Manažment konfliktov medzi programátormi a testermi

ĽUBOŠ LEČKO

Slovenská technická univerzita
Fakulta informatiky a informačných technológií
Ilkovičova 3, 842 16 Bratislava

Abstrakt. Konflikty sú bežnou súčasťou ľudského života a nevyhýbajú sa ani softvérovým projektom. Najčastejším dôvodom konfliktu býva nedostatok nejakých zdrojov. Uvedené sú najčastejšie zdroje konfliktov medzi členmi vývojárskych tímov spolu s potrebným pozadím. Rozobraté sú konflikty z rôznych pohľadov, vplyvy pracovného prostredia na atmosféru a vzťahy v tíme. Práca obsahuje informácie o agilných metódach vývoja softvéru, z ktorých priamo alebo nepriamo vyplývajú možnosti a potreby testovania, a s nimi súvisiace prístupy a opatrenia na riešenie konfliktu. Spomenuté sú odporúčané manažérske prístupy k riešeniu konfliktov v štandardných aj agilných softvérových procesoch.

Úvod

Organizácie stále tvoria jednotlivci, a tí, či už chceme alebo nie, prinášajú so sebou do spoločného prostredia rôzne zvyky, návyky, morálne, sociálne a náboženské hodnoty. V takomto mnohonázorovom prostredí preto stačí len malá iskierka na to, aby sa rozhorel konflikt, ktorý bude mať neodvratný dopad na všetko a všetkých, ktorých sa týka.

Konflikt - definícia

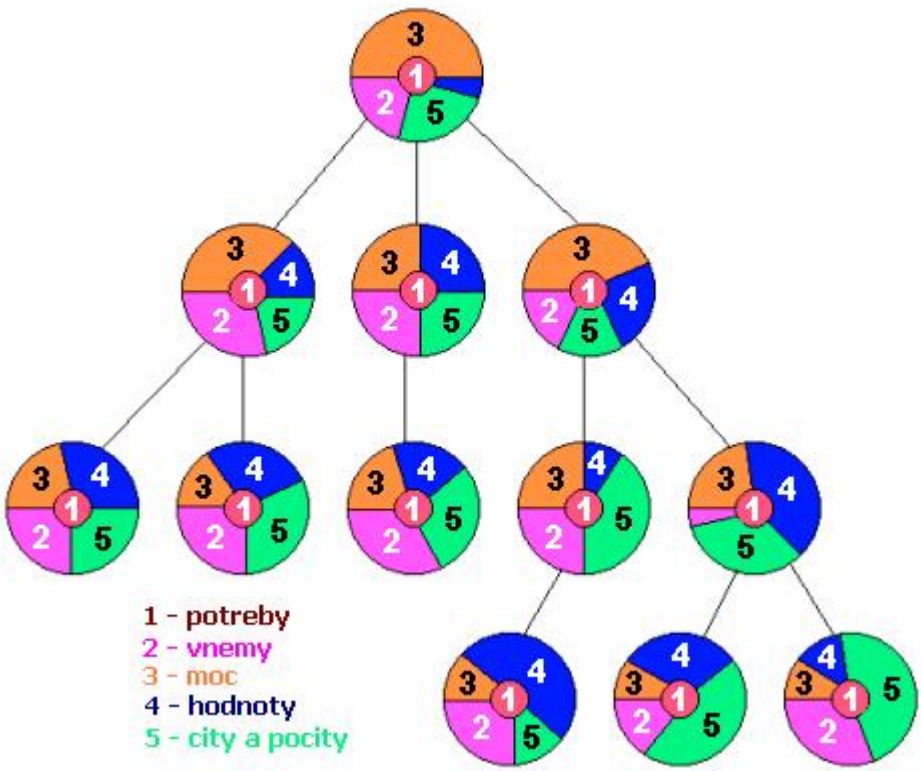
Konflikt je prirodzený nesúhlas vznikajúci medzi skupinami alebo jednotlivými osobami, ktoré sa líšia v postojoch, náboženstvách, hodnotách alebo potrebách. Zárodok tiež môže mať v sporoch z minulosti a personálnych rozdieloch. Ďalšie príčiny konfliktu zahŕňajú snahu o vyjednávanie v nepravú chvíľu alebo skôr než sú dostupné potrebné informácie[4].

Manažment v softvérovom inžinierstve, máj 2005, s. 1-7.

