VÝBER SPRÁVNEJ CESTY V MANAŽMENTE VERZIÍ SOFTVÉRU

Je lepšie venovať viac času správnemu výberu, než sa neskôr zaoberať zbytočnými problémami.

Pavel Michalko

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

Abstrakt. *Vývoj softvéru je v dnešnej dobe podmienený využitím viacerých ľudí. Rozsiahlosť niektorých projektov si vyžaduje, aby títo ľudia pracovali na vývoji súčasne. Tým však vzniká problém s manažovaním vytváraných zdrojových kódov. V takýchto prípadoch je nutné použiť podporné prostriedky pre správu verzií. Táto esej sa zaoberá otázkou výberu správneho typu softvéru pre správu verzií. Spôsob manažovania zdrojových kódov sa líši najmä v závislosti na zvolenom type vyvíjaného softvéru. Konkrétne sa preto táto esej zaoberá výberom najvhodnejšieho prístupu verziovacieho softvéru v závislosti na type projektu. Tento výber dosahuje porovnaním niektorých prístupov a zároveň demonštrovaním problémov, ktoré môžu v jednotlivých prípadoch nastať.*

Kľúčové slová: *manažment verzií, prístupy softvéru pre správu verzií, verziovanie, verziovanie v softvérových projektoch*

Prečo, kedy a ako používať softvér pre správu verzií?

Vytvorenie softvéru v dnešnej dobe znamená zainteresovanie väčšieho množstva ľudí do jeho vývoja. To so sebou prináša výhody ako aj nevýhody. Výhodou je využitie viacerých ľudských zdrojov súčasne. Znamená to, že viacerí ľudia môžu paralelne pracovať na nejakej časti zdrojového kódu. Toto však so sebou prináša problém s manažovaním týchto

ľudí. Jedným z mnohých problémov pri vývoji softvéru je teda potreba uchovávať a spravovať jednotlivé kroky implementácie.

Na to nám slúžia podporné prostriedky (softvéry) pre správu verzií. Kľúčom k úspechu je výber správneho softvéru pre správu verzií podľa toho, ktoré požiadavky musí pre nás primárne spĺňať. V niektorých prípadoch môžeme od takéhoto softvéru vyžadovať aj podporu v ostatných oblastiach vývoja. Môže zohrať dôležitú úlohu napríklad pri sledovaní postupu vývoja. Je teda na nás, aby sme zohľadnili, čo od takéhoto softvéru vyžadujeme. Existuje totižto viacej typov a prístupov k správe verzií. Každý z týchto prístupov je vhodnejší pre určitý spôsob vývoja alebo typ softvéru.

Spôsob manažovania zdrojových kódov sa líši najmä v závislosti na zvolenom type vyvíjaného softvéru. Tieto softvérové projekty môžeme z hľadiska významu softvéru pre správu verzií rozdeliť do niekoľkých názorných skupín (pozri Tab. 1).

Tab. 1. Názorné rozdelenie softvérových projektov.

Počet ľudí v tíme	Rozsah zdrojových kódov	Štruktúra softvéru
Malý tím (menej ľudí v tíme)	Menší počet rozsiahlejších tried (obsahujúce viac metód)	Viac nezávislých modulov (častí, ku ktorým môžeme priradiť jedného autora)
Veľký tím (viacčlenný tím)	Väčší počet menších tried (obsahujúce napríklad jednu metódu)	Jeden väčší modul (viacero autorov pracuje na rovnakej časti)

Ako si teda zvoliť čo najvhodnejší typ takéhoto podporného softvéru? Na začiatku sa musíme rozhodnúť, či je pre náš projekt vhodnejší centralizovaný alebo distribuovaný prístup.

Kedy si vybrať centralizovaný prístup?

Centralizovaný prístup vychádza z architektúry klient-server, kde je použité jediné spoločné úložisko. V tomto prípade si používatelia môžu ľubovoľne sťahovať obsah tohto úložiska. Právo na zápis má však len vybraná skupina ľudí. [2]

Jeho použitie, práve z dôvodu potreby prístupových práv, je vhodné len v menších tímoch. Ak si používateľ stiahol zdrojový kód, tak po pridaní nejakej funkcionality musí vykonať niekoľko ďalších krokov. Najskôr musí vyhľadať osobu s právom na zápis do centrálneho repozitára. Potom sa s ňou musí skontaktovať a svojpomocne odovzdať upravené súbory. Až potom ich táto zodpovedná osoba môže sprístupniť pomocou centrálneho repozitára. Vo väčších tímoch by to znamenalo značné spomalenie vývoja a zbytočné komplikácie. Prináša takýto prístup vôbec nejakú výhodu?

