MONITOROVANIE SOFTVÉROVÉHO PROJEKTU NA ZÁKLADE JEHO TYPU

Pridanie pracovnej sily do meškajúceho projektu ho robí ešte viac meškajúcim.

Roman Kováč

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

Abstrakt. *Monitorovanie softvérového projektu je činnosť, ktorá zabezpečuje kontrolu nad priebehom samotného projektu a môže včas odhaliť možné problémy, ktoré môžu v rámci tohto procesu vzniknúť. Softvérové projekty sú však často rôznorodé, zložitosť ich monitorovania sa v mnohých prípadoch odlišuje, a tak boli v uplynulých rokoch vypracované rôzne metodológie na ich monitorovanie. Príkladom rôznorodosti softvérových projektov je open-source projekt na jednej strane a klasický projekt na strane druhej, kde je zrejmé, že ich monitorovanie bude prebiehať rozdielne. Táto esej poskytuje pohľad na rôzne prístupy k monitorovaniu softvérových projektov v závislosti od typu projektu, ktorý chceme monitorovať, poukazuje na faktory ovplyvňujúce tento proces a konfrontuje vybraný prístup s možnými problémami, ktoré môžu pri danom type projektu nastať.*

Kľúčové slová: *softvérový projekt, monitorovanie, outsourcing, open-source*

Úvod

Jednou z kľúčových činností riadenia softvérového projektu je sledovanie jeho priebehu. Aj napriek najnovším metodológiám vývoja softvéru, 80% informačných systémov prekročí svoj plánovaný rozpočet, prípadne nedodrží dohodnutý časový harmonogram

[1]. Práve vďaka výberu vhodnej metódy monitorovania projektu sme schopní objektívne zachytiť progres projektu, a tak včas odhaliť možné problémy, ktoré môžu v rámci tohto procesu vzniknúť. Ako však správne zvoliť túto metódu?

Počas uplynulých rokov vzniklo niekoľko prístupov monitorovania softvérového projektu, ktoré sú viac, či menej vhodné pre daný typ projektu. Väčšina z nich sa zaoberá monitorovaním *klasických projektov*, kde sú členovia tímu v úzkom kontakte a môžu jednoducho referovať svoju vykonanú prácu. Projektový manažér má v takomto type projektu priamy dosah na členov tímu a môže jednoduchšie sledovať priebeh projektu.

Iným typom projekt je *outsourcovaný projekt*, kde jednotlivé časti softvéru vytvárajú rôzne tímy, ktoré sa nenachádzajú na tom istom pracovisku, dokonca sa nemusia nachádzať v tom istom štáte, či kontinente. Je zrejmé, že monitorovanie takýchto projektoch sa bude líšiť od monitorovania tých klasických.

Ďalším špecifickým typom projektu je *open-source projekt*. Často na tomto type projektu pracujú ľudia individuálne bez toho, aby ich činnosť niekto sledoval. Je však dôležité aj v tomto type projektu sledovať jeho priebeh, aby bolo možné včas odlíšiť bezproblémový projekt od projektu, ktorý je v ohrození.

V tejto práci opisujem jednotlivé typy projektov, problémy pri ich monitorovaní a riešenie týchto problémov vhodným výberom metódy na ich monitorovanie.

