O DVOCH REVÍZNYCH SYSTÉMOCH A ICH BUDÚCNOSTI

Vývoj softvéru posúva ľudstvo každým dňom o malý kúsok vpred. Preto tým, ktorí sa o tento posun starajú, treba vytvoriť čo najlepšie podmienky.

Juraj Višňovský

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

Abstrakt. *Systémy na správu verzií sú základným kameňom väčšiny dnešných softvérových systémov. Manažment verzií totiž v nemalej miere rozhoduje o úspechu, prípadne neúspechu celého projektu. Dôvodom je narastajúca zložitosť vyvíjaných softvérových systémov. Z tohto dôvodu je výber vhodného nástroja na manažment verzií kľúčový. Pre všetky podporné prostriedky na správu verzií je však charakteristická príslušnosť k jednému z dvojice revíznych systémov. Tými sú centralizované a distribuované systémy na správu verzií. Táto esej analyzuje oba menované systémy, opisuje proces výberu systému z tejto dvojice a ponúka alternatívu, ktorá by ich v budúcnosti mohla doplniť, prípadne nahradiť.*

Kľúčové slová: *správa verzií, distribuované systémy, centralizované systémy, budúcnosť správy verzií, podpora vývoja*

Úvod

V dnešnej dobe sú systémy na správu verzií neoddeliteľnou súčasťou vývoja akéhokoľvek softvéru. Táto skutočnosť je spôsobená najmä narastajúcou zložitosťou novovznikajúcich projektov. Dalo by sa povedať, že úspech veľkých i malých softvérových projektov stojí práve na základoch vytvorených systémom na správu verzií.

Systémy na správu verzií pozostávajú z úložiska, obsahujúceho zdrojové súbory vyvíjaného projektu. Tieto systémy sledujú zmeny vykonané v jednotlivých súboroch a umožňujú vývojárom prístup k ľubovoľnej verzií softvérového systému. Okrem samotných zmien systémy na správu verzií umožňujú napríklad aj identifikáciu autorov jednotlivých modifikácií.

Z hľadiska architektúry sa systémy na správu verzií delia na centralizované a distribuované [1]. Centralizované systémy na správu verzií využívajú klient-server model, kde server uchováva všetky údaje o projekte a vývojári sa na tento server dopytujú so žiadosťou o sprístupnenie určitej časti projektu, na ktorej chcú pracovať. Protipólom sú distribuované systémy na správu verzií, pri ktorých každý z vývojárov pracuje lokálne s fungujúcou kópiou celého repozitára.

V prvej časti eseji prinášam alternatívny pohľad na využitie systémov na správu verzií. V ďalšej časti sa zameriavam na faktory, ktoré majú rozhodujúci vplyv pri výbere vhodného systému na správu verzií, následne prinášam pohľad na súčasný stav týchto systémov najmä z hľadiska ich použiteľnosti a z toho vyplývajúcej popularity. V samom závere eseje opíšem moju víziu budúcnosti systémov na správu verzií.

Alternatívne spôsoby využitia

Skutočnosť, že systémy na správu verzií sú určené primárne na manažment zdrojového kódu, je zrejmá. Nakoľko človek je tvor vynaliezavý, našiel pre tieto systémy aj viacero alternatívnych spôsobov využitia.

Systém na správu verzií môže poslúžiť napríklad ako pomôcka pri výučbe. Tento prípad použitia budem konkretizovať na príklade, kedy chce pedagóg ponúknuť študentom preberanú látku zaujímavejším spôsobom. V tomto prípade sa môže jednať o vopred pripravené aplikácie, ktoré ich autor rozdistribuuje využitím systému na správu verzií medzi študentov. Takýto spôsob výučby je založený na zážitkovej pedagogike, kedy študenti neprijímajú novú vedomosť len vo forme teoretických poznatkov, ale majú možnosť si túto vedomosť osvojiť práve vďaka získaniu skúsenosti. Ako konkrétny príklad si dovolím uviesť situáciu, kedy pedagóg vysvetľuje svojim študentom dôvody a spôsoby ošetrovania výnimiek. V takomto prípade môže pedagóg vytvoriť prototyp nejakej reálne využiteľnej aplikácie a prostredníctvom systému na správu verzií túto aplikáciu rozšíri medzi svojich študentov. Študenti nie sú nútení vyvíjať aplikáciu, ktorá je irelevantná z pohľadu preberanej látky, ale môžu sa zamerať na pochopenie preberanej témy a riešenie stanovených úloh. V tomto prípade by ich úlohou bolo ošetrenie všetkých výnimiek v aplikácii pripravenej pedagógom, ktorý ich zámerne neošetril. Takýmto spôsobom by mali študenti lepšie pochopiť problematiku výnimiek, nakoľko by im bol predostretý konkrétny prípad ich použitia a na vlastnej koži by pocítili následky neošetrenia výnimiek.