Najväčšia výhoda je kvalita a úroveň zdrojových kódov uložených v centrálnom úložisku. Ľudia, ktorí disponujú právom na zápis, sú väčšinou odborníci vo svojom obore. Preto, keď všetky zdrojové kódy musia prejsť popod ich ruky, minimalizuje sa šanca výskytu chybného alebo nekvalitného kódu.

Je však výhoda zabezpečenia kvality postačujúca? Používatelia totižto musia byť stále pripojení ku sieti. Toto je skutočne veľké obmedzenie v dnešnej dobe, kde platí „čas sú peniaze“. Predstavme si napríklad situácie, kde internetové pripojenie nie je dostupné.

Vývojár je tak silne obmedzený napríklad pri cestovaní a využitie jeho ľudských zdrojov nie je maximalizované.

Môj názor je preto veľmi dobre zvážiť využitie takéhoto prístupu. V súvislosti s pripojením nejde o najšťastnejšie riešenie a pôsobí tak neflexibilne a zastaralo. Veď podporný softvér by mal v čo najväčšej miere prácu uľahčovať. Obmedzenie, v podobe potreby neustáleho pripojenia, má presne opačný efekt a častokrát vývojárovi skôr zväzuje ruky. V dnešnej dobe je asi lepšie siahnuť po inom riešení.

Kedy zvoliť distribuovaný prístup?

Najväčšia výhoda distribuovaného prístupu je upustenie od potreby neustáleho sieťového pripojenia. Okrem nej prináša ako modernejší nástupca samozrejme aj mnoho iných výhod.

Jeho použitie je vhodné v menších, ale aj vo väčších tímoch. Tento prístup je veľmi flexibilný a vyhovujúci za každých podmienok. [1]

Ďalšími nespornými výhodami sú viacnásobná záloha, dostupnosť, spoľahlivosť a podobne. Avšak výhodou takéhoto prístupu je častokrát označovaná výkonnosť, ako napríklad v [3]. Tu však musím s autorom nesúhlasiť. Je skutočne potrebné sa niečím takýmto zaoberať? Podľa môjho názoru otázka výkonu a trvania jednotlivých operácií je v niektorých prípadoch zanedbateľná. Ide predsa o operácie nad textovými súbormi, ktorých súčet veľkostí je rádovo niekoľko megabajtov. Pri práci s takýmito malými súbormi je v podstate nemožné nejaké rozdiely postrehnúť. Niekto môže síce namietať, že v prípade distribuovaného prístupu sa jedná o operácie na lokálnom disku. Avšak systém sa používa ako výpomoc vo viacčlennom tíme. Ak chceme náš výtvor sprístupniť ostatným, sme tak či tak nútení niekedy použiť sieťové spojenie. Otázka výkonnosti softvéru pre správu verzií by preto podľa môjho názoru nemala mať veľkú váhu pri rozhodovaní. Je lepšie sa sústrediť na ostatné ponúkané výhody.

Tak, či onak, je v dnešnej dobe podľa mňa použitie distribuovaného prístupu jasná voľba. Centralizovaný prístup je skutočne zastaraný. Prečo teda nepoužiť novší a vylepšený spôsob, keď je možnosť?

Spôsoby riešenia viacnásobného prístupu

Využitie softvéru na automatické manažovanie verzií zdrojových kódov, pri vyvíjaní väčším počtom ľudí, však so sebou prinieslo aj niekoľko problémov. Jedným z nich bolo vyriešenie spôsobu prístupu viacerých ľudí k jednému súboru súčasne. Je preto dôležité sa v ďalšom kroku rozhodnúť pre ten najlepší spôsob v závislosti na type projektu.

Predstavme si, že pracujeme na úprave nejakého zdrojového kódu. Súčasne s nami však chce v tom istom súbore vykonať zmeny niekto iný. Ako vyriešiť tento problém k všeobecnej spokojnosti? Môžeme rozlíšiť dva základne spôsoby prístupu.

Snáď najjednoduchším riešením problému viacnásobného prístupu k súboru je použiť uzamknutie súboru. Človek, ktorý chce modifikovať nejaký súbor, ho najskôr „uzamkne“ a po vykonaní zmien ho „odomkne“ (umožní prístup pre ostatných).

Druhým je spôsob paralelných prístupov. Takéto riešenie viacnásobného prístupu umožňuje viacerým ľuďom vykonávať zmeny v tom istom súbore v rovnakom čase.

Jednoznačne tým zefektívňuje prácu a umožňuje lepšie využitie času. Namiesto čakania na sprístupnenie potrebného súboru sa môžeme plne sústrediť na samotný vývoj.