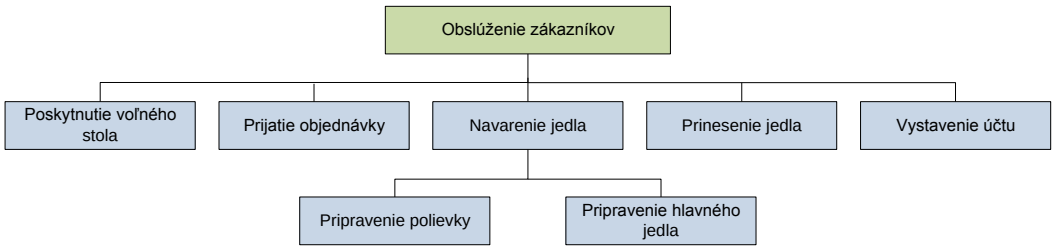
Monitorovanie klasických projektov

Už v úvode bolo naznačené, čo pod pojmom klasický projekt rozumieme. V našom prípade ide o softvérový projekt, na ktorom pracujú ľudia z jednej firmy, na jednom pracovisku, a ktorý riadi projektový manažér. Jednou z úloh projektového manažéra je aj monitorovanie projektu, za ktorý je zodpovedný. Takýto projekt sa javí ako bezproblémový na jeho monitorovanie, ale nie to však v mnohých prípadoch tak. Ako teda monitorovať takýto projekt? Jednou z možností je odhadovať stav projektu subjektívne. Každý z nás dokáže do určitej miery odhadnúť ako sa približne projektu, na ktorom pracuje darí a koľko percent práce je už hotovej, samozrejme, pokiaľ ide o malý projekt. Pokiaľ chceme objektívne monitorovať väčší, ale koniec koncov aj menší projekt, je vhodné zvoliť jednu z používaných metód.

Takouto metódou je *analýza získanej hodnoty* (Earned Value Analysis), ktorej cieľom je objektívne merať výkonnosť a progres projektu [2]. Poskytuje spôsob, ako merať skutočne vykonanú prácu, ako predpovedať náklady projektu a jeho čas dokončenia, určiť, ako sa projektu darí oproti pôvodnému plánu, a na základe tejto informácie predpovedať, ako sa mu bude dariť v budúcnosti. Dôležitou súčasťou tejto metódy je rozdelenie celej práce na projekte do menších úloh. Po tomto rozdelení je možné k jednotlivým úlohám priradiť ich *plánovanú hodnotu* (PH), čo predstavuje plánované výdavky, ktoré plánujeme minúť na danú úlohu. *Získaná hodnota* (ZH) predstavuje rozpočtované výdavky na prácu, ktorú sme už dokončili a *aktuálne výdavky* (AV) sú výdavky, ktoré sme už minuli. Na základe týchto parametrov je možné sledovať priebeh projektu a aj určiť, ako sa mu darí. V prípade AV>ZH vieme, že projekt prevyšuje svoj rozpočet a je potrebné vykonať opatrenia, ktoré tento stav napravia [3].

Dôležitosť rozdelenia úloh na menšie podúlohy zobrazuje aj nasledujúci obrázok (Obr. 1). Predstavme si, že sa nachádzame v reštaurácií a ideme obslúžiť zákazníkov.

V prípade, že si prácu nerozdelíme na menšie podúlohy, môže byť zložité a často aj nepresné odhadnúť, koľko nám celá úloha bude trvať.



Obr. 1. Správne rozdelenie úlohy na podúlohy zabezpečí ich jednoduchšie ohodnotenie.

Čím lepšie si teda veľkú úlohu rozdelíme na menšie, tým presnejšie budeme schopní monitorovať priebeh a stav projektu. Monitorovanie takéhoto typu projektu však zjednodušuje fakt, že projektový manažér projektu má priamy dosah na členov tímu, ktorí mu jednoducho môžu referovať postup na svojich úlohách. V prípade objavených problémov je projektový manažér schopný zasiahnuť do prebiehajúceho projektu a prijať opatrenia, ktoré problém napravia. V každom type projektov sú však základom pre monitorovanie samotní ľudia a ich objektívne hodnotenie stavu úloh, na ktorých pracujú. Toto považujem aj za najväčšie riziko v prípade, že účastníci projektu neinformujú o stave ich úloh objektívne a vedome zamlčujú problémy, ktoré pri ich plnení nastávajú.

Aby sa takýmto rizikám predišlo, boli vypracované rôzne metriky, ktorých cieľom je merať vykonanú prácu. Tradičnou metrikou je *počet riadkov kódu*, ktorá však nie je úplne dokonalá. Je potrebné si uvedomiť, že vytváranie softvéru nie je iba o písaní kódu, ale aj o návrhu riešenia, testovaní, či písaní dokumentácie. Ďalšou metrikou je *analýza funkčných bodov*, ktorú je možné použiť v čase vytvárania a analýzy požiadaviek. Na ňu nadväzuje metóda *bodov prípadov použitia*, ktorá predpokladá použitie prípadov použitia pri návrhu riešenia a tieto prípady použitia analyzuje, pričom berie do úvahy hráčov v prípadoch použitia, technické a iné faktory, ktoré nakoniec abstrahuje do rovnice.

Použitím kombinácie metód analýzy získanej hodnoty a bodov prípadov použitia sme schopní dostatočne presne monitorovať klasický softvérový projekt a včas odhaliť vznikajúce problémy.