Prísady konfliktu

Konflikt ako taký je len dôsledok nerovnováhy medzi nasledovnými charakteristikami jedincov v organizácii – Obrázok 1:

- *Potreby* – všetko, čo je nevyhnutné k našej telesnej a duševnej pohode; ignorovanie vlastných potrieb, potrieb druhých ľudí alebo potrieb skupiny vedie ku konfliktu.
- *Vnemy* – ľudia vnímajú realitu rôzne, rozdielne vidia závažnosť, príčiny a dôsledky problémov.
- *Moc* – dôležitý vplyv na počet a typ konfliktov v závislosti od toho, ako ju ľudia definujú a najmä používajú; ovplyvňuje aj to, ako je konflikt zmanažovaný.
- *Hodnoty* – všetko, čomu my prikladáme osobitý význam; konflikty vznikajú, keď ľudia majú nezlúčiteľné alebo nejasné hodnoty.
- *City a pocity* – veľký počet ľudí necháva city a pocity ovplyvniť ich vysporiadanie sa s konfliktom.



Obrázok 1: Prísady konfliktu a možné rozloženie ich vplyvu v hierarchii riadenia organizácie.

Krúžky na Obrázok 1 predstavujú jednotlivcov v organizácií, prepojenia krúžkov pracovné vzťahy medzi nimi. Každý krúžok má iné delenie vnútornej plochy, rovnako ako je iný pomer prísad v každom zamestnancovi.

Každý zamestnanec si z každej uvedenej prísady prináša rôzne množstvo, čo sa prejaví na jeho individuálnej povahe a prístupe. Keby všetci zamestnanci boli rovnocenní z pohľadu vzájomných pomerov jednotlivých prísad, určite by dochádzalo k menšiemu množstvu konfliktov. Na základe starej známej pravdy – oko za oko, zub za zub – každý by sa snažil vyhnúť konfliktu, lebo vie, že rovnako sa zachová celé jeho okolie. Samozrejme môžeme použiť aj úvahu, že každý by sa snažil okoliu čo najviac robiť napriek, pretože rovnako by sa okolie správalo k nemu, lenže potom by takýto systém asi nedosiahol ciele, za účelom ktorých bol vytvorený.

Konflikty v softvérových tímoch

Vývoj softvéru, tak ako ostatné ľudské činnosti, obsahuje veľa potencionálnych zdrojov konfliktu, ktoré môžu poškodiť výkonnosť činnosti. Rovnako ako môže vzplanúť konflikt medzi vývojármi a používateľmi, môže sa objaviť aj vo vnútri tímu, najčastejšie medzi vývojármi a testermi.

Pretože konflikt často ovplyvní pracovnú výkonnosť a kvalitu výsledného produktu, identifikácia jeho zdrojov je kritická nielen v súkromnom sektore, ale aj na akademickej pôde. Na základe informácií uvedených v [1] možno na konflikt v softvérových procesoch nahliadať v troch rovinách ilustrovaných na Obrázok 2:

- Proces
- Ľudia
- Organizačná štruktúra

Obrázok 2: Roviny konfliktu[1].

Konflikt z pohľadu procesu

Aj keď nedostatok času nie je unikátnym javom v softvérových procesoch, najčastejšie spomínaným zdrojom konfliktu podľa programátorov a manažérov je rozdelenie času medzi fázy vývoja a testovania. Organizácie sa snažia dodať na trh komplikované, bezchybné produkty a čas sa jednoducho stáva menej prístupným a teda viac hodnotným. Testovanie je často oneskorené a jeho dĺžka skrátená za účelom dodržania termínov dodávky.

Testeri a programátori súperia navzájom o čas na splnenie svojich záväzkov, výsledkom čoho je konflikt. Vďaka prirodzenej následnosti testovania po programovaní testerom často zostáva málo času. Testeri vidia najväčší problém vo fakte, že manažéri nechávajú programátorov produkovať kód až do posledného momentu, často až do večera pred finálnou dodávkou. A keďže programátori nie sú tí, ktorých starosťou je dodať finálny produkt zákazníkovi, pri pohľade na harmonogram nepociťujú potrebu rýchlo dokončiť to, s čím už meškajú. Skrátka ako keby nemali rovnaký zmysel pre povinnosť. A čím viac sa implementácia oneskoruje, tým viac rastie napätie medzi programátormi a testermi.

Ako ďalší dôvod na konflikt sa uvádza rozdielne chápanie požiadaviek. Zatiaľ čo testeri bývajú viac zameraní na požiadavky používateľov, programátori upriamujú svoju pozornosť na technické aspekty výsledného produktu. Napriek tomu, že určite ide o dôležité pohľady na proces, výsledkom nemusí byť len lepšie pochopenie a zvládnutie problémovej oblasti, ale aj veľa problémov.

Čas sám osebe je požiadavkou na efektívne testovanie. Aj keď niektorí testeri očakávajú nedostatok času na výkon svojej profesie, stále to môže vyvolať stres a konflikt. Skrátenie času na testovanie interpretujú tak, že ich práca nie je dostatočne rešpektovaná. Manažéri musia pre každú fázu softvérového procesu naplánovať dostatočne dlhý čas a neuberať z neho, ak nejaká skupina nestihne svoje termíny.