Systémy na správu verzií je takisto možné využiť pri písaní dokumentov. Takýto prípad použitia je síce už na prvý pohľad zrejmý, avšak vzhľadom na primárne poslanie týchto systémov, ktorým je manažment zdrojového kódu, môžeme aj tento spôsob využitia považovať za alternatívny. Systémy na správu verzií sú založené na identifikácií rozdielov v zdrojových textoch, a teda z toho vyplýva obmedzenie tohto prípadu použitia. Dokumenty totiž nemôžu mať ľubovoľný formát, nakoľko ich formátovací jazyk dokumentu musí mať podobu obyčajného textu, aby ho bolo možné verziovať. Správa

verzií vytváraného dokumentu má opodstatnenie v dvoch prípadoch. Prvým z nich je prípad, kedy autor píše samostatne veľmi rozsiahly dokument. Využitie niektorého zo systémov na správu verzií dáva autorovi v tomto procese možnosť rozdistribuovať vytváraný dokument. Namiesto lokálnej kópie takto vznikne minimálne jedna kópia dokumentu v repozitári. Ďalšie kópie sa môžu nachádzať na rôznych ďalších zariadeniach, na ktorých autor na tomto dokumente pracoval. Táto skutočnosť zvyšuje komfort autora, keďže mu je umožnená práca na dokumente z ľubovoľného zariadenia a znižuje riziko straty údajov v prípade zlyhania jedného zo spomínaných zariadení. Systémy na správu verzií však prirodzene umožňujú aj jednoduchý návrat do niektorého z predchádzajúcich stavov dokumentu. Okrem tohto spomínaného prípadu je pochopiteľne možné takýmto spôsobom podporiť aj kolektívnu prácu na dokumente. Systémy na správu verzií umožňujú jednoduchú distribúciu dokumentov medzi jednotlivými spoluautormi a navyše, rovnako ako pri verziovaní zdrojového kódu, umožňujú identifikáciu autorov jednotlivých zmien. Táto funkcionalita je pri kolektívnom písaní dokumentu azda najzásadnejšia, nakoľko umožňuje vyvodiť zodpovednosť v prípade problémov, respektíve dokazuje autorstvo prispievateľov jednotlivých častí dokumentu.

Ako ďalší prípad použitia uvediem vývoj projektov s otvoreným zdrojovým kódom (angl. *open source*). V tomto prípade je možné vyvíjať projekt už od svojho počiatku ako projekt s otvoreným kódom, prípadne je možné konvertovať už existujúci projekt na projekt s otvoreným kódom. Oba tieto prípady vyžadujú jedinú vec, a tou je verejný repozitár. V tomto prípade je zdrojový kód dostupný širokej verejnosti a je umožnené prakticky akémukoľvek prispievateľovi upravovať existujúcu funkcionalitu, pridať novú funkcionalitu prípadne použiť časť vyvinutej funkcionality na iné účely.

S posledným menovaným prípadom pomerne úzko súvisí podpora kolaborácie prostredníctvom systémov na správu verzií. Okrem spomínanej kolaboračnej činnosti v rámci vývoja projektov s otvoreným kódom, je možné využiť distribuované systémy na správu verzií aj ako nástroj na podporu kolaborácie v užšom kontexte. Týmto kontextom myslím vývoj softvérového systému tímom, ktorý je geograficky distribuovaný v rámci určitej krajiny, kontinentu alebo aj v rámci celého sveta. Systémy na správu verzií podporujú spoločnú prácu takejto skupiny ľudí na dosahovanie nimi stanovených cieľov. Podpora kolaborácie spočíva v zjednodušení zdieľania a prístupu k zdrojovým súborom vyvíjaného projektu. Za príjemný vedľajší produkt tohto prípadu použitia považujem skutočnosť, že takýmto spôsobom sa dá jednoducho zostaviť najlepší, prípadne najlacnejší možný tím a vďaka tomu je možné vyvinúť kvalitnejší softvérový systém, respektíve je možné ušetriť zdroje zostavením finančne nenáročného tímu.

Problematika výberu systému na správu verzií

Avšak na alternatívne spôsoby využitia systémov na správu verzií, pri voľbe jedného z nich, prihliada málokto. Z tohto dôvodu sa pokúsim opísať objektívnejšie príčiny, ktoré majú dopad na výber vhodného systému. Tento výber je totiž jedným z prvých dôležitých rozhodnutí, ktorým sa pri zahájení práce na novom projekte nevyhne žiadny tím a vo väčšine prípadov ani jednotlivec. Pri rozhodovaní treba v prvom rade zvážiť povahu projektu, ktorý sa bude riešiť. Centralizované systémy sú vhodné najmä pre málopočetné

tímy a na projekty, pri riešení ktorých sa pracuje s objemnými binárnymi súbormi a bolo by neekonomické snažiť sa ich distribuovať. Na druhej strane distribuované systémy sa hodia prakticky na všetko ostatné. Umožňujú totiž prácu bez nutnosti pripojenia k sieti Internet, sú vhodné obzvlášť pre tímy s veľkým počtom vývojárov, umožňujú prácu na jednotlivých funkcionalitách softvérového systému aj takým prispievateľom, ktorí nie sú riadnymi členmi vývojárskeho tímu a majú aj množstvo iných výhod.

V rámci predmetu tímový projekt sme sa pochopiteľne už v ranej fáze riešenia projektu museli rozhodnúť pre jeden zo systémov na správu verzií. Rozhodnutie padlo na distribuovaný systém na správu verzií, konkrétne na nástroj Git. Náš tím pochopiteľne nepozostával z veľkého počtu vývojárov a dôvodom našej voľby nebol ani predpoklad pomoci od externých prispievateľov. Pre Git sme sa rozhodli z iného, prozaickejšieho dôvodu. Špecifikom nášho tímu, v rámci predmetu tímový projekt, je absencia denného fyzického kontaktu jednotlivých členov tímu. Stretnutia, počas ktorých by sme spoločne vyvíjali určitú funkcionalitu sú pomerne zriedkavé a väčšina komunikácie medzi členmi tímu je realizovaná elektronickou formou. Ďalšou osobitosťou nášho tímu je distribúcia vývojárov v rámci celého Slovenska. Berúc v úvahu tieto špecifiká a skutočnosť, že s nástrojom Git mám ja osobne ako aj väčšina ďalších členov tímu pomerne bohaté a pozitívne skúsenosti, bola naša voľba systému na správu verzií jednoznačná.

Sťahovavé projekty

Rovnako ako náš tím, sa pre distribuované systémy na správu verzií rozhoduje stále väčšie množstvo vývojárov. V podstate už od začiatku dvadsiateho prvého storočia sme svedkami veľkého rozmachu distribuovaných systémov na správu verzií [1]. Dôvodom tohto rozkvetu je jediná skutočnosť, a tou je fakt, že distribuované systémy sú jednoducho vhodnejšie pre podporu vývoja väčšiny projektov než centralizované systémy. Tento nárast popularity je spôsobovaný nie len vznikom nových projektov postavených na distribuovanom systéme, ale aj postupnou migráciou projektov [2] z centralizovaných systémov na distribuované. Toto sťahovanie si vysvetľujem tak, že vývojári týchto projektov začali využívať centralizované systémy ešte v čase, kedy distribuované systémy neboli natoľko rozšírené. A pravdepodobne ich využitie nemalo v predchádzajúcich dekádach také opodstatnenie ako je tomu v dnešnej dobe. Dôvodov, ktoré spôsobujú túto postupnú migráciu, je hneď niekoľko. Ja opíšem päticu najzásadnejších výhod distribuovaných systémov, ktoré spôsobujú migráciu veľkých projektov z centralizovaných systémov [2].

