DOKÁŽEME MERAŤ SOFTVÉROVÚ SPOĽAHLIVOSŤ ?

Jednoduchosť je predpokladom spoľahlivosti.

Matej Kompánek

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

Abstrakt. *Pri monitorovaní vývoja softvérových systémov narážame na viacero problémov. Okrem tých zjavnejších ako zložitosť a nejasnosť požiadaviek je veľkým problémom práve softvérová neviditeľnosť. Pre hladký priebeh projektu je veľmi dôležité odhadnúť množstvo úsilia, ktoré je potrebné vyvinúť na vývoj v jednotlivých etapách. Práve v tejto problematike nám môžu napomôcť softvérové metriky. Softvérové metriky predstavujú kvantitatívnu metódu merania určitých vlastností softvéru a môžu byť obzvlášť nápomocné pri monitorovaní vývoja projektu. Existuje však mnoho metrík, ktoré často môžu podávať aj protirečivé výsledky. Je potrebné zamyslieť sa, ktoré z týchto metrík nám môžu ponúknuť dôveryhodné výsledky na to, aby sme mohli plánovať ďalší priebeh projektu. Táto esej sa zameriava na metriky týkajúce sa spoľahlivosti ako jednej z dôležitých vlastností softvéru.*

Kľúčové slová: *monitorovanie softvérového projektu, softvérové metriky, spoľahlivosť*

Potreba softvérových metrík

Potreba softvérových metrík ako kvantitatívnej metodiky merania určitých vlastností softvéru siaha do šesťdesiatych rokov minulého storočia do čias takzvanej „softvérovej krízy“. Toto obdobie bolo charakteristické rozvojom softvérového odvetvia, ako aj s tým spojených rizík ako je napríklad zvyšujúca sa komplexnosť vytváraného softvéru. S tým súvisí aj mnoho iných problémov ako potreba sofistikovanejšieho modelu získavania požiadaviek, odhadovania nákladov alebo údržby systému.

Proces vývoja softvéru čelil v dobe softvérovej krízy trom základným problémom [2]:

- Veľmi hrubé odhadovanie nákladov na vývoj a nepresné plánovanie projektov
- Nízka kvalita produkovaného softvéru
- Prudký nárast dopytu po softvéri (potreba vyššej produktivity)

Z naznačených problémov nám jasne vyplýva potreba zložitejšieho prístupu pri manažmente softvérových projektov a ich monitorovaní. Existencia mechanizmu, ktorý by umožňoval identifikovať, merať a kontrolovať základne parametre vývojového procesu, je nevyhnutným predpokladom pre zefektívnenie manažérskeho procesu. Keďže do obdobia softvérovej krízy neexistovali dostatočné (dobre definované, spoľahlivé, merateľné) metodiky na vyhodnotenie softvérového projektu, vznikla potreba softvérových metrík.

Podľa môjho názoru by však nebolo správne zjednodušovať príčinu krízy len na absenciu metrík. Metriky sú len jednou z vecí, ktorá môže dopomôcť ku kvalitnejšiemu výstupu. Ako hlavný problém tohto obdobia by som skôr videl nedostatok skúseností s tvorbou komplikovanejšieho softvéru, čo sa týka všetkých účastníkov vývojového procesu (od programátorov až po manažment).

Definícia metrík a ich vlastností

Softvérovú metriku teda môžeme definovať ako prostrediok na meranie základných charakteristík softvérového produktu a s ním súvisiacich vývojárskych procesov. Pre ich použitie a získavanie dát pre rozhodovanie sú však dôležité vlastnosti jednotlivých metrík, na základe ktorých vieme hodnotiť ich použiteľnosť.