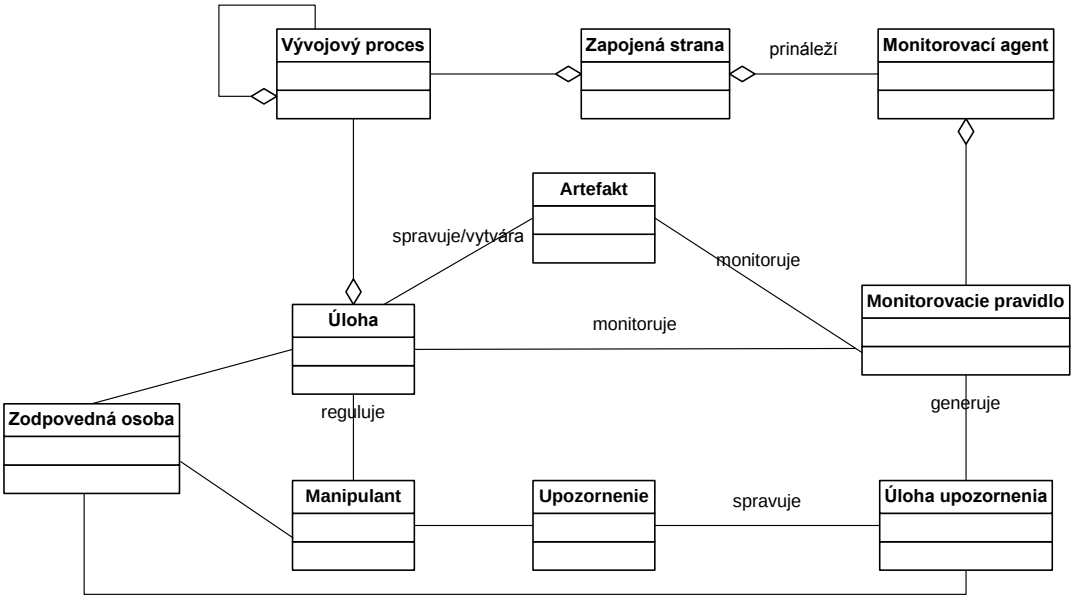
Monitorovanie outsourcovaných projektov

Predstavme si teraz, že do softvérového projektu vstupuje viacero strán, ktorých organizácia môže byť v rôznej miere rozličná. Každá strana môže navyše zasahovať do rozličných fáz softvérového projektu. V takomto prípade hovoríme o outsourcovaných projektoch, ktorých monitorovanie si vyžaduje mierne odlišný prístup ako tomu bolo pri klasických, spomenutých v predchádzajúcej kapitole. Dôvod tohto odlišného prístupu spočíva najmä v tom, že projektový manažér nemá priamy dosah na členov tímu a v určitých fázach ani nemá k dispozícii výsledky jednotlivých strán, ktoré sú do projektu zapojené. Navyše úlohy pre jednu zúčastnenú stranu môžu ovplyvňovať úlohy pre tú druhú, takže monitorovanie stavu úloh je v tomto type projektu veľmi dôležité.

Zúčastnené strany môžu svoje úlohy monitorovať rovnako ako to bolo pri klasických projektoch, avšak vyžaduje sa tu istá úroveň synchronizácie.

Z vlastnej skúsenosti viem, že sa často rieši telefonátmi, prípadne mailmi, v ktorých sa jedna strana informuje o stave úloh pre druhú stranu. Je jasné, že takýto spôsob monitorovania projektu nie je najideálnejší a pre väčší projekt nepostačujúci. Z tohto dôvodu je vhodné sa zamyslieť nad akýmsi centrálnym riešením, kde by jednotlivé strany mali prístup a mohli jednoducho meniť stav svojich úloh, pričom zodpovedná strana za samotný projekt by bola o týchto zmenách stavu informovaná a mohla by adekvátne zareagovať.

Obr. 2 zobrazuje zjednodušený model riešenia, ktoré bolo navrhnuté práve pre monitorovanie outsourcovaných projektov, do ktorých je zapojených viacero strán počas ktorýchkoľvek fáz vývojového procesu.



Obr. 2. Model monitorovacieho systému, kde každej zapojenej strane prináleží monitorovací agent s monitorovacími pravidlami (upravené z [4]).