Konflikt často vznikne, ak programátori a testeri nezdieľajú rovnaké pracovné ciele. Programátori sú zamestnávaní najmä preto, že nemajú problém vymýšľať nové riešenia. V záujme ich realizácie nie vždy dodržia používateľské požiadavky. A to je tŕňom v oku testerov, ktorých úloha v tíme je jednak overiť funkčnosť a kvalitu riešení od programátorov, a jednak čiastočne striehnuť ne dodržiavanie používateľských požiadaviek. Z toho vyplýva, že je potrebné jasne definovať skupinové a individuálne ciele pre potreby vytvorenia softvéru, ktorý sa na trh dostane načas a súčasne bude kvalitný.

Konflikt z pohľadu ľudí

Základným stavebným kameňom organizácií sú stále ľudia – individualisti, z ktorých každý prináša individuálny prístup, vnímanie problémov, mentálne pochody atď.

Testeri samých seba zväčša opisujú ako puntičkárov, pedantov, ktorí sú schopní vytvoriť súdržnú skupinu. Programátori sú manažérmi opisovaní ako kreatívni, temperamentní individualisti. Aj keď každá táto charakteristika jednotlivým profesiám len prospieva, v rámci tímu sa môže prejaviť ako rušivý element.

Podľa vyjadrení testerov programátori napíšu kód aplikácie a pre vlastné potreby ho otestujú spôsobom, akým by sa mala aplikácia používať[1]. Myšlienkové pochody programátora sú často veľmi odlišné od pochodov testera, týka sa to aj ich rozdielneho chápania potreby testov. Testeri teda program otestujú spôsobom, akým sa aplikácia bude používať, čo je pre programátorov ťažko pochopiteľná záležitosť. Môže to byť sčasti spôsobené aj faktom, že niektorí programátori vnímaju svoj kód ako predĺženie svojej osobnosti a v ňom objavenú chybu chápu osobne. Konflikt naberá na intenzite so vzrastajúcim počtom odhalených chýb.

Aj keď niektorí testeri sa postupom času naučili pomocou metódy pokus-omyl komunikovať s programátormi, keď očakávajú konflikt, tento predpoklad nemožno použiť na všetkých testerov. Stále častejšie si manažment organizácií uvedomuje, že trénovanie členov tímu v komunikačných a psychologických zručnostiach prispieva k zníženiu rizika vzniku konfliktov, a tiež k ich rýchlejšiemu riešeniu.

Konflikt často vychádza z faktu, že programátori a testeri nevnímajú softvérový proces rovnako. No čo je ešte dôležitejšie, svoje rozdielne pohľady si neuvedomujú. Organizované pracovné aktivity, ktorých úlohou je oboznámiť programátorov a testerov s podstatou práce druhej strany, pomáhajú v prevencii pred problémami.

Konflikt z pohľadu organizačnej štruktúry

Aj keď väčšina organizácií si uvedomuje, že potrebujú vysoko kvalifikovaných testerov a ich zručnosti, testeri stále zápasia o zisk zaslúženého rešpektu, o rešpekt porovnateľný s programátormi.O veľkosti ega niektorých programátorov hovorí výrok:“Ak by ste mali diagram s Bohom na vrchu, programátori by svoju pozíciu umiestnili nad neho.“ Aj vďaka tomu musia testeri dennodenne vyvíjať úsilie na to, aby si udržali rešpekt porovnateľný s programátormi. Nedostatok podpory zo strany manažérov robí prácu testera náročnejšou z fyzickej, psychickej a časovej stránky.

Tento zápas im nijako neuľahčuje ani pokrok v komunikačných technológiách, keď problémy, ktoré by vyriešili priamou komunikáciou za 10 minút, riešia cez mail alebo fax celý deň.

Softvérové metódy a ich riešenie konfliktu

V súčasnosti najrozšírenejšími metódami vývoja softvéru sú tie plánom riadené, v posledných rokoch sa však stále viac presadzujú agilné metódy. Stále je však ťažké určiť, či je lepšie, jednoduchšie alebo lacnejšie plánovať alebo prerábať.

Plánom riadené metódy

Čistý plánom riadený vývoj softvéru predpokladá, že architektúra, používateľské požiadavky a celkový dizajn systému môžu byť vytvorené už v skorých fázach procesu a implementácia je ponechaná na neskoršie fázy.