Základnou črtou, ktorá sa očakáva od softvérovej metriky je, aby bola schopná nielen opísať nejaký softvérový proces, ale dokázať predvídať jeho ďalší priebeh. Toto je aj kľúčová vlastnosť pre monitorovanie projektu. Okrem toho sa v literatúre[2] ako kľúčové uvádzajú nasledovné vlastnosti:

- jednoduchosť
- objektívnosť
- jednoduchosť získavania
- validnosť
- robustnosť

Na softvér, ako zložitý technický systém, sú kladené viaceré požiadavky. Prvou z nich je funkčnosť, teda schopnosť prevádzať úlohu, na ktorú je určený. Kritériom hodnotenia je teda korektnosť a spoľahlivosť jeho prevádzky. Ďalšou požiadavkou je výkonnosť, ktorú si pri softvéri môžeme predstaviť ako čas odozvy, rýchlosť spracovania a podobne. Do poslednej hlavnej kategórie požiadaviek patria náklady na systém, teda efektívnosť prevádzky systému.

Vyššie spomínané charakteristiky a požiadavky sú práve predmetom skúmania softvérových metrík.

Kategorizácia metrík

Pojem softvérovej metriky je relatívne všeobecný a oblasť aplikovania predstavuje široké spektrum merateľných vlastností a procesov týkajúcich sa softvéru.

Základné rozdelenie metrík predstavuje najvšeobecnejšiu klasifikáciu na produktové a procesné metriky. Produktové metriky vyjadrujú vlastnosti softvérového produktu počas celého času jeho vývoja, čo sa týka hlavne jeho veľkosti, komplexnosti alebo vyprodukovanej dokumentácie. Procesné metriky opisujú vlastnosti vývojového procesu. Pod týmito vlastnosťami si predstavujeme napríklad čas potrebný na vývoj, použité prístupy, metódy, technológie, prípadne schopnosti vývojárskeho tímu.

Ďalšou dôležitým rozdelením je deľba na vlastnosti objektívne a subjektívne. Pojem objektívnosti môže byť v tomto prípade mierne zavádzajúci. Týmto pojmom sú opísané metriky, ktoré by mali vždy produkovať rovnaké výsledky vzhľadom na použitú metodiku počas merania rôznymi pozorovateľmi. Typickým predstaviteľom tohto typu metrík je LOC (lines of code – riadky kódu). Z môjho pohľadu sa intuitívny pojem objektívnosti môže líšiť od toho vyššie definovaného, keďže táto technika merania (LOC) môže často poskytovať zavádzajúce výsledky, pretože množstvo vyprodukovaných riadkov nemusí vždy vystihovať postup projektu alebo programátorskú efektivitu. Okrem toho často diskvalifikuje vyššie programovacie jazyky (keďže zdrojové kódy nie sú také rozsiahle).

Na druhej strane subjektívne metriky predstavujú rôzne kategorizácie softvérov do skupín. Subjektivita tohto merania je definovaná tým, že rôzni pozorovatelia môžu rozdeliť procesy do kategórii iným spôsobom, pretože tieto rozdelenia sú často vnímané rôzne.

Produktové metriky

Keďže hlavným zameraním tejto práce sú charakteristiky softvéru týkajúce sa hlavne kvalitatívnych vlastností softvéru a spoľahlivosti (teda produktové metriky), je potrebné zadefinovať ich základné charakteristiky a rozdelenie.

Základným východiskom produktových metrík pre získavanie kvalitných dát sú zdrojové kódy. Tento postup bol zaužívaný hlavne v začiatočných pokusoch o meranie vlastností softvérových produktov, ale počas vývoja v tejto oblasti sa dospelo k záveru, že je potrebné vychádzať aj zo samotného návrhu a dizajnu či architektúry. Dôvodom týchto záverov bola potreba metrických informácii práve v prvých fázach vývojového cyklu, kedy nie je možné vychádzať zo zdrojových kódov.