V takomto systéme sú jednotlivým zapojeným stranám priradené konkrétne fázy vývojového procesu a k nim priradené úlohy, ktorých cieľom je spravovať alebo vytvárať artefakty [4]. Dôležitou súčasťou tohto systému sú monitorovací agenti pre každú zúčastnenú stranu, ktorí presadzujú vytvorené monitorovacie pravidlá. V prípade, že nastane situácia, ktorá si vyžaduje upozornenie, vygeneruje sa úloha s upozornením.

Pre ilustráciu takéhoto riešenia je vhodné uviesť konkrétny príklad. Predstavme si, že jedna zúčastnená strana má za úlohu navrhnúť rozhranie (úloha č. 1), ktoré bude následne druhá strana implementovať (úloha č. 2). Existujú teda dve úlohy pre dve zapojené strany, pričom jedna z nich je závislá od tej druhej. Z tohto dôvodu bude vytvorené monitorovacie pravidlo, ktoré bude sledovať stav úlohy č. 1. V prípade, že sa úloha č. 1 nesplní do požadovaného termínu, vygeneruje monitorovacie pravidlo upozornenie do centrálneho

systému, ktoré ovplyvní aj úlohu č. 2. Takéto riešenie je veľmi efektívne a zabezpečuje prehľad zúčastnených strán o stave úloh, ktoré sa ich týkajú.

Je potrebné si však uvedomiť, že vytváranie úloh a monitorovacích pravidiel si bude vyžadovať dodatočné úsilie, ktoré budú musieť zúčastnené strany vynaložiť. Ak si to však porovnávame so stavom, kedy jednotlivé strany boli oboznámené s problémom telefonicky, prípadne niektoré strany o probléme neboli oboznámené vôbec, je určite na mieste sa nad takýmto spôsobom monitorovania projektu zamyslieť. Z môjho pohľadu je takéto riešenie vhodné pre veľké projekty, v ktorých sú zúčastnené aspoň tri strany a ich monitoring iba prostredníctvom písomnej, či ústnej komunikácie by bol málo efektívny.

Monitorovanie open-source projektov

Špecifickou kategóriou softvérových projektov sú projekty, na ktorých pracujú rôzne komunity ľudí, prípadne individuálni vývojári. V prípade takéhoto typu projektov by bola namieste otázka, prečo vôbec monitorovať takýto projekt?

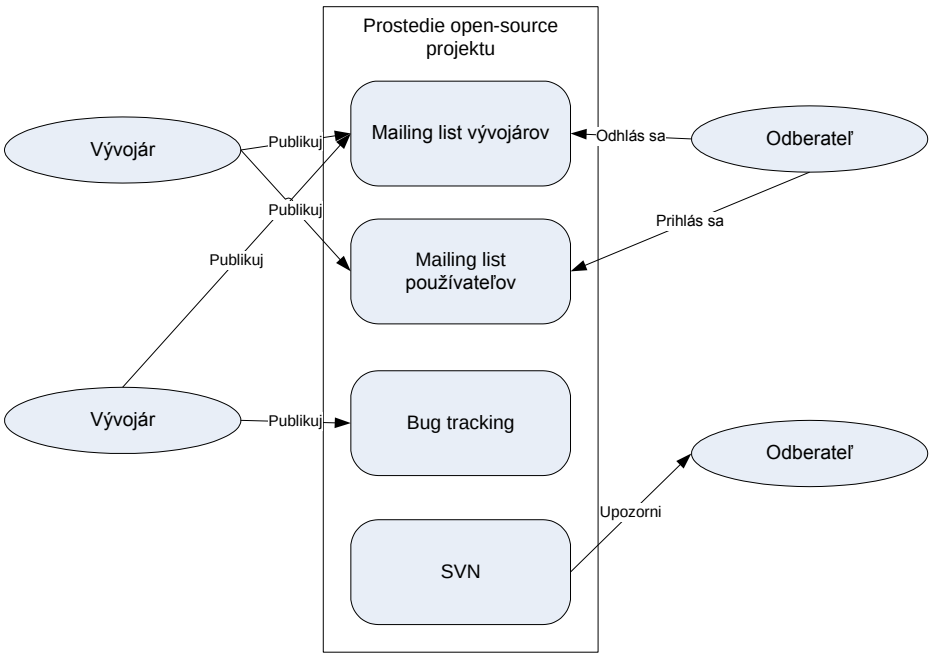
Existujú open-source projekty, ktoré majú rovnako ako klasické projekty svoj rozpočet a termíny, do ktorých musia byť jednotlivé časti projektu dokončené. Investori teda musia akýmsi spôsobom rozlíšiť medzi projektom, ktorému sa darí a termíny bezproblémovo stíha a projektom, ktorý je riskantný a je dokonca v ohrození. Práve monitorovanie nám môže takéto dva projekty rozlíšiť, a preto je nevyhnuté zaviesť spôsob, ako je možné takýto špecifický projekt monitorovať. Open-source projekt je špecifický najmä v tom, že zainteresovaní ľudia sa často nepoznajú, zapájajú sa do projektu niekedy len dočasne, v iný čas a na inom mieste. Často je ich spolupráca odkázaná len na posielanie správ, dát a artefaktov cez rôzne komunikačné nástroje, vývojárske nástroje a podobne. Základné charakteristiky open-source projektu sú [5]:

