového manažéra, je práve problém s disciplínou vývojárov. Autori poukazujú na to, že disciplína reportovania je v tímoch často nedostatočná aj napriek existencii rôznych nástrojov v organizáciách, ktoré podporujú pravidelné vykazovanie činnosti. Tie môžu byť vo forme mailov, rôznych nástrojov na sledovanie úloh, ale aj osobných stretnutí.

Tu si myslím, že tento problém môže byť menší v prípade malých organizácií s malými projektmi, kde projektový manažér riadi menší počet ľudí alebo sa dokonca sám podieľa na vývoji. Ale naopak vo veľkých firmách, kde sa pracuje na niekoľkých rozsiahlych projektoch súčasne, manažéri často nemajú čas na urgovanie reportovania a preto im podstatné veci nemusia byť včas známe.

V štúdii Maaleja a Happela [5] nájdeme zaujímavé zistenia, ktoré popisujú okrem iného aj vynaložené úsilie vývojárov pri reportovaní úloh v projektoch rôznych rozsahov a z rôznych prostredí. Podľa ich zistení 10-20% skúmaných reportov, v závislosti od projektu, tvorili pseudoreporty, čo sú vlastne reporty, ktorých obsah buď úplne absentuje alebo nemá zmysel. Podľa môjho názoru tieto výsledky do určitej miery vypovedajú o nedisciplinovanosti vývojárov. Z vlastných skúseností viem, že takéto reporty sa vyskytujú pomerne často. Ich nekompletnosť sa potom rieši dodatočnou komunikáciou medzi spolupracovníkmi, resp. medzi manažérom a vývojármi. Nedostatočná informovanosť predstavuje komplikáciu, ktorej riešenie spočíva v nutnosti vynaložiť dodatočné úsilie a čas k získaniu potrebných informácií.

Ale prečo je to tak?

Keď sa zamyslím nad možnými dôvodmi tohto problému, naskytne sa viacero možností. Niektoré uvádza aj literatúra [4]. Pokúsil som sa ich doplniť o vlastné, ktoré som odpozoroval z praxe.

- Vývojári často nemajú čas zaoberať sa reportovaním počas pracovnej doby, pretože majú veľa práce.
- Po pracovnej dobe nemajú motiváciu túto aktivitu dokončiť. Pokiaľ ich k tomu núti nejaký nástroj, popisujú prácu nekorektne a vytvárajú pseudoreporty.
- Úlohy, ktoré sú reportované s dlhším časovým odstupom, sú reportované nepresne.
- Pri reportovaní počas pracovného dňa sa stáva, že vývojári sú zahltení prácou a nie sú schopní napísať podrobnejší report.
- Pri striedaní úloh počas dňa nie je vždy možné aktualizovať ich stav.

Niektoré z uvedených príčin taktiež nemusia byť relevantné, napr. príliš časté reportovanie (niekoľkokrát počas dňa) nemusí byť vždy žiaduce. Člen tímu by mal vedieť posúdiť, s čím je potrebné manažéra oboznámiť a tiež to, kedy tak má urobiť.

Bežný proces vykonávania rozvrhu trvá približne 2-3 týždne [6], potom dôjde k jeho prehodnoteniu. Preto by niektoré reporty mohli byť urobené ku koncu tohto obdobia. Keď odhliadneme od situácií v priebehu plnenia úloh, ktoré nastávajú náhle a treba ich urgentne riešiť, mohlo by to postačovať. Avšak aj z vlastných skúseností viem, že pokiaľ sa reporty robia z časovým odstupom, nie vždy sa dá spomenúť si na určité detaily, ktoré by mohli byť dôležité.

Na toto poukazuje aj spomínaná štúdia Maaleja a Happela [5], kde zo skúmaných dát vyplynulo, že veľká časť reportov bola vytvorená až na druhý deň po príslušnej aktivite a v menšom rozsahu. Tu mi chýba informácia o tom, ako sa na rozsahu reportu prejaví ešte dlhšie obdobie od vykonania danej aktivity. Nakoľko sa v praxi stáva, že vývojári si nechávajú reportovanie aj na koniec pracovného týždňa.

Počas riešenia rozpracovanej úlohy je niekedy potrebné riešiť nejakú inú úlohu s vyššou prioritou, opraviť chybu blokujúcu ďalšie práce alebo sa dočasne venovať inej pracovnej činnosti. Po návrate k pôvodnej úlohe sa vývojár v prvom rade snaží vrátiť k pôvodnej myšlienke a reportovanie úlohy radšej necháva na neskôr. Problém vzniká, keď sa takýchto úloh nazbiera počas dňa viac (môže ísť pri tom o jednoduché úlohy). Tu sa môže stať, že si pri reportovaní prvej úlohy nespomenie, že sa venoval aj inej činnosti, pretože sa primárne sústredil na rozpracovanú úlohu.

O čo sa vlastne oberáme ?

V prípade, že sa rozvrh pravidelne neaktualizuje, stáva sa nepoužiteľný. Obmedzí to manažérovu schopnosť sledovať vykonávanie plánu [4]. Tak, ako problém nedostatočného reportovania môže spôsobiť vážne problémy, je dobré zamyslieť sa aj nad výhodami, ktoré získame, keď sa tento problém podarí odstrániť.

Nástroj pre manažment

Z pohľadu tímu ako celku je možné identifikovať včas vzniknuté problémy a reagovať na ne. V prípade, že projekt naberie sklz, napríklad v dôsledku neočakávaných problémov s nasadením novej technológie, môže projektový manažér rýchlo zasiahnuť vytvorením špeciálnej úlohy s vysokou prioritou, ktorú pridelí skúsenému členovi tímu, ktorého dočasne odvolá z riešenia inej, menej dôležitej úlohy. Doba, za ktorú vie zareagovať, závisí v podstatnej miere od reportovania stavu pôvodnej úlohy.

Kvalitné reporty môžu poslúžiť ako ukazovateľ produktivity zamestnancov. Z odovzdávacích správ je možné vypočítať približný počet riadkov kódu, ktoré boli vyprodukované za časovú jednotku. Myslím si však, že lepší prehľad by mohol poskytnúť počet takýchto správ za jednotku času, nakoľko počas editovania zdrojového kódu sa riadky aj mažú.

Prínos pre vývojárov

Reportovanie však môže byť výborným nástrojom aj pre samotných vývojárov. Reporty im poskytujú skvelý nástroj na sledovanie ich práce. Napríklad to môže byť doba strávená na riešení úlohy, problémy, ktoré sa následne odhalili pri testovaní a charakter takýchto chýb. Tieto informácie sa dajú využiť pri analýze činnosti v osobnom softvérovom procese (PSP – Personal Software Process) [1]. Tento proces definuje prostriedky, ako sa môže jednotlivec na základe analýzy svojich predchádzajúcich výsledkov naučiť, ako vytvorí efektívne kvalitný produkt.

Reporty s určitou informačnou hodnotou môžu byť taktiež využité ako pomôcka, keď sa vývojár potrebuje počas dňa pohybovať medzi rozpracovanými úlohami. Dozvie sa, na čom pracoval naposledy, čo mu pomôže dostať sa do obrazu.

Myslím, že reporty by mohli podporovať aj spoluprácu pri práci na určitých častiach kódu. To, ako kód funguje a popis jednotlivých komponentov sa zvyčajne zachytáva v komentároch. Pokiaľ ale kód prechádza zmenami a na jeho úpravách pracuje viac ľudí, zdá sa mi vhodné a žiaduce, aby sa informácie o zmenách a rozhodnutiach s nimi spojenými písali do reportov. Spätné nahliadnutie do histórie odovzdávaní daného kódu by potom mohlo v súčinnosti s komentármi pomôcť pri vytváraní si predstavy o kóde alebo pri odhaľovaní príčiny chýb.

Možnosti riešenia

Problémy s nedostatkami v reportovaní sa dajú riešiť viacerými spôsobmi. Pokiaľ projektový manažér zistí, že nastávajú komplikácie v súvislosti s nedisciplinovanosťou vývojárov pri reportovaní, má možnosť zakročiť.

Tvrdá ruka

Niektorí sa pokúšajú túto situáciu vyriešiť formou rôznych postihov, čo môže byť efektívne, ale rovnako aj kontraproduktívne. Môže viesť k reportovaniu, ktoré bude síce pravidelné, ale o to nepresnejšie a povrchnejšie.

Podľa mňa by riešenie mohlo ležať niekde v strede. Bola by dovolená určitá tolerancia z časového hľadiska, avšak členom tímu by bolo vysvetlené, ako reportovaním pomáhajú k dosiahnutiu cieľa, a teda k úspešnému ukončeniu projektu. Taktiež by som sa snažil vyzdvihnúť, ako reportovanie môže pomôcť pri práci jednotlivca a skupiny a zvýšiť tak celkovo produktivitu práce. Myslím, že pokiaľ by sa to stretlo s pochopením, mohlo by to vylepšiť situáciu a navyše, po pochopení cieľa reportovania, by sa mohol zlepšiť ich obsah – t.j. uvádzanie presnejších a relevantnejších informácii. Na druhej strane priznávam, že by nemuselo ísť o jednoduchú úlohu.

Lepšie nástroje

Prostriedky na zlepšenie sú aj v samotných nástrojoch na tvorbu softvéru, respektíve v nástrojoch na správu úloh.

Jeden z dôvodov, prečo vývojári neradi trávia čas reportovaním počas pracovného dňa, ktorý som ešte neuviedol, je to, že reportovanie statusu a samotné vývojové nástroje sú často úplne odlišné aplikácie. Pokiaľ vývojár niečo vytvorí, uzavrie úlohu a podobne, je

potrebné, aby si otvoril systém na zapisovanie činnosti a manuálne popísal túto činnosť, čo zaberá určitý čas a ruší jeho sústredenie.

Tento nedostatok trápi nielen manažment, ale aj samotných vývojárov. Preto vznikajú vo vnútrofiremnom, ale aj vonkajšom prostredí rôzne nástroje a vylepšenia podporujúce tieto aktivity. Napríklad jedny integrujú vývojové prostredia a nástroje na sledovanie úloh a reportovanie [4], iné nástroje slúžia na zber a analýzu informácií, ktoré vznikajú aktivitami vývojára počas písania programu [2]. Myslím si, že tieto nástroje môžu do veľkej miery pomôcť odstrániť spomínané problémy.

Dostupné nástroje

Doteraz som mal možnosť stretnúť sa s viacerými riešeniami, ktoré by mohli byť prakticky užitočné pre vývojárov. Pokúsim sa niektoré z nich v krátkosti popísať.

Dobrým príkladom je *Eclipse Project and Resource Tracking Plugin* predstavený v štúdii Li-ho a Deuga z roku 2008 [4]. Tento nástroj je vytvorený ako zásuvný modul do vývojového prostredia Eclipse[4]. Vývojár v ňom má pri písaní kódu k dispozícii informácie o samotnom projekte, o aktuálnych úlohách pre seba, ale aj svojich spolupracovníkov. Tieto informácie sa nachádzajú v tej istej aplikácii, v ktorej produkuje kód, takže nie je potrebné prepínať sa medzi rôznymi aplikáciami. Používateľ môže vytvárať aj nové úlohy, čo je prínosom v prípade, ak sa vyskytnú nečakané úlohy a činnosti počas práce. V programe mi však chýba podpora automatického zaznamenávania stráveného času, kedy by sa používateľ počas práce jednoducho pohyboval medzi úlohami, a tak by sa strávený čas pripočítaval automaticky.

Ďalší podobný nástroj pre Eclipse je *Mylyn*[5]. Ten sa zameriava na integráciu s rôznymi dostupnými systémami na sledovanie úloh. Okrem spomenutých funkcií, vrátane definovania úlohy, na ktorú sa aktuálne sústredíme a zaznamenávania stráveného času s ňou, umožňuje definovať aj kontext, teda jednotlivé zložky kódu, ktoré s úlohou súvisia a ďalej ich pri práci sledovať. Tento nástroj podľa môjho názoru spĺňa najviac z požiadaviek, aké boli zatiaľ definované.

Záver

Problémy s reportovaním úloh v softvérovom projekte môžu byť nepríjemné, pretože zabraňujú projektovým manažérom pri získavaní potrebných informácii. Potlačenie týchto problémov môže viesť k zvýšeniu produktivity jednotlivcov a celého tímu, a taktiež môže poskytnúť nástroje na zlepšenie riadenia. Možností, ako to dosiahnuť, existuje viac. V podstatnej miere to môže byť zmena prístupu členov tímu ako aj používanie nástrojov s lepšou podporou pre správu úloh.

---

[4] http://www.eclipse.org
[5] http://www.eclipse.org/mylyn/

Použitá literatúra

1. Humphrey, W.S.: Using a defined and measured personal software process. In: *IEEE Software*, Vol. 13, No. 3, (1996), 77–88
2. Johnson, P.M., Kou, H., Paulding, M.G. et al.: Improving software development management trough software project telemetry. In: *IEEE Software*, Vol. 22, No. 4, (2005), 76-85
3. LaToza, T.D., Venolia, G., DeLine, R.: Maintaining mental models: a study of developer work habits. In: *ICSE '06: Proceeding of the 28th international conference on Software engineering*, New York, (2006)
4. Li, K., Deugo, D.: Eclipse Project and Resource Tracking Plug-in. In: *2008 Ninth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing,* (2008), 820-827
5. Maalej, W., Happel, H.J.: From work to word: How do software developers describe their work? In: *6th IEEE International Working Conference on Mining Software Repositories,* (2009), 121-130
6. Sommerville, I.: *Software Engineering,* Addison-Wesley, 8th edition, (2006), 92-113

Annotation

Monitoring of tasks in software project and the regarding problems

This essay deals with tasks monitoring in software project. Problems occur when task reports are deprived or insufficient. Essay decribes aftermaths resulting from these problems and analyses how they affect the progress of software project from the aspect of further planning and controlling. It is shown how it is possible to utilize informations acquired with task monitoring for the whole project or individual developers self management. To eliminate these problems it is necessary to find efficient solutions. Some effective tools are briefly introduced, which where proposed to solve these kind of problems.