Prvou z charakteristík opisovaných produktovými metrikami je veľkosť softvéru alebo programu. Táto vlastnosť je bežne meraná prostredníctvom metódy LOC. Ako už bolo spomenuté, táto metóda môže prinášať mätúce informácie. Napriek tomu je najpoužívanejšia spomedzi veľkostných metrík, hlavne vďaka svojej jednoduchej a presnej definícii. Takisto je jednoduché získavať takéto údaje zo zdrojových kódov a nadviazať ďalšími metrikami na získané dáta. Množstvo metrík ďalej pracuje s dátami týkajúcimi sa počtu riadkov kódu. Na základe tohto údaja vieme vypočítať napríklad náklady na jeden riadok, potrebný čas a podobne.

Ďalšou z hlavných kategórii softvérových metrík je komplexnosť. Táto metrika predstavuje mieru zložitosti použitých procedúr. Pod takouto mierou sa skrývajú podproblémy rýchlosti, množstva nutných zdrojov, prípadne čitateľnosť kódu. Keďže každý algoritmický problém je možné riešiť rôznymi spôsobmi, je potrebné analyzovať komplexnosť každej možnosti. Na riešenie tejto problematiky sa používajú hlavne metódy

pracujúce s grafovou reprezentáciou toku programu. Na základe množstva uzlov a hrán sa vypočítava komplexnosť programu (Cyclomatic Complexity) [2].

Ak vynecháme menej používané metrické modely, poslednou kategóriou sú práve metriky zodpovedajúce kvalite tvoreného produktu.

Kvalitatívne metriky v sebe zahŕňajú vlastnosti ako korektnosť, spoľahlivosť, prenosnosť alebo udržovateľnosť. Tieto vlastnosti sú často aj v rozpore navzájom, preto neexistuje samostatná metrika, ktorá by vyjadrovala celkové kvalitatívne vlastnosti softvéru.

Udržovateľnosť ako kvalitatívna softvérová metrika je definovaná ako náročnosť zmien, podpory a údržby softvérového systému. Intuitívne môžeme predpokladať, že množstvo tohto úsilia bude úmerné komplexnosti a veľkosti systému. Tento predpoklad však môže byť ovplyvnený použitým technológiami a programovacími jazykmi alebo architektúry systému (server - klient, distribuovaný a pod.). Problémom, ktorý sa uvádza v literatúre [1] je práve klesajúca udržovateľnosť počas prevádzky systému, keďže dochádza k jeho postupným zmenám. Z toho teda vyplýva, že aj dobre udržovateľný systém časom podlieha zmenám a teda dochádza k zhoršeniu tejto vlastnosti. Na druhej strane, ako východisko sa uvádza práve správne štrukturovanie systému, ktoré môže tomuto procesu zamedziť.

Softvérová spoľahlivosť

Organizácia IEEE definuje spoľahlivosť ako schopnosť systému alebo komponentu vykonávať požadované funkcie za stanovených podmienok v určitom časovom rozmedzí [4].

Na základe tejto definície sa pojem spoľahlivosti zužuje na tri základné problémy[4]:

- Prevencia chýb
- Detekcia porúch a ich odstránenie
- Merania, ktoré umožňujú maximalizáciu spoľahlivosti (teda prvých dvoch problémov)

Čo si vlastne predstavujeme pod pojmami ako chyby a poruchy ? Vo súvislosti so softvérovou spoľahlivosťou tieto pojmy predstavujú programátorské chyby, ktoré vedú k veľkým nedostatkom vo funkčnosti programu a výrazné odchýlenie od špecifikovaných požiadaviek. Naopak častým zdrojom takýchto chýb môžu byť aj nepresné stanovené požiadavky. Preto je dôležité, aby bola spoľahlivosť systému monitorovaná už počas prvých fáz vývoja, keďže detekcia aj potenciálnych chýb je rovnako dôležitá v tejto fáze ako aj pri samotnej implementácii. Vďaka včasnej identifikácii počas vývoja sú náklady na jej odstránenie oveľa nižšie ako v neskorších štádiách nasadzovania.