- Väčšina úloh je vykonávaná neplatenými účastníkmi
- Vysoká miera obmeny účastníkov
- Nízka miera formálnosti: dizajn, plánovanie a manažment

Ako aj pri predchádzajúcich 2 typoch projektov, aj v tomto prípade sú najväčším rizikom pri projekte jednotliví účastníci, ktorých úlohou je objektívne oznamovať problémy, ktoré počas projektu vznikajú. Kým v predchádzajúcich dvoch typoch projektov sa činnosť jednotlivých účastníkov dala monitorovať, či už projektovým manažérom alebo inou zodpovednou osobou, v prípade open-source projektov je takéto monitorovanie veľmi náročné. Existujú však indikátory, na základe ktorých je možné posúdiť stav open-source projektu. Medzi takéto indikátory patria [5] :

- *Otvorené problémy, meškajúce služby* – chyby a problémy, ktoré sa nachádzajú v systéme pre ich správu, avšak opravy neboli vykonané do stanoveného termínu
- *Proporcie* - spočítajú sa príspevky v mailing liste, chyby za zvolené časové obdobie, prípadne zmeny v SVN systéme
- *Aktivita komunity a jej intezita* – počet stiahnutí v porovnaní s príspevkami v mailing liste, interakcia medzi vývojármi a pod.

Pre monitorovanie týchto indikátorov si je potrebné najprv vytvoriť model open-source projektu. Jedným z najlepších sa javí prístup, ktorý modeluje open-source projekt ako multiagentný udalostný systém. Účastníci sú v takomto modeli agenti a interakcia medzi nimi predstavuje udalosti. Takýto model projektu zachycuje Obr. 3.



Obr. 3. Open-source projekt môžeme chápať ako multiagentný udalostný systém (upravené z [5]).

Aby mal takýto model a jeho monitorovanie vôbec zmysel, je potrebné nejakým spôsobom merať získané dáta a vytvoriť z nich informácie, ktoré sú zmysluplné pre niekoho, kto o projekte rozhoduje. Základom je zachytiť udalosť, ktorá vznikla a identifikovať ju. Následne je potrebné analyzovať získané udalosti a vytvoriť informácie, ktoré sú potrebné pre následné rozhodovanie.

Najideálnejšie bude spôsob monitorovania open-source projektu predstaviť na príklade. Ako bolo spomenuté, jeden z indikátorov stavu projektu je aktivita vývojárov v mailing liste, ktorý predstavuje akýsi virtuálny výbor projektu. Vytvoríme si teda definíciu monitorovania [5]:

- **Cieľ:** monitorovanie aktivity vývojárov v mailing liste z pohľadu projektového lídra
- **Otázka:** aká je aktuálna aktivita vývojárov v mailing liste?
- **Metrika:**
  - Priemerný počet mailov za mesiac. Táto metrika ukáže trend intenzity prispievania vývojárov. Pozitívny trend indikuje, že vývojári sú aktívni
  - Percento prípadov, kedy počet mailov za mesiac klesne pod dolný limit. Táto metrika indikuje, že je nízka aktivita vývojárov

Ak už máme vytvorenú definíciu monitorovania, ďalšou dôležitou fázou je identifikovanie udalostí, ktoré vychádzajú zo zvolených metrík. Luckam [6] predstavil *agenta pre spracovanie udalostí*, ktorý pozostáva z troch základných krokov spracovania udalosti:

1. Identifikovanie vstupnej akcie, čo je v našom prípade udalosť prijatý mail
2. Identifikovanie výstupnej akcie. V našom prípade pôjde o upozornenie, že aktivita vývojárov je pod určitým štandardom
3. Identifikovanie správania a pravidiel udalosti. U nás to bude predstavovať pravidlo, že ak počet mailov za mesiac neprekročí 60, vyvolaj upozornenie

Takýmto spôsobom sme schopní vytvoriť monitorovacie definície a vlastných agentov, ktorí nás budú upozorňovať o neželaných stavoch open-source projektu a pomôžu nám včas tieto problémy vyriešiť. Tento spôsob je z časti podobný spôsobu, ktorý bol opísaný pri outsourcovaných projektoch, kde bol tiež navrhnutý monitorovací agent s monitorovacími pravidlami a predstavuje podľa môjho názoru najideálnejšie riešenie monitorovania softvérových projektov tohto typu. Kritickou časťou takéhoto riešenia je návrh vhodných monitorovacích definícií a pravidiel, ktoré informujú o akejsi riskantnej situácií.

Záver

V tejto práci boli porovnané jednotlivé typy softvérových projektov a ich rôzne možnosti monitorovania. V prípade klasického projektu som dospel k názoru, že úspešnosť jeho monitorovania závisí predovšetkým od vhodne zvolenej metriky merania vykonanej práce. Dôležité je aj rozdelenie úloh na menšie podúlohy, ktorých monitorovanie a odhadovanie je jednoduchšie a presnejšie.

Pri outsourcovaných projektoch, kde sa jednotlivé úlohy pre zúčastnené strany môžu ovplyvňovať, je už vhodné zamyslieť sa nad zavedením monitorovacieho systému, ktorý by upozorňoval o vzniknutých problémoch pri plnení úlohy. Načrtol som tu existujúce riešenie, v ktorom hlavnú úlohu hral monitorovací agent, jeho monitorovacie pravidlá a upozornenia, ktoré tieto pravidlá generovali. Podľa môjho názoru sa takýto systém stáva efektívnym, keď sú zúčastnené aspoň tri strany a ich vzájomné monitorovanie iba prostredníctvom mailov či telefonátov, sa stáva ťažkopádnym.

Na záver som sa zameral na špecifickú kategóriu softvérových projektov, a to open-source projektov. Monitorovanie takýchto projektov sa veľmi podobá riešeniu outsourcovaných projektov avšak vyžaduje si udalostné chápanie projektu. Udalosti sú v tomto prípade interakcie medzi účastníkmi projektu, ktoré nám môžu indikovať stav projektu. Pre získanie relevantných informácií je potrebné vytvoriť monitorovacie definície a identifikovať správne udalosti, ktoré sú pre danú definíciu relevantné.

Použitá literatúra

1. Gordon, R. J., Gordon, R. S.: *Information Systems A Management Approach*, The Dryden Press, 1999.

2. Fleming, Q.W., Koppelman, J.M.: *Earned Value Project Management*. 3rd Edition, Project Management Institute, Atlanta, 2006.
3. Li, J., Ma, Z., Dong, H.: Monitoring Software Projects with Earned Value Analysis and Use Case Point. In: *Seventh IEEE/ACIS International Conference on Computer and Information Science (2008)*, 475-480.
4. Chan, V. T. C., Chiu, K.W.D.: e-Monitoring of Outsourcing IS Project in Financial Institutions: A Case Study on Mandatory Provident Fund Projects in Hong Kong. In: *IEEE International Conference on e-Business Engineering (2007)*, 460-465.
5. Wahyudin, D., Tjoa, A. M.: Event-Based Monitoring of Open Source Software Projects. *The Second International Conference on Availability, Reliability and Security (2007)*, 1108-1115.
6. Luckham, D.: *The Power of Event: An Introduction to Complex Event Processing in Distributed Enterprise System*, Addison Wesley, 2002.

Annotation

Monitoring software project on the bases of his type.

Monitoring software project is an activity that provides control over the project and can detect potential problems that may occur in this process. Software projects are often different, the complexity of their moniroting differs in many cases, so in the recent years, there have been created many methodologies for monitoring them. An example of the diversity of software projects is the open-source project on the one side and the classical project on the other side, where we can expecet that monitoring will be done differently. This essay provides an insight into different approaches to the monitoring software project depending on the type of project, indicates the factors influencing this process and compares the selected approach to possible problems that are likely to occur during the given type of project.