Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach

Žilina 2.–5. júla 2009

BUGZILLA PRI RIADENÍ VÝVOJA SOFTVÉRU

FODREK, Peter, (SK)

1 Úvod

Zúčastnil som sa na akcii „Dni príležitostí 2009“, kde sa stretali firmy so študentami a pedagógmi a ponúkali pracovné miesta. V diskusiách s firmami mi bolo povedané alebo potvrdené, že jedným z najväčších nedostatkov absolventov je práca v tímoch. Časť problémov riešia Source Code Management (SCM) systémy, ktorých vedľajšie efekty sme sa rozhodli použiť aj v hodnotení. SCM, a teda zdieľaním kódov, sa zaoberám v inom článku. Druhá časť problémov je organizácia práce v tíme. Na to slúžia programy hlásenia chýb a plánovania úloh. Jedným z programov na riešenie tímovej synchronizácie je OSS systém Bugzilla. Samotný názov napovedá pôvod systému. Bol totiž vytvorený „na lovenie vší“ (bugov) v programoch vytvorených Mozilla Foundation (odtiaľ koniec názvu).

2 Inštalácia

Bugzilla je súbor Perl skriptov. Z toho vyplýva, že rozhranie je primárne webové, hoci sa môže použiť aj lokálne. Na svoju činnosť potrebuje Bugzilla databázu Oracle, MySQL alebo PostgreSQL a k tomu príslušné Perl moduly, ktoré sa spúšťajú ako CGI skripty. Inštalačný skript s parametrom `--check-modules`, zistí potrebné a voliteľné moduly Perl-u potrebné na beh Bugzilly. Napriek tomu, že východzie nastavenie je pre MySQL, zvolil som PostgreSQL, lebo mi pripadala menšia a ľahšie administrovateľná ako MySQL a navyše som ju musel na BSD server s menom fuzzy inštalovať lokálne, lebo tam nebola inštalovaná

a s inštaláciou MySQL som nemal skúsenosti. Po inštalovaní modulov Perl je, pre inú databázu ako MySQL, nutné prepísať položku Driver v konfiguračnom súbore na driver pre nainštalovanú databázu, a to taký, ktorý sme doinštalovali ako modul Perlu. Pre PostgreSQL je to ovládač „Pg“ s dodržaním veľkého písmena P. Problém môže nastať ak nainštalujeme len jeden ovládač k databáze a chceme použiť inú databázu. Inštalačný skript s vyššie uvedeným parametrom prestane zobrazovať zoznam možných ovládačov k databázam, ak je v systéme nainštalovaný aspoň jeden z troch ovládačov pre databázy v Perle. Potom treba nastaviť databázu, a keďže na serveri Fuzzy nemám práva administrátora, musel som urobiť lokálnu inštaláciu PostgreSQL.

Z rovnakého dôvodu som použil lokálnu inštaláciu Perlu, keďže ten pôvodný mi nedovoľoval doinštalovať moduly a ovládače databáz bez práv užívateľa root (Unix administrátor). Lokálna inštalácia navyše zvyšuje bezpečnosť systému, ktorá je na fuzzy až paranoidná, čo ale nie je na škodu. Kvôli tomu treba použiť shell skript s využitím editora sed na zmenu cesty k interpereteru jazyka Perl. Po nastavení PostgreSQL treba vytvoriť databázu s menom bugs, ktoré je prednastavené v konfiguračnom súbore Bugzilly. Ak použijeme iné meno, treba prepísať konfiguračný súbor. Ten musíme aj tak prepísať, lebo musíme zadať prihlasovacie meno a heslo k databáze. Potom musíme ešte nastaviť port PostgreSQL, ak nie je použitý prednastavený, a to ako v nastavení PostgreSQL tak v nastavení Bugzilly. Tesne pred záverom musíme spustiť PostgreSQL démona(v MS Windows je obdoba Service, v DOS-e je to TSR program).