Softvér a jeho poruchovosť počas svojho životného cyklu prechádza viacerým štádiami. V prvých štádiách integrácie a testovania vykazuje najvyššiu poruchovosť, ktorá postupne klesá počas použitia. Ako softvér zastaráva, klesá aj jeho používanosť a tým aj množstvo vzniknutých chýb.

Čo teda môžeme urobiť pre to, aby vytváraný systém vykazoval čo najnižšiu chybovosť od prvých fáz vývoja ? Najdôležitejším prvkom tejto prevencie sú prepracované požiadavky, teda aby všetky požiadavky jasne a presne špecifikovali požadovanú

funkčnosť. Ďalším dôležitým prvkom je prepracovaný plán testovania v náväznosti na požadovanú funkčnosť. Tretím predpokladom nízkej chybovosti je dobrá štruktúra systému, ktorá nám dokáže zaručiť, že potrebné zmeny počas nasadzovania a údržby budú prevedené jednoducho a nespôsobia vznik ďalších chýb.

Ako teda meriame spoľahlivosť

Hlavnou úlohou softvérových metrík zameraných na spoľahlivosti je identifikácia oblastí v špecifikácii požiadaviek a samotnom kóde, ktoré by mohli byť potenciálnym zdrojom chýb. Nevyhnutnou súčasťou tohto procesu je dôkladné plánovanie testovania za účelom prevencie chýb v celom rozsahu projektu. Je potrebné zamerať úsilie na prevenciu chýb hlavne na fázy, ktoré môžu mať vplyv na celkovú spoľahlivosť softvéru. Sú to fázy od špecifikácie požiadaviek až po testovanie.

Najbežnejším metrickým prístupom používaným pri stanovovaní spoľahlivosti je meranie priemernej doby medzi jednotlivými vzniknutými chybami počas prevádzky systému. Na základe takéhoto modelu vieme úspešne predpokladať častosť porúch, teda sčasti aj spoľahlivosť. Jedná sa však o pomerne jednoduchú a intuitívnu metódu stanovovania spoľahlivosti a je potrebné pozerať sa na spoľahlivosť v širšom kontexte vývojového procesu.

Spoľahlivosť a špecifikácia požiadaviek

Zmyslom špecifikácie požiadaviek je presné stanovenie funkcionality, ktorú má softvér zabezpečovať. Pod spojením „presné“ sa skrýva kritická potreba toho, aby nedošlo k nedorozumeniu medzi zadávateľom a vývojárom. Je zjavné, že kompletné, dobre štruktúrované a ľahko aplikovateľné požiadavky sú dobrým predpokladom vysokej kvality finálneho produktu a teda aj jeho dobrej spoľahlivosti.

Ako sa spomína v literatúre [4], je dobrým zvykom, ak je vopred definovaná štruktúra samotného dokumentu, vďaka čomu sa zabráni hoci aj neúmyselnému vynechaniu dôležitých detailov.

Vlastnosti dokumentácie požiadaviek, ktoré sú asi viac ako zrejmé, je konzistentnosť požiadaviek, úplnosť alebo dostatočný dôraz na detail.

V špecifikácii požiadaviek by sa minimálne nemali objavovať neurčité frázy (úplnosť požiadaviek), nakoľko nejasnosti môžu mať zlý dopad na navrhnutý dizajn a tým aj na finálnu kvalitu a spoľahlivosť. Takisto, základom dobrého návrhu je použitie štandardizovaných nástrojov, metód a notácií, vďaka ktorým sa vieme vyhnúť zlému pochopeniu spísaných faktov.

Dôležitosť špecifikácie požiadaviek motivovala softvérový priemysel k produkcii metrík, ktoré by napomáhali k vytváraniu, manažovaniu, ale hlavne k vyhodnocovaniu kvality vytvorených dokumentov. Ukážkou takejto metriky je softvér na parsovanie špecifikačných dokumentov [4] , ktorý v dokumentoch vyhľadáva určité frázy. Na základe výskytu a množstva týchto fráz vyhodnocuje kvalitu spracovania požiadaviek.