Každý z týchto spôsobov je vhodné použiť pri inom type projektu. V akých situáciách je ktorý spôsob vhodné použiť?

Pre ktorý spôsob sa rozhodnúť?

Keď uvažujeme situáciu, kde sa na vývoji podieľa menší tím, zdá sa ako postačujúce riešenie použitie uzamykania súborov. Šanca, že k tomu istému súboru budú súčasne pristupovať viacerí ľudia, je veľmi nízka. S týmto tvrdením však treba narábať veľmi opatrne. Nemôžeme sa na to s istotou spoliehať a je lepšie vždy rátať s najhorším. Tento spôsob by som odporúčal zvoliť len v prípade skutočne malého tímu a to v počte 1-2 ľudí.

Čo keď na projekte pracuje väčší počet ľudí? Jeden člen tímu síce pracuje, ale čo ostatní? Nie je tu maximálne využitá ľudská sila. V najhoršom prípade sa môže stať, že človek, ktorý zamkol súbor, ho zabudol odomknúť. Takto prídeme o veľa času, ktorý mohol byť využitý pre vývoj. V tomto prípade je určite lepšie zvoliť spôsob paralelných prístupov. To nám pomôže zefektívniť prácu a využiť skutočné výhody softvéru pre správu verzií.

Adaptujme tento problém výberu na softvér, z hľadiska rozsahu zdrojových kódov. V projekte, ktorý pozostáva z menšieho počtu rozsiahlejších tried, nastáva pri použití uzamykania súborov problém aj v menších tímoch. Zväčšuje sa totižto pravdepodobnosť, že dvaja rozdielni ľudia chcú súčasne upravovať jeden súbor. Pri zvolení paralelných prístupov ide vo väčšine prípadov o úplnú elimináciu tohto problému. Kým jeden človek upravuje v súbore (triede) nejakú časť (metódu), iný človek môže súčasne upravovať v tom istom súbore jeho inú časť (inú metódu). Po vykonaní zmien nie je problém s ich spojením. Výsledkom je jeden funkčný zdrojový kód. Takto to vyzerá vo väčšine prípadov. Ani s použitím takéhoto spôsobu sa však nedá úplne vylúčiť potreba ľudského zásahu. Väčšina softvérov sa síce snaží túto potrebu minimalizovať. Niekedy však nastanú prípady, s ktorými si softvér sám jednoducho neporadí. Ako príklad môže poslúžiť úprava toho istého súboru viacerými ľuďmi na tom istom mieste. Softvér nevie vybrať, ktorá časť od ktorého autora je nakoniec tá správna. Dokáže len tieto konfliktné časti označiť a prezentovať používateľovi. Ten nakoniec musí ručne súbor upraviť do požadovaného tvaru. Je to síce robota navyše, ale tieto prípady sú skutočne zriedkavé.

Naopak, pri projektoch, ktoré pozostávajú z viacerých menších tried, je použitie spôsobu uzamykania súborov jasná voľba. Veľmi často by sa stávalo, že pri prístupe k tomu istému súboru by používatelia upravovali rovnakú časť. Takto by vznikali konflikty, ktoré sa automaticky nedajú vyriešiť. Uzamykanie súborov v tomto prípade rieši všetky problémy.

Podobné je to pri projektoch z hľadiska štruktúry vyvíjaného softvéru. V prípade, že vytváraný softvér pozostáva z viacero modulov, je použitie uzamykania súborov tiež možné. Každý člen tímu totiž pracuje na „svojom piesočku“ a zodpovedá za svoju časť zdrojových kódov. Môže si tak svoje súbory ponechať zamknuté v čase editácie bez toho, aby obmedzoval niekoho iného.

Iné by to bolo, keď by ku modulu museli mať prístup viacerí ľudia. V tíme sa môže napríklad nachádzať osoba zodpovedajúca za vývoj alebo opravu zdrojových kódov. Táto

osoba potrebuje taktiež pristupovať ku zdrojovým kódom. Tu teda môže vzniknúť konflikt v prístupe autora a zodpovednej osoby. V takomto prípade je vhodnejšie zvoliť spôsob paralelných prístupov.

Záver

Použitie softvéru pre správu verzií prináša značné výhody. Pri vývoji softvéru je to určite nemalá pomoc. Aby sa však jeho potenciál využil v plnej miere, je potrebné najskôr dobre zvážiť, aký typ alebo prístup verziovacieho softvéru použiť. Každý softvérový projekt má svoje špecifiká. Práve ich analýzou môžeme dospieť k správnemu výberu pomocného softