Bez podpory niet úspory

STANISLAV JURSKÝ

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

Abstrakt. Každý dobrý majster si pred začiatkom svojej práce pripraví všetky potrebné nástroje. Dokonca aj počas nej tieto nástroje zdokonaľuje, zamieňa za lepšie a neuspokojí sa, kým nebude jeho dielo také, aké si ho predstavoval. Nástroje ako pomocné prostriedky rozširujú náš talent. Ak sa ich naučíme správne používať, tak sa stanú predĺžením našich rúk. Táto práca opisuje hlavne nástroje na sledovanie úloh v softvérovom projekte vo všeobecnosti. Popisuje takisto dôsledky pri ich používaní, a teda aj ich vplyv na riadenie tímu. Obsahuje však aj zopár prostriedkov, ktoré priamo s nimi nesúvisia, no sú základom pre väčšinu projektových tímov. Cieľom tejto eseje, je ponúknuť pohľad z druhej strany, zo strany sledovanej úlohy. Teda ako vplývajú tieto nástroje na jej vývoj a či ju v konečnom dôsledku skutočne urýchlia, ako by sa očakávalo. Esej ponúka rovnako aj niekoľko mojich zaujímavých skúseností pri práci s takýmito nástrojmi.

Základné nástroje

Podľa mňa akýkoľvek prostriedok na podporu tímu v konečnom dôsledku podporí nie len spracovávanú úlohu, ale aj celý projekt. Možno mi nebudete veriť, ale pre mňa sú najhlavnejšími nástrojmi pri vývoji softvéru vo veľkých tímoch tri veci. Keď sa na ne pozerám z iného ako softvérového uhla, mohol by som ich prirovnať k bežným veciam ako je *tabuľa, výplatná páska a rozumná hádka.*

Namiesto tabule si môžeme predstaviť akýkoľvek nástroj na vizualizáciu projektu. Mnohé z bežne dostupných nástrojov obsahujú túto funkcionalitu. Rozdiel je zväčša len pri práci s nimi, teda v zobrazení, exportovaní a pod. Odplatou za používanie nástroja je užitočná a k riadeniu projektu prospešná spätná väzba a výstupy z týchto programov. No a náhradou za hádku by mohol byť nástroj podporujúci tok informácií v tíme. Napríklad diskusie, príspevky, emaily, okamžité správy, ba dokonca aj anketový systém.

Manažment projektov softvérových a informačných systémov, október 2008, s. 1-8.

Poznaj svoj projekt

Aké aspekty projektu by sme mali sledovať, aby sme o ňom vedeli dostatočné množstvo informácií? Vhodné informácie pomáhajú riadiť projekt. Určujú, či sa projekt vyvíja správnym smerom alebo je potrebné aktivovať mechanizmy pre návrat do želaného stavu. Ako by mala asi vyzerať správna kontrola nad projektom? Predovšetkým by sme mali v každom okamihu presne vedieť jeho stav. Tento stav sa sleduje a musí sledovať všetkými nasledujúcimi pohľadmi [2]:

- cena projektu,
- plán,
- kvalita,
- funkcionalita.

Je dôležité sledovať všetky tieto pohľady súčasne. Je vylúčené niektorý z týchto faktorov vynechať, hoci aj na úkor ostatných. Ak by sme sledovali len niektoré z uvedených, tak by sa v iných faktoroch nahromadila nesledovaná práca. Výsledok bude práve opačný, ako by sme očakávali. Vynechali by sme kvalitu. Práca by bola hotová na čas ako-tak funkčná, no nekvalitná. Zistilo by sa to až príliš neskoro a prípadná oprava by zapríčinila zaostávanie za stanoveným plánom. Stalo by sa tak z projektu to, čo popisuje Hunt a Thomas ako teória rozbitého okna (angl. broken window theory) [1]. Ak necháte jedno rozbité okno na budove neopravené, vyvolá to v okolí dojem nezáujmu. Postupne sa rozbijú ďalšie okná. Budova sa poškodí až tak veľmi, že ju vlastník nebude schopný opraviť. Jeho ignorovanie problému dospeje do stavu, kedy sa problém stane realitou a všetko sa pomaly vytmne z pod kontroly.

Kvalitný manažment projektu a včasné odhalenie problémov je závislé práve od efektívneho sledovania vývoja projektu.

Ideálny nástroj

V nasledujúcej časti sa pokúsim opísať ideálny nástroj na podporu tímu pri sledovaní úloh z môjho pohľadu. Obsahuje zlúčenie funkcionalít viacerých prostriedkov, ale aj funkcionalitu nástoja, s ktorým mám osobné skúsenosti. Práve kvôli tomu ponúka rozšírený pohľad na možnosti nástrojov. Je to možné využiť, ak ho porovnáme s nástrojom, o ktorý máme potenciálny záujem. Ľahšie tak spozorujeme nepodporované možnosti.

Najvyššou prioritou je podľa mňa schopnosť zvládnuť evidenciu úloh. Názvoslovia pre úlohu sa líšia od programu k programu. Niekde sú to lístky (angl. tickects), inde zas riešené otázky (angl. issues). Tieto úlohy by mal vedieť aspoň rozdeliť a evidovať v atomických častiach ako:

- chyby v projekte,

- zmeny už dokončených funkcionalít,
- tvorba nových funkcionalít.

Mal by ponúknuť dostatočné množstvo atribútov opisujúcich detaily úloh, pričom niektoré polia by boli spoločné ako napríklad:

1. *Opis* danej úlohy, ktorý vytvára zadávateľ. Tu by sa uvádzalo očakávané a aktuálne správanie.
2. *Riešiteľ*. Dobré by bolo mať možnosť zadať aj viacerých riešiteľov, ktorým sa automaticky priradí táto úloha v prípade, že primárny riešiteľ nie je dostupný (má dovolenku a pod.), kde je ale nutné viesť evidenciu ich dostupnosti alebo ju vedieť získať určitým spôsobom.
3. Časť pre *diskusiu*. Tá môže byť rozdelená na internú a externú. Interná by slúžila pre členov tímu a externá by sa využila na komunikáciu so zákazníkom, prípadne so zadávateľom úlohy.
4. Rôzne *stavy*, do ktorých sa môže daná úloha dostať. Tu by bolo vhodné uviesť zoznam osôb projektu, ktoré môžu meniť aktuálny stav spolu s uvedením do akého stavu môžu úlohu preradiť. Jeden zo špeciálnych stavov by bolo zmrazenie úlohy, kde sa úloha nejaví ako aktívna, ale nie je ani doriešená.
5. Identifikácia *fázy projektu*, v ktorej má byť úloha vyriešená a v ktorej naozaj bola vyriešená.
6. *Váha* konkrétnej úlohy. Aspoň tri odstupňované váhy a jedna špeciálna, tzv. obmedzujúca, ktorá blokuje vývoj iných úloh či dokonca iných projektov.
7. *Dátum do* kedy má byť úloha hotová a *za ako dlho* sa naozaj vyriešila. Je to dôležité, ak sa projekt nachádza v časovom sklze. Vtedy sa neprihliada na váhu úlohy, ale na jej požadovaný dátum dokončenia.
8. Pole *pokroku* v riešení. Možné buď ako zostávajúci čas na jej vyriešenie alebo ako doriešená percentuálna časť. Nájde využitie pri neskoršej vizualizácii úloh.
9. *Vytvorenie ďalšej* úlohy, ktorá z nej vychádza. Nápomocné, ak úloha zasahuje do iných častí, ktoré sú iného typu alebo v inom projekte.
10. *Prílohy* súvisiace s danou úlohou. Ak ide o chybu, tak by sa jednalo o záznam z logov. Pri novej funkcionalite to bude analytický dokument, ktorý úlohu popisuje.

Samozrejmosťou by boli charakteristické polia podľa typu úlohy. Napríklad ak by sa jednalo o chybu v projekte, ktorá by chybou nebola, tak pri zmene stavu na odmietnutá, by sa vyplňovala položka dôvod odmietnutia. Toľko k evidencii úloh.

Produkty úloh

Výstupom dokončenia úloh je vo väčšine prípadov zdrojový kód. Výstupy je nutné niekde bezpečne uchovať. Ideálny nástroj preto obsahuje aj úložisko údajov. Ak

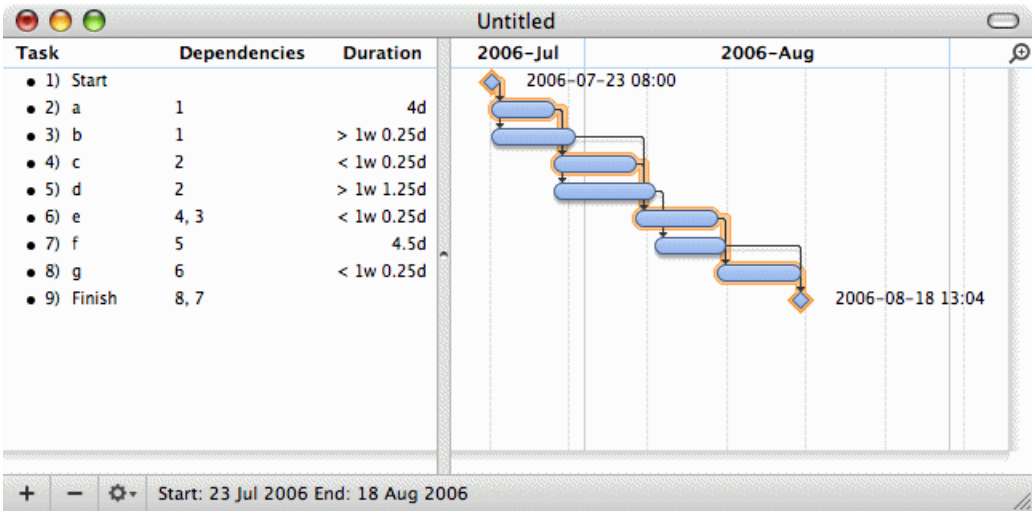
ho neobsahuje, tak sa dokáže integrovať na už existujúce úložisko. Výhodou je, ak úložisko dokáže monitorovať aj predchádzajúce verzie súborov. Poskytne to možnosť sledovať vykonané zmeny. Je potom možné získať informácie o tom kto, kedy akú zmenu vykonal. Odporúčam všetky výstupy úloh ukladať do úložiska. Dobre to vystihuje [1] kde uvádza, že vývoj nespočívajúci na zmenách závisí na uchovávaní. Tí ktorí si nepamätajú minulosť, sú odsúdení ju opakovať.

Prehľady

Ďalšiu vlastnosť, ktorú by ideálny nástroj mal používateľovi ponúknuť, je vizualizácia vyššie spomenutej evidencie a histórie úloh. Dali by sa zobraziť rôzne pohľady na projekt ako:

- v akej fáze sa projekt nachádza,
- kedy sa čo dokončí či predpokladané dokončenie projektu,
- aktivita jednotlivých členov tímu,
- podiel zadaných chýb oproti zamietnutým. Vhodné na zamyslenie sa nad prácou členov, ktorí riešia hľadanie chýb v projekte. Môže indikovať nutnosť ich preškolenia alebo nedostatočnej dokumentácie,
- zobrazenie mne pridelených úloh,
- zobrazenie hotových úloh pripravených na otestovanie, v prípade, že testovanie má na starosti iná samostatná skupina ľudí, ako tá ktorá úlohu vyriešila,
- vyriešené úlohy odovzdané zákazníkovi.

Na prehľad fáz projektu sa obvykle používa Ganttov diagram (pozri Obr. 1). Pozostáva z hierarchickej štruktúry úloh a časového diagramu. Horizontálne je zobrazené trvanie jednotlivých úloh a vertikálne ich následnosť, v akej by mali byť realizované. Dá sa z neho vyčítať aj závislosť vzťahov medzi jednotlivými aktivitami. Predovšetkým čo za čím nasleduje, teda čo je nutné doriešiť skôr ako sa začne pracovať na konkrétnej úlohe.



Obr. 1. Ganttov diagram v nástroji[1]

Pre zvýšenie informovanosti na projekte by nástroj posielal krátku správu pri akejkoľvek zmene atribútov úlohy členom vývojového tímu, ktorých sa úloha týka. Najvhodnejšia by bola emailová notifikácia. Uvedené prehľady tvoria neoddeliteľnú súčasť každého kvalitného podporného nástroja zameraného na sledovanie úloh.

Správna voľba

Prečo míňať peniaze na nevhodné veci, ktoré si myslíme, že nám pomôžu. Treba si správne vybrať a napokon naučiť sa správne používať daný nástroj na dosiahnutie zlepšenia výkonu tímu. Nástroje na podporu spoločného procesu by sme mohli rozdeliť napríklad podľa spôsobu komunikácie na synchrónne a asynchrónne [4]. Nie je však vylúčené ak nástroje podporujú oba spôsoby komunikácie. Asynchrónnu spoluprácu spoznáme podľa toho, že uchováva záznamy individuálnych aktivít. Príkladom na asynchrónne sú emaily, diskusie a fóra. Naproti tomu synchrónna spolupráca ponúka pohľad na aktivitu zúčastnených približne v reálnom čase. Vývoj spolupráce sa dá spozorovať pri tom, ako rýchlo vyriešia kolegovia nepochopenie či preskúmajú istý okruh otázok.

Existujú však aj nástroje, ktoré obsahujú veľké množstvo možností ako zdieľanie pracovného prostredia, uskutočňovanie konferencií, plánovanie udalostí, zdieľanie súborov, posielanie emailov či písanie správ. Treba si preto uvedomiť, ktoré z ponúkaných možností budú pre tím užitočné. Silu tímu môžeme rozdeliť do troch aspektov, ktoré nám lepšie pomôžu pochopiť potreby tímu [3]. Sú to:

– výkon použitého procesu,

[1] http://www.gantt-chart.biz/gcImages/Gantt_Diagram_Example.gif

- citlivosť na vzniknuté situácie,
- efektívnosť projektu.

Tu vidím súvislosť v tom, čo je potrebné sledovať podpornými nástrojmi. Ak má skupina skvelý výkon, stihne spraviť svoju prácu na čas a ušetrí tak peniaze. Čas napríklad ušetrí, ak nástroj podporuje plánovanie práce alebo ušetrí čas na cestovanie pri videokonferenciách a pod. Tieto funkcionality rovnako napomôžu pri vnímaní prostredia okolo projektu a zainteresovaným sa tak pomôžu lepšie zorientovať. No a väčšina z týchto vlastností zvyšuje efektívnosť projektu tým, že napomáha zbieraniu, spájaniu a zdieľaniu vedomostí jednotlivých osôb na projekte.

Rozdiely

Najväčšie rozdiely medzi nástrojmi sú spôsobené rôznorodosťou použitých metodík v projekte. Každý nástroj sa špecializuje na konkrétnu metódu vývoja. Ak sa zameriavame na agilné programovanie, kde sa používajú metodiky ako Scrum a extrémne programovanie, tak si vyberieme nástroj podporujúci inkrementálny vývoj a procesy vhodné pre iterácie s nutnosťou ich sledovania a riadenia. Medzi najvýznamnejšie rozdiely patria najmä:

- typ aplikácie,
- databáza,
- integrácia na softvér pre manažment verzií,
- možnosti exportu.

Typ aplikácie

Hlavným rozdielom je dostupnosť z webu alebo sa jedná o samostatnú aplikáciu. Ak si predstavím nie veľký tím, ktorého členovia sa nemajú možnosť každodenne stretávať, majú rôzny operačný systém, tak je pre nich určite výhodnejšie použiť webovú aplikáciu. Naopak pre pomerne veľký tím, pracujúci z jedného centrálneho miesta v rovnakých hodinách, používajúci služobné počítače je vhodná samostatná aplikácia. Pre nich je totiž podstatná rýchlosť a dostupnosť tohto nástroja.

Do používateľského rozhrania možno zahrnúť aj menej dôležité, ale užitočné vlastnosti. Príkladmi sú jeho prispôsobivosť, podpora emailov, RSS, príkazový riadok a pod.

Databáza

Myslím, že u by som sa rozhodoval podľa používateľského rozhrania. Ak zvládne celú prácu s databázou za používateľa a tá je pre neho transparentná, tak by som sa nezaoberal typom použitej databázy. Ak však prostredie vyžaduje aspoň minimálne znalosti databázy, napríklad pri tvorbe vlastných zobrazení, tak by som sa rozhodol podľa úrovne znalostí jednotlivých databáz v tíme. Najčastejšie voľby sú MySQL, SQL Server a Oracle.

Manažment verzií

Ak nástroj podporuje aj integráciu na manažovanie verzií, môže to len pomôcť pri riadení projektu. Napríklad dokážeme určiť, kde sa vykonali zmeny, čo bolo vytvorené a čo odobrané. Podobne ako pri databáze, aj tu by som vyberal podľa znalostí. Zaužívané sú predovšetkým SVN a Subversion. Ja osobne by som si zvolil Microsoft Source Safe, pretože mám s ním pozitívnu skúsenosť.

Možnosti exportu

Na čo mi je taká znalosť, s ktorou môžem narábať len na jednom mieste a nemôžem ju prenášať a výsledky graficky zobraziť alebo inak spracovať? Export je dôležitý. Môže sa napríklad jednať o dynamické generovanie dokumentácie, vykázanie vynaloženého úsilia pre zákazníka a pod. Zákazník tak vie presne za čo si zaplatil a my sme ušetrili čas na venovanie sa dôležitejším veciam.

Záver

Akýkoľvek správne použitý podporný nástroj je prínosom pre tím, ktorý s ním pracuje. Poskytuje praktické, elegantné a jednoduché riešenie pri riadení projektu. Domnievam sa, že sú dokonca nutnosťou pre efektívne zvládnutie manažovania projektu.

Správnemu výberu nástroja treba však venovať viac času, ako by sa mohlo zdať na prvý pohľad. Používať ho totiž bude celý tím. Netreba túto fázu nijako podceniť. Dobrovoľne nevhodný nástroj je najhorší nástroj. V práci som poukázal na niekoľko faktorov, podľa ktorých je možné výber uskutočniť. To napomôže zúženiu okruhu nástrojov, medzi ktorými sa rozhodujeme. Je potrebné mať na pamäti aj zaškolenie tímu, aby sa dokázal rýchlejšie stotožniť s vybraným nástrojom.

Nakoniec tak ako aj o reálny nástroj zo života, aj o tento sa treba starať, vylepšovať ho a získať z neho čo najviac. Na oplátku on potom prinesie členom tímu ich zaslúžené ovocie. Ak ho ale zanedbáme, projekt zapadne prachom a začnú sa objavovať jeho „rozbité okná“.

Použitá literatúra

1. Hunt, A., Thomas, D.: *The Pragmatic Programmer: From Journeyman to Master,* Addison-Wesley Professional, 1999.
2. McConnell, S.: The Software Manager's Toolkit, *IEEE Software*, vol. 17, no. 4, 2000, 5-7.
3. Pavlou, P.A., Dimoka, A., Housel, T.J.: Effective Use of Collaborative IT Tools: Nature, Antecedents, and Consequences, In: *Proceedings of the 41st Annual Hawaii International Conference on System Sciences,* HICSS (2008).
4. Rhyne, J. R. and Wolf, C. G.: Tools for supporting the collaborative process. In: *Proceedings of the 5th Annual ACM Symposium on User interface Software and Technology,* UIST '92. ACM, Monteray, California, United States (1992).

Annotation

No support, no savings

Every skilled master starts his job with preparing all necessary tools. Even during the work he tries to improve these tools, changes them to better ones until he is satisfied with his work. Instruments such as support tools extend our talent. They will become like an extension of our hands if we learn to use them correctly. This paper generally describes the main tools in monitoring tasks in software project development, but particularly in terms of their impact on usage, in essence, it describes team management for their use. However, it contains some handful tools that do not relate directly to them, but there is the basis for most project teams. The aim of this essay is to offer a view from the other side, the side of the scheduled task. So how these tools effects task evolution and whether they will actually speed up task, as would be expected. Article offers several my interesting experience in working with instruments of that kind as well.