V spomínanom prístupe boli použité nasledovné kritériá:

- Riadky textu ako ukazovateľ veľkosti dokumentu

- Prikazovacie tvary, teda slová a frázy, ktoré prikazujú zabezpečenie nejakej činnosti (základné požiadavky)
- Pasáže bezprostredne nasledujúce po príkazoch, ktoré predstavujú nižšiu granularitu špecifikácie
- Odkazy na tabuľky, grafy a pod.
- Neurčité frázy – vnímané ako nepresné stanovené požiadavky
- Nekompletné požiadavky – frázy predstavujúce nekompletnosť sekcie (v angličtine „TBD – to be determined“)
- Varianty – požiadavky stanovené spôsobom, ktorý nemá jednoznačnú interpretáciu

Na základe vyššie stanovených kritérií sa vyhodnocuje kvalita dokumentácie v jednotlivých iteráciách jeho tvorby. Takáto automatizovaná interpretácia kvality môže byť značne nápomocná pri hľadaní slabých miest v dokumentácii. Nemôže však úplne nahradiť manuálne metódy vyhodnocovania.

Je potrebné zdôrazniť, že implementácia podobného problému (analýza textu) je nepomerne náročnejšia v jazyku ako je slovenčina. Napriek tomu, že konvenciou v oblasti písania zdrojového kódu je používanie anglických termínov, tento prístup sa všeobecne neuplatňuje pri tvorbe dokumentácie. Preto je všeobecná možnosť využitia tohto prístupu dosť problematická.

Spoľahlivosť v návrhu a implementácii

Pri navrhovaní a implementovaní softvérového systému sa vyhodnocovanie spoľahlivosti zakladá hlavne na softvérových metrikách opisujúcich veľkosť a komplexnosť.

Je všeobecne akceptovaným faktom, že komplexnosť má priamy vplyv na pravdepodobnosť vzniku chýb. Čím komplexnejší je nejaký úsek kódu, tým je aj ťažšie pochopiteľný, stúpa náročnosť realizácie zmien a teda aj množstvo chýb. Okrem spoľahlivosti je teda výrazne ovplyvnená aj udržovateľnosť. Na vyhodnocovanie komplexnosti jednotlivých modulov sa používajú metódy spomínané v prvej časti eseje.

Ďalším prístupom, ktorý sa používa vo fáze implementácie je veľkosť. Jedná sa o najjednoduchšiu formu softvérových metrík, kde sa využíva ako jednotka merania práve riadok kódu (LOC).

Podľa literatúry [4] je najefektívnejším spôsobom ako vyhodnocovať spoľahlivosť implementačných modulov práve kombinácia metrík veľkosti a komplexnosti. Ako príklady vysokej pravdepodobnosti vzniku chýb sa uvádzajú rozsiahle moduly s vysokou komplexnosťou, ale aj moduly obsahujúce hutný kód (veľmi malé rozsahom, veľké komplexnosťou).

Pri objektovom návrhu je možné využiť ďalšie jednoduché metriky kódu, akou je napríklad počet metód na triedu, súdržnosť medzi objektmi a podobne. Znova sa predpokladá, čím zložitejší je kód podľa stanovených kritérií, tým je vyššia pravdepodobnosť vzniku chýb.

Testovanie spoľahlivosti

Testovanie pokrýva dve oblasti odhadovania spoľahlivosti produktu. Prvou z nich je už vyššie spomínaná častosť výskytu chýb a tempo ich nájdenia a ošetrenia, ktorá je asi aj vlastnosťou, ktorá si najčastejšie asociujeme so spoľahlivosťou softvéru. Druhá činnosť predstavuje testovanie špecifikovanej funkčnosti, kde je hlavnou snahou eliminácia chýb v súvislosti nedostatkom funkčnosti.