Na záver musíme spustiť HTTP démona, najlepšie Apache. Tu vznikne na dobre zabezpečených systémoch ďalší problém: tieto systémy neumožňujú použiť CGI skripty v domovských adresároch užívateľov. Preto je možné buď inštalovať lokálny Apache, alebo umiestniť skripty do určeného adresára, čo ale zníži bezpečnosť. Preto sme sa rozhodli pre lokálny Apache. Ten sme nakonfigurovali tak, že je možné pristupovať k adresáru, kde sme rozbalili inštaláciu Bugzilla. Port bol nastavený na 5000 a spustili sme Apache. Potom už len stačilo, v adresári, kde bola rozbalená Bugzilla, skontrolovať nastavenie hesiel a databázy. Potom je nutné spustiť inštalačný skript Bugzilla bez parametrov. Tým sa upraví nastavenie a práva k skriptom. Od toho okamihu možno Bugzillu použiť z lokálneho počítača a na nedokonale zabezpečených systémoch aj z iných počítačov. Na test sme použili prehliadač lynx z BSD.

3 Proces riadenia projektov ako paralelné programovanie

Na server s menom Fuzzy sa nie je možné pripojiť cez porty obsluhované užívateľskými démonami. Takým démonom je aj náš Apache. Preto sme museli vytvoriť SSH tunel. Pre programátora s akýmikoľvek skúsenosťami s Unixom, nie je nič ľahšie, ako napísať tunelový klient a tunelový server, skompilovať a spustiť ich. To ale platí len za predpokladu, že chce pracovať na jednoprocesorovom počítači s jednojadrovým procesorom. Čas takých programov je ale spočítaný. Tvrdí to aj Intel, keďže v rámci konferencie Supercomputing 08, konanej v novembri 2008, poriadal diskusiu s veľavravným názvom: „There Is No More

Sequential Programming. Why Are We Still Teaching It?“ [1, 2]. V diskusii s Intelom to potvrdili aj zástupcovia spoločností AMD, Sun, IBM a nVidia. Pre efektívne paralelné programovanie však platí Amdahlov zákon [3]:

$$k = \frac{1}{1 - f + \frac{f}{n} + H(n)}, \tag{1}$$

kde k je pomer rýchlosti paralelnej aplikácie k rýchlosti sekvenčnej aplikácie,
f je pomer časti kódu, ktoré môžu bežať paralelne, a teda nie je dané poradie behu úloh k celkovej veľkosti kódu resp. pomer časov behu týchto kódov,
n je menšie z čísel N_1 a N_2 kde N_1 je počet jadier počítača, na ktorom beží kód a N_2 je počet procesov alebo vlákien aplikácie,
$H(n)$ je réžia správy procesov a vlákien pre N_2 procesov na N_1 jadrách. Pre ideálny operačný systém platí $H(n) = 0$, ak $N_1 > N_2$.

Pre ideálny aj reálny operačný systém platí (rovnica pre $N_1 = N_2$, v reálnom systéme platí aj pre $N_1 > N_2$)

$H(n) \to 0$, ak $N_1 = N_2$,

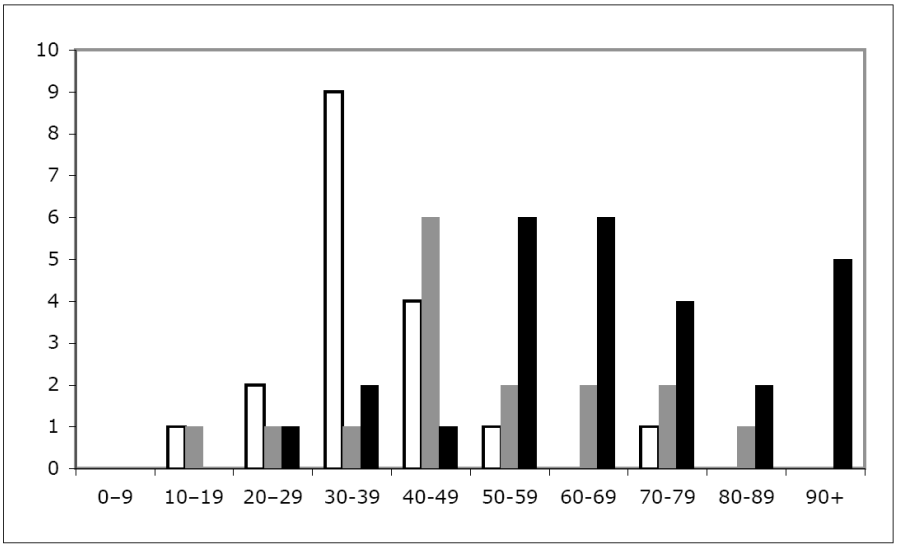
$0 > H(n) \leqq f$, ak $N_1 < N_2$.

Kedže klient bežiaci na serveri a server bežiaci u klienta majú kódy podľa prídavných súborov BugzillaKlient.c a BugzillaServer.c, ktoré majú f=1 pre $N_2 = 1$ v prípade klienta a $f = 1$ pre $N_2 = 2$ v prípade servera. V tom prípade je pre server $k = 2$ na dvojjadrovom procesore, ale $1/2 \leqq k < 1$ na jednojadrovom procesore. Preto výrazne neodporúčam použiť tento server na počítačoch s menej ako 2 jadrami, keďže aj webový prehliadač je proces a aj Apache je najmenej jeden proces. Ide o školský jednoduchý príklad. *Bežný zložitý softvér nikdy nemôže dosiahnuť $f = 1$, lebo vždy existujú časti kódu, ktoré nemôžu bežať paralelne*.

Obdobne ako s paralelným kódom je to aj s členmi programátorského tímu. Platí paralela, že členovia tímu sú procesy a súbory sú jadrá CPU. Dehnadi [4] rozdelil študentov programovania do troch skupín:

- 44% študentov použilo na riešenie všetkých, alebo skoro všetkých úloh rovnaký spôsob uvažovania *bez ohľadu na to či správny alebo nesprávny*. Nazvime ich *ľudia s konzistentným uvažovaním.*
- 39% študentov použilo na riešenie rozdielnych úloh rozdielny spôsob uvažovania. Nazvime ich *ľudia s nekonzistentným uvažovaním.*
- Zostávajucich 8% študentov odmietlo odpovedať na všetky alebo skoro všetky otázky resp. odpovedalo neviem. Nazvime ich *ľudia bez vyhraneného uvažovania.*

Potom dal rôznym skupinám odhadnúť, ktorá zo skupín bude najúspešnejšia. Programátori a iní zamestnanci v IT, takmer všetci, *predpovedali úspech skupiny bez vyhraneného*



Obr. 1: Úspešnosť typov študentov – biela = nekonzistentné myslenie na začiatku aj konci semestra, sivá = nekonzistentné alebo nevyhranené myslenie na začiatku a na konzistentné konci semestra, čierna = konzistentné myslenie na začiatku aj konci semestra, na osi *X* sú získané percentá bodov, na osi *Y* je množstvo študentov

uvažovania. Neprogramujúci humanitní vedci, matematici a historici, takmer všetci, *predpovedali úspech skupiny s nekonzistentným uvažovaním*.

Reálne výsledky sú na Obr. 1. S nimi korešpondujú aj iné výsledky. 57% profesionálov v IT sú strašní individualisti [5]. Aby toho nebolo dosť, tak len *spôsobov programovania je asi* $2,6 \cdot 10^{42}$.

Riadenie tak rozdielneho tímu, sa teda podobá plánovaniu úloh na „hybridnom multijadre“. Jediný použiteľný koncept plánovania úloh na procesore s nerovnakými jadrami t. j. „hybridnom multijadre“ je koncept koprocesorov t. j. hlavné jadro prideľuje úlohy ostatným jadrám podľa toho, ktoré sú voľné a na čo sú špecializované. Ak neexistuje voľné jadro, tak úlohu počíta hlavné jadro *(koncept Commodore C64)*. Modifikácia môže byť v tom, že koprocesor „číha“ na jemu vhodnú úlohu a ak ju nájde oznámi hlavnému jadru, že ju bude počítať. *Oba tieto koncepty podporuje, ako metodiky riadenia práve Bugzilla.*

Reálne je to tak, že tím je rozdelený na podtímy, ktorým hlavný vývojár rozdáva úlohy. Každý tím má svojho podvedúceho. Títo podvedúci vedia stanoviť aj časový odhad riešenia problému alebo pridania novej vlastnosti do svojho podprojektu. V princípe nič nebráni, aby každý programátor bol spoluvedúcim podprojektu a teda podvedúcim. Na čo sa ale zabúda, sú testeri. Tí by mali tiež byť členmi tímu a nemal by medzi nimi byť programátor, s výnimkou, ak ide o bezpečnostné aspekty t. j. podprojekt je bezpečnostnou časťou projektu. Ale aj vtedy by mal byť aspoň jeden z testerov laik až človek, čo počítač vidí prvý krát (tzv. BFU). Toto má vyriešené jedine Microsoft. Napr. pri Windows 7 sú tímy podieľajúce sa na vývoji zložené z n vývojárov, n testerov a $n/2$ programových manažérov.

Veľké OpenSource projekty nemôžu zohnať testerov z radov BFU, a teda nedbajú na použiteľnosť pre ľudí s nekonzistentným myslením. Toto sa ale dá riešiť tým, že BFU používajúci OSS, budú chyby hlásiť práve cez systém typu Bugzilla. Problém je, že OSS používajú BFU vo veľmi malej miere a to práve kvôli netestovaniu „použiteľnosti pre BFU“. To znamená jediné: OSS potrebuje podporu testerov vo veľkých firmách ťažiacich z existencie OSS, alebo ešte lepšie zvýšiť počet BFU používajúcich OSS administratívnym nariadením. O to prvé sa snažia hlavne Sun(Oracle) a IBM. O to druhé sa snaží Riaditeľstvo pre informatiku Európskej komisie a Európsky parlament. To, čo nemá Microsoft vyriešené je technická stránka veci, aby ulahodil testerom používa veľké tímy o veľkosti 100 ľudí. V českom preklade knihy *Programmers at work* sa píše doslova:

„Došel jste k nějakému konkrétnímu stylu práce, který je pro vaše účely nejproduktivnější? C. Wayne Ratliff: Pracuji buď sám, nebo s velice malou skupinou lidí. *Jakmile má tým víc než šest členů, nastane chaos*. Ted Glasser, který má řadu patentů, záznam v *Kdo je kdo* a patří mezi důležitější postavy počítačového oboru, jednou prohlásil, že *Největší tým, který ještě zvládne řídit, je tým, který může vzít na pizzu ve volkswagenu*. Od té doby změnil písničku – *teď už je to obyčejné americké auto*. Vřele s ním souhlasím“. Práve nevyhnutnosť mať mikrotímy, aby sa dal písať technicky kvalitný kód, má vyriešené vynikajúco Open source.

Preto v Microsofte nevedú tímy programátorov programátori a to dáva OSS lepšiu kompatibilitu s inými riešeniami a/alebo kvalitnejšie technické prevedenie.

4 Aplikácia v systéme Bugzilla

Ako sme uviedli, Bugzilla má aj webové rozhranie a po spustení klienta, servera z SSH tunelu, ktoré sme skompilovali, sa k bugzille dostaneme cez URL `http://localhost:5000/` ak port servera je 5000 a server má spustený Web server, SQL a verejný *kľúč* k súkromnému kľúču v súbore web_rsa je na serveri v súbore authorized_keys pre tcpcli01, pozri Listing1.

Listing 1

```
-bash-2.05b$ more .ssh/authorized_keys

command="/home/fodrek/tunel/tcpcli01"ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAIEAmpqvwj0UDJScv2mRjELmOMvMoeZxCn1Y35Bs80se5o6
0W2I8Hxv3ZEItaIQv/yAmshwMOnpm7udNYjRy9Ba91rIauD4OkJEdPrIaTlYf9FPUne+YBa
uXMZ6EOdvTr8gIMBfayuek3Qp1uEGsbvC54KSBA LtsdjgXuhAGb8ScCe8=peto@fodrek

command="/home/fodrek/tunel/tcpcli01"ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAQEAt4EcD6afQVzC9wOWq/5FlkJjI+M9YAPbdBOeAROAukN
dJvFvCqpQVz8GbRQMgZ+exhQUx.......usTzn9YIf8a2GZykleVvHyaFfQPDEE6Y1cuK+D
OMdIj0rb1IuapyqG7xva0lz1Glo+2z83Q==peto@fodrek
```

Ako vidieť jeden užívateľ môže mať viac a rôzne dlhých a tým rôzne bezpečných kľúčov. Riadok kódu v prílohe server

```
execl("/usr/bin/ssh","ssh","-T","-i","/home/peto/.ssh/web_rsa",
"fodrek@pid.kasr.elf.stuba.sk",ppid,NULL);
```

má hrubo vytlačenú cestu k súkromnému kľúču bez hesla a modrým užívateľa pod ktorým sa na webserver prihlásime. Obrázok 2 a Obr. 3 je web rozhranie systému Bugzilla ako aj Eclipse rozhranie pre Bugzilla v doplnku na prácu v tíme Mylyn Bugzilla Connector. Login name je e-mail, kam sa posielajú emaily pre Vás ak to povolíte.

Do elektronickej verzie článku (je dostupný na webovej stránke konferencie `http://frcatel.fri.uniza.sk/OSS09/Materialy`) sú vložené obrázky, na ktorých je vidieť používanie systému Bugzilla v prostredí Eclipse. Pre nečitateľnosť a z priestorových dôvodov tieto obrázky v tlačenej verzii článku neuvádzame.

1. Nastavenie Bugzilla Servera.
2. Vytvorenie nového riadiaceho reportu.
3. Vyplnenie reportu.
4. Odoslanie reportu.
5. Zmena stavu reportu – t. j. napr. z vytvorený na pridelený vývojárovi X, alebo na vyriešený.
6. Využitie vyhľadávania v rámci reportov napr. tie, ktoré mám riešiť.
7. Zobrazenie nájdeného reportu v druhej záložke.

Ďalšie obrázky sú snímky obrazoviek pre webové rozhranie Bugzilly:

1. Prihlasovacia obrazovka.
2. Hlavná obrazovka užívateľa.
3. Výber projektu (produktu).
4. Okno s vlastnosťami produktu.
5. Prístup k reportom produktu.

5 Záver

Ukázali sme vhodnosť použitia stratégie použitej v Bugzille na riadenie projektu, s tým, že podtímy riešia „Komponenty Produktov“ a ukázali sme ako sa používa Bugzilla v Eclipse a prehliadači.

Literatúra

[1] STEINBERG, P.: *Sequential programming is no more. Let's teach parallel programming*. [online], Santa Clara : Intel Software Network, 2008, dostupné na URL adrese: `http://software.intel.com/en-us/blogs/2008/11/12/sequential-programming-is-no-more-lets-teach-parallel-programming/`

[2] MANCHESTER, P.: *Intel rallies rivals on parallel programming education*. [online], London : Situation Publishing, Ltd., 2008, dostupné na URL adrese: `http://www.theregister.co.uk/2008/11/13/intel_parallel_programming_priority`

[3] VADKERTI, L.: Dvořák,V.: V*ývoj paralelních aplikací s Intel threading tools*. [online], Brno : diplomová práca – Vysoké učení technické v Brne, 2007, dostupné na URL adrese: `http://www.fit.vutbr.cz/study/DP/rpfile.php?id=539`

[4] DEHNADI, S. – BORNAT, R: *The camel has two humps (working title)*. [online], London : Middlesex University, 2006, dostupné na URL adrese: `http://www.cs.mdx.ac.uk/research/PhDArea/saeed/paper1.pdf`

[5] The Inquirer: *57 percent of IT professionals are sad individuals*. [online], London : Incisive Media Limited, 2007, dostupné na URL adrese: `http://www.theinquirer.net/inquirer/news/145/1041145/57-cent-it-professionals-sad-individuals`

[6] NOSKA, M.: *Windows 7: Na jeho vývoji pracuje 2500 expertov Microsoftu*. [online], Bratislava : Digital Visions, spol. s r.o, 2008, dostupné na URL adrese: `http://www.itnews.sk/buxus_dev/generate_page.php?page_id=55956`

[7] LAMMERS, S. – ZNAMENÁČEK, T.: *C. Wayne Ratliff – 2 (Programmers at Work)*. [online], Praha : Stickfish, s. r. o, 2008, (preklad z Redmod, WA : Microsoft Press, 1986), dostupné na URL adrese: `http://www.abclinuxu.cz/clanky/rozhovory/c.-wayne-ratliff-2-programmers-at-work`

Kontaktná adresa
Peter FODREK (Ing., PhD),
Ústav riadenia a priemyselnej informatiky FEI STU v Bratislave,
Ilkovičova 3, 041 20 Bratislava
peter.fodrek@stuba.sk

Fakulta riadenia a informatiky
Žilinská univerzita

OTVORENÝ SOFTVÉR VO VZDELÁVANÍ, VÝSKUME A V IT RIEŠENIACH

Zborník príspevkov medzinárodnej konferencie OSSConf 2009

2.–5. júla 2009
Žilina, Slovensko