Metódy využívajúce plánovanie sú náchylnejšie na nedodržanie termínov, stačí, aby sa oneskorila čo i len jedna fáza. Postupy, ako zabrániť konfliktu alebo ho riešiť, zahŕňajú:

- Zlepšenie komunikácie v rámci tímu – zamestnanci musia byť už od začiatku trénovaní v riešení konfliktov, rozvíjaní svojich komunikačných schopností a využívaní neformálnych sprostredkovacích techník.
- Využitie roly nestranného arbitra – ak konflikt prerastie do neúnosných rozmerov, neodkladne ho treba riešiť. Nestranný arbiter je nie príliš často používanou možnosťou na jeho riešenie. Vypočuje obe strany, predloží návrhy riešenia a vyjedná zmierlivé riešenie.
- Podpora tímových aktivít – organizácia z režijných nákladov financuje aktivity pre zamestnancov vo voľnom čase; zamestnanci si na seba rýchlejšie zvyknú a komunikácia medzi nimi prejde do menej strohej a formálnej roviny, menej formálna komunikácia často zrýchli riešenie problémov.
- Ekvivalentný prístup k testerom v porovnaní s programátormi.

Agilné metódy

Stratégie agilných metód vývoja softvéru sa usilujú zredukovať cenu neustálych zmien pomocou vývoja jednoduchých riešení, pravidelných zmien návrhu a stáleho testovania. Medzi najvýznamnejšie môžeme zaradiť:

XP – Extreme Programming

Spomedzi všetkých agilných metód si vyslúžila najviac pozornosti. Ide o proces založený na experimentovaní a vylepšovaní, na základe používateľských požiadaviek sa stanoví séria testov, ktorú musí implementácia splniť, aby bola považovaná za riešenie.

Scrum

Iteratívny, inkrementálny proces. Výsledkom každej iterácie je časť funkcionality, ktorá môže byť eventuálne samostatne expedovaná k zákazníkovi. Dĺžka iterácie je 30 dní[5].

Feature Driven Development

Proces navrhnutý najmä pre objektovo orientovaný prístup. Výhodou sú krátke iterácie v dĺžke 14 dní, ktorých úlohou je rýchlo doručiť hmatateľnú funkcionalitu zákazníkovi.

Spoločnou vlastnosťou agilných metód je skrátenie dodacích cyklov, čím sa ale tiež skracuje doba, počas ktorej je možné vytvorené riešenie testovať. Preto manažéri, ktorých organizácie používajú niektorú z agilných metód, musia s ešte väčším dôrazom podporovať testovaciu fázu projektu a vyhýbať sa konfliktom. Napríklad *XP* používa na obranu pred konfliktmi v tíme kolektívne vlastníctvo kódu. Povzbudzuje každého člena tímu prispieť k procesu novými nápadmi, nech už ide o fázu implementácie alebo testovania. Ak každý člen tímu má rovnaký podiel snahy na výslednom produkte,

zaslúži si aj rovnaký diel úcty a rešpektu = účinné predchádzanie konfliktu. Tímy bývajú tvorené menším počtom ľudí, čím sa v porovnaní s ostatnými metódami vytvorí lepšie prostredie na neformálnu komunikáciu, zlepší prehľad o možnostiach a schopnostiach jednotlivých členov. Vylepšovanie kódu, ktorého vlastníkom je každý člen tímu, je spoločný cieľ a stanovenie spoločného cieľa pre programátorov a testerov predchádza konfliktom.

Záver

Na konflikt medzi programátormi a testermi možno nazerať v troch rovinách, všetky sú špecifické pre prostredie vývojárskych tímov. Jednotlivci v pozíciách programátorov a testerov niekedy majú problém spolu vychádzať. V skupine ľudí to nie je nič neočakávané. Nepriateľská povaha testovania softvéru a rozdiely medzi jeho jednotlivými účastníkmi sú istou zárukou konfliktov. Úlohou manažérov je týmto konfliktom predchádzať, na výber majú široké spektrum metód, sprostredkovaných, manažovaných aj nemanažovaných. Samozrejme je potrebné konflikty riešiť tak, aby ich dopad na projekty bol minimálny. Záleží len od ich skúseností, prostriedkov alebo zvolenej metódy, aký výsledok riešenia konfliktu sa dostaví.

Použitá literatúra

1. Cynthia F. Cohen, Stanley J. Birkin, Monica J. Garfield, Harold W. Webb: Managing Conflict in Software Testing. In Communications of the ACM, Vol. 47, No. 1 (January 2004), 76-81.
2. Martin Fowler: The New Methodology, http://www.martinfowler.com/articles/newMethodology.html, 7.5.2005
3. businesslistening.com: A Simple Process for Resolving Business Conflicts, http://www.businesslistening.com/conflict_resolution-2.php, 7.5.2005
4. Managing conflict, http://www.ctic.purdue.edu/KYW/Brochures/ManageConflict.html, 7.5.2005
5. Scrum, http://www.controlchaos.com/about/, 7.5.2005

Annotation

Managing conflicts between developers and testers

Conflicts are natural part of human life, including software projects. Lack of resources is the most frequent source of conflict. The most frequent sources of conflict among development teams are given including necessary background. Various layers of conflict are considered. The paper contains information on agile methods of software development with testing requirements and conflict resolving precautions. Management methods in standard and agile processes are given.