V súvislosti s výskytom chýb sa využívajú matematické modely. Tieto modely pracujú na základe údajov získaných pri testovaní. Pod týmito údajmi si môžeme predstaviť priemerný počet porúch v nejakom časovom intervale alebo ich interval. Na základe takto získaných dát je možné v jednotlivých fázach testovania predpovedať aj také veličiny ako je napríklad počet „zvyšných chýb“ alebo čas potrebný na ich odstránenie. Tento druh údajov je veľmi vhodný pre sledovanie postupu celého projektu.

Záver

Táto práca nastoľuje otázku, či dokážeme zmerať softvérovú spoľahlivosť ? Problematika sa dá viac-menej zovšeobecniť na otázku „dokážu naozaj metriky vyjadriť všetky vlastnosti softvéru ?“.

Odpoveď nie je podľa môjho názoru úplne jednoznačná. Na jednej strane musím potvrdiť, že existuje množstvo rôzne komplexných metrík na meranie v zásade každého aspektu pri vývoji softvéru. Na druhej strane treba povedať, že ani jedna metrika nedokáže stopercentne vystihnúť svoj cieľ. Projekt s vyšším počtom riadkov nemusí byť časovo náročnejší, takisto projekt, ktorý vykazuje nízku mieru chybovosti, sa môže zákazníkovi zdať ako úplne nekvalitný a nespoľahlivý.

Myslím si, že softvérové metriky treba vnímať ako súčasť softvérového inžinierstva, ktorá nám môže pomôcť objaviť a sústrediť sa na konkrétne problémy, ktoré môžu nastať počas vývoja.

Na druhej strane určite nepredstavujú autonómny systém na základe ktorého by sme jednoznačne vedeli hodnotiť softvérové produkty bez širšieho kontextu a ako hovorí literatúra [3]: „Softvérové meranie je náchylné na chyby a často aj neúspešné“. Na základe týchto faktov si myslím, že metriky by nemali hrať kľúčovú, ale len doplnkovú úlohu pri monitorovaní softvérových projektov.

Keďže nemám vlastné skúsenosti, na ktorých by som mohol ilustrovať svoje závery, len čas a ďalšie skúsenosti ukážu, či sa moje, tu vyslovené, závery potvrdia.

Použitá literatúra

1. LAND, Rikard. Measurements of Software Maintainability. Västerås, 2002. 5 s. Odborná práca. Mälardalen University.
2. MILLS, Everald. Software Metrics. Seattle, 1998. 43 s. Referát. Carnegie Mellon University.

3. ROCHE, John Software metrics and measurement principles. In ACM SIGSOFT Software Engineering Notes [online]. New York : ACM, 1994 [cit. 2010-10-22]. Dostupné z WWW: <http://doi.acm.org/10.1145/181610.181625>. ISSN 0163-5948.
4. ROSENBERG, Linda; HAMMER, Ted; SHAW, Jack. NASA [online]. 1998 [cit. 2010-10-22]. Software Metrics and Reliability. Dostupné z WWW: <http://satc.gsfc.nasa.gov/support/ISSRE_NOV98/software_metrics_and_reliability.html>.

Annotation

Can we measure software reliability ?

When monitoring the development of software systems we are facing a number of problems. Besides the obvious problems such as complexity and uncertainty of requirements, major problem is software invisibility. For the smooth course of the project it is very important to estimate the amount of effort that is needed for development in various stages. It is in an issue on which the software metrics are aiming. Software metrics are quantitative methods measuring certain characteristics of software and can be particularly helpful in monitoring the project development. However, there are many metrics that can be used but they might often produce contradictory results. It is necessary to consider which of these metrics can offer reliable results so that we can plan further course of the project. This essay focuses on the reliability metrics as one of the important properties of software.