Prvým dôvodom je možnosť vývoja bez nutnosti pripojenia k sieti Internet. Vývojár môže odovzdávať artefakty vyvíjaného zdrojového kódu do lokálneho repozitára. Následne po získaní pripojenia k sieti Internet je možné všetky vykonané zmeny odoslať do hlavného repozitára. Okrem prípadu, kedy vývojár nemá prístup k sieti Internet, je táto funkcionalita prospešná napríklad aj v prípade zlyhania servera, ktorý uchováva repozitáre vyvíjaného projektu. Moja skúsenosť, ktorú som získal počas vývoja softvérového systému v rámci predmetu tímový projekt, výhodnosť distribuovaných systémov len potvrdzuje. Pri riešení projektu bola pomerne bežným javom nefunkčnosť servera s repozitárom. V prípade, že by sme pracovali s centralizovaným systémom, bol by náš projektu po dobu výpadku tohto servera prakticky kompletne paralyzovaný. Nakoľko

sme však pracovali s nástrojom Git, bol každý jeden vývojár schopný pokračovať v práci lokálne. Následne, po obnovení servera bolo vývojárom umožnené aktualizovať verziu repozitára postupným odovzdávaním artefaktov zdrojového kódu.

Druhým dôvodom migrácie je jednoduchý spôsob vytvárania nových vetiev projektu. Toto zvyšuje odvahu vývojárov pri experimentovaní s vyvíjaným produktom bez toho, aby sa museli obávať dopadu ich zmien na projekt vo vetve, s ktorou momentálne pracujú. Skutočnosť, že distribuované systémy umožňujú vývoj vo viacerých vetvách, môže prísť vhod napríklad v prípade, kedy vývojár pracuje na funkcionalite v rámci pomerne veľkého softvérového systému. Nie je si istý možným dopadom jeho práce na ostatné časti systému, a tak si vytvorí experimentálnu vetvu v ktorej je mu umožnené bez akýchkoľvek obáv vyvíjať a testovať danú funkcionalitu. Následne, vo chvíli, kedy vývoj funkcionality dokončí a overí si funkčnosť celého systému môže prakticky bez obáv experimentálnu vetvu zlúčiť s tou pôvodnou. V rámci tímového projektu sa vývoj v izolovaných vetvách osvedčil mne a aj ostatným členom tímu. Možnosť vetvenia bola jedna z príčin, vďaka ktorej sa nám podarilo udržiavať repozitár v konzistentnom stave.

Tretím dôvodom je zjednodušenie spájania vetiev projektu. V distribuovaných systémoch na správu verzií je tento proces do istej miery automatizovaný, nakoľko systém má väčší prehľad o zmenách vykonaných vývojárom. Distribuované systémy najskôr vyhľadajú spoločného predka oboch spájaných vetiev, identifikujú množinu zmien medzi danými vetvami a v prípade, že nedošlo k žiadnemu konfliktu, vytvoria novú verziu v zlučovanej vetve. Na druhej strane pri centralizovaných systémoch je proces spájania vetiev nepomerne komplikovanejší, nakoľko sa sám vývojár musí rozhodnúť, ktorá verzia zlučovanej vetvy poslúži ako ideálny predok oboch vetiev. Automatizácia procesu zlučovania vetiev zo strany distribuovaných systémov sa prejaví hlavne v prípade práce na veľkých projektoch, kedy je jedinou úlohou vývojára riešenie prípadných konfliktov. V prípade, že by vývojár spájal vetvy rovnako veľkého projektu využívajúc centralizovaný systém na správu verzií, bol by nútený ručne určiť verziu, ktorá poslúži ako najvhodnejší základ pre tento proces. Pri veľkom počte verzií však vývojár ľahko pochybí a nepodarí sa mu zvoliť vhodnú verziu. Na druhej strane distribuované systémy vždy zvolia najvhodnejšiu verziu danej vetvy. Aj tento rozdiel hodnotím veľmi pozitívne z hľadiska vývoja softvérového systému v rámci tímového projektu. Automatizácia spájania vetiev nástrojom Git šetrila čas vývojárov tímu, pričom tento ušetrený čas boli schopní využiť produktívnejšie v prospech vyvíjaného projektu.

Štvrtým zásadným pozitívom je poskytnutie plnohodnotného prístupu k výhodám systémov na správu verzií pre všetkých vývojárov. Prispievatelia, ktorí nemajú oprávnenie k modifikácií zdrojového kódu určitého projektu, v prípade že tento projekt využíva centralizovaný systém na správu verzií, majú síce možnosť dostať sa k aktuálnej verzií daného projektu, avšak nie je im umožnené odovzdanie artefaktu zdrojového kódu. Na druhej strane, distribuované systémy na správu verzií, toto odovzdávanie umožňujú, vďaka tomu, že každá kópia projektu predstavuje samostatný fungujúci lokálny repozitár, s ktorým sa dá plnohodnotne pracovať. Táto funkcionalita distribuovaných systémov sa prejavuje najmä pri projektoch s otvoreným zdrojovým kódom. Komukoľvek je umožnené napríklad dopĺňanie novej funkcionality prípadne oprava chýb zdrojových súborov projektov s otvoreným kódom. Túto novovzniknutú

verziu je možné odovzdať a nechať na zvážení pôvodného autora, či ju zahrnie do novej verzie svojho projektu. Alternatívnym dôsledkom je vznik úplne nového projektu. Rovnaké dôsledky je možné dosiahnuť aj využitím centralizovaných systémov. Cesta k týmto cieľom je však omnoho tŕnistejšia ako je tomu v prípade distribuovaných systémov.

Posledným dôvodom je podpora atomických zmien. Vývojárovi je umožnené odovzdať artefakt zdrojového kódu len v prípade, že týmto odovzdaním nedôjde k žiadnemu konfliktu. Zjednodušene by sa dalo povedať, že atomické odovzdávanie artefaktu zdrojového kódu má za následok odovzdanie buď všetkých zmien vykonaných vývojárov alebo sa nevykonajú v repozitári žiadne zmeny, až pokým nie sú vyriešené všetky konflikty. Atomické odovzdávania artefaktov slúžia primárne na udržanie konzistencie repozitára. Ako príklad uvediem prípad, kedy vývojár vytvára novú funkcionalitu. Táto funkcionalita pozostáva z modifikácie dvoch existujúcich zdrojových súborov, pričom tieto sú na sebe závislé. V prípade, že by pri odovzdávaní artefaktu došlo v jednom zo súborov ku konfliktu a proces odovzdávania by nebol atomický, repozitár by sa dostal do stavu, kedy by sa nová funkcionalita v jednom zo súborov dopytovala na zdrojový kód v druhom súbore, ktorý kvôli vzniknutému konfliktu nebol odovzdaný. Túto funkcionalitu hodnotím taktiež veľmi pozitívne, nakoľko nám umožňovala udržiavanie konzistentného stavu repozitára.

Aj úspešné systémy však majú svoje muchy

Najzásadnejšie klady distribuovaných systémov som rozobral v predchádzajúcej časti. A pozitíva centralizovaných systémov som naznačil v časti opisujúcej problematiku výberu vhodného systému. Dá sa teda povedať, že oba spomínané systémy majú svoje nesporné výhody, voľba ktoréhokoľvek z nich však so sebou prináša i nezanedbateľné množstvo nevýhod. Nadnesene by sa dalo povedať, že výber vhodného systému na správu verzií spočíva v zostavení zoznamu záporov oboch systémov pričom ten, ktorý ich bude mať v konečnom zúčtovaní menej je označený za víťaza.

Pri využití distribuovaného systému sa môže projekt dostať do stavu, kedy nie je jasné, kto má najnovšiu verziu projektu. Tento systém je pomerne zložitý [4] a vzniká problém pri riadení prístupu k určitým častiam projektu [1]. Takisto si dovolím polemizovať o pravdivosti tvrdenia, že medzi hlavné výhody tohto systému patrí možnosť práce bez pripojenia k Internetu. Od vývojárov projektu totiž musíme vyžadovať neustálu aktualizáciu verzie, s ktorou pracujú. V opačnom prípade by sa ľahko mohlo stať, že niekto bude pracovať s príliš starou verziou a neúmyselne zaháji tvorbu akéhosi podprojektu. Ako príklad uvádzam situáciu, kedy tím ľudí pracuje na projekte. Jeden z členov tímu odíde napríklad na niekoľko dní do zahraničia, kde mu chýba pripojenie k sieti Internet. Chce však na projekte pracovať aj naďalej a z toho dôvodu si pred odchodom zaktualizuje svoju lokálnu verziu projektu. Zvyšok členov tímu bude v dobe absencie tohto člena ďalej pracovať na projekte. Z pochopiteľných dôvodom je komunikácia s členom tímu, ktorý sa nachádza v zahraničí, minimálna alebo v horšom prípade dokonca žiadna. Po návrate tohto člena sa však zistí, že v konečnom dôsledku vznikli dve rozdielne verzie projektu, pretože osamotený vývojár mohol pracovať na takej funkcionalite, ktorú sa rozhodol zvyšok tímu medzičasom napríklad úplne vynechať alebo nejakým spôsobom

modifikovať. Z tohto uhľa pohľadu je vývoj bez pripojenia k sieti Internet prakticky nemožný.

Väčšina záporov, ktorými oplývajú centralizované systémy, je zjavných, preto spomeniem v krátkosti len niektoré z nich. Môže dôjsť k spomaleniu vývoja v prípade, že jeden vývojár uzamkne určitú časť projektu. Ďalším problémom je samotná centralizácia. V prípade, že dôjde k výpadku centrálneho servera, prakticky sa tým znemožní akákoľvek práca na projekte. Takisto hrozí strata údajov, ak neboli riadne zálohované. Centralizované systémy na rozdiel od distribuovaných, ktoré uchovávajú len množinu zmien medzi jednotlivými verziami, archivujú celé súbory, v prípade, že boli modifikované. Toto môže mať nepriaznivý dopad obzvlášť pri vývoji veľkých softvérových systémov s množstvom zdrojových súborov. V takomto prípade totižto režijný adresár repozitára môže nadobudnúť doslova gigantické rozmery. Pri veľkých projektoch môže byť tento rozdiel v porovnaní s distribuovanými systémami aj viac ako desaťnásobný. Ďalším zásadným nedostatkom centralizovaných systémov je absencia podpory atomických odovzdaní artefaktov zdrojového kódu. Posledným a azda najzásadnejším negatívom, ktoré spomeniem je prílišná komplikovanosť spájania vetiev v centralizovaných systémoch.

Hudba budúcnosti

Ako som už naznačil, súčasné najpopulárnejšie systémy na správu verzií majú pomerne veľa nedostatkov. Veľkú časť z nich je však možné ospravedlniť, nakoľko samotný manažment verzií je relatívne mladé odvetvie, ktoré je ešte stále vo fáze vývoja. Je však otázne akým smerom sa budú tieto systémy v najbližšej budúcnosti uberať. V tomto zmysle sa mi páči myšlienka projektu CoRed [3]. Tento projekt využíva možnosti, ktoré prináša dnešná doba, kedy sú kapacity zdieľania prostriedkov pomocou siete (angl. *cloud computing*) zdanlivo nevyčerpateľné. Projekt CoRed spočíva v tom, že vyvíjaný projekt je uchovávaný na jedinom mieste, a to na serveri zdieľania prostriedkov (angl. *cloud*). Takýchto webových integrovaných vývojových prostredí je v súčasnosti prístupných hneď niekoľko. Ďalšími alternatívami sú napríklad Koding[1] alebo Cloud 9 IDE[2].

Integrované vývojové prostredia bežiace na serveroch zdieľania prostriedkov predstavujú podľa môjho názoru budúcnosť vývoja softvéru. V najbližšom období totižto bude neustále klesať problém so získaním prístupu k sieti Internet. Integrované vývojové prostredia poskytované prostredníctvom serverov zdieľania prostriedkov ponúkajú svojim používateľom hneď niekoľko výhod. Prvou výhodou je zníženie nárokov na zariadenie, ktoré používateľ využíva na prístup k týmto vývojovým prostrediam. Jediné čo v takomto prípade používateľ potrebuje je pripojenie k sieti Internet a zariadenie, ktoré mu umožňuje písanie zdrojového kódu. Môže sa teda jednať o akékoľvek zariadenie, či už osobný počítač alebo mobilný telefón. Ďalšou výhodou je minimalizácia pamäťových nárokov, nakoľko repozitár, obsahujúci zdrojový kód vyvíjaného projektu, sa nachádza na serveri zdieľania prostriedkov.

Za nedostatok, ktorý považujem pri aktuálne existujúcich webových integrovaných vývojových prostrediach za kritický, by som označil nedostatočnú podpory správy verzií

[1] https://koding.com/
[2] https://c9.io/

vyvíjaného zdrojového kódu. Väčšina dnešných klasických aplikácií integrovaných vývojových prostredí má túto problematiku pomerne dobre vyriešenú. Mnohé prostredia umožňujú jednoducho, niekoľkými kliknutiami vykonať odovzdanie artefaktu zdrojového kódu, spájanie vetiev a podobne. V tomto smere majú vývojové prostredia poskytované prostredníctvom serverov zdieľania prostriedkov ešte omnoho väčší potenciál, nakoľko už len z toho dôvodu, že uchovávajú zdrojové súbory softvérového systému na serveri zdieľania prostriedkov, má pre ne význam tento zdrojový kód aj spravovať a verziovať. V prípade nápravy tohto nedostatku, nepochybujem o tom, že ich popularita bude v nasledujúcich rokoch narastať. A to najmä z dôvodu nemalého zoznamu mnou spomínaných výhod, ktoré integrované vývojové prostredia bežiace na serveroch zdieľania prostriedkov poskytujú.

Záver

Systémy na správu verzií delíme z hľadiska architektúry na distribuované a centralizované. Okrem manažmentu verzií zdrojových súborov softvérových systémov je však možné verziovacie systémy využiť aj na rôzne iné účely. Je možné ich využiť napríklad ako učebnú pomôcku, prípadne môžu byť nápomocné pri písaní dokumentov. Pri vývoji softvérových systémov však pri výbere vhodného systému na správu verzií zohrávajú rolu iné aspekty. V tomto smere sú na tom lepšie distribuované systémy, nakoľko podporujú okrem iného atomické odovzdania artefaktov zdrojového kódu alebo čiastočne automatizujú proces spájania dvojice vetiev. Napriek množstvu výhod majú distribuované, ale aj centralizované systémy nezanedbateľné množstvo nevýhod. Z toho dôvodu sa esej zaoberá aj alternatívnou ideou, ktorá by existujúce systémy na správu verzií mohla posunúť o malý kúsok vpred, v ústrety budúcnosti.

Použitá literatúra

1. O'Sullivan, B.: Making sense of revision-control systems. *Communications of the ACM* (2009). New York: ACM, vol. 52, no. 9, p. 56.
2. de Alwis, B., Sillito, J.: Why are software projects moving from centralized to decentralized version control systems? *ICSE Workshop on Cooperative and Human Aspects on Software Engineering*, (2009). Washington DC: IEEE Computer Society, p. 36–39.
3. Mikkonen, T., Nieminen, A.: Elements for a cloud-based development environment: online collaboration, revision control, and continuous integration. *Proceedings of the WICSA/ECSA* (2012). New York: ACM, p. 14–20.
4. Clatworthy, I.: Distributed version control system – why and how. *Proceedings of OSDC* (2007).

Annotation

A Tale of Two Revision Systems and their Future

Most of the software systems developed today are dependent on the use of version control systems. Version management is considered to decide success of projects, especially the large ones. This is due to increasing complexity of the source code of software systems. Therefore, the choice of a suitable tool for the version management is a crucial task. All tools for version management are characterized by two revision systems - centralized and distributed version control systems. This essay analyzes these systems, describes the selection of suitable version control system and offers an alternative idea to eventually supplement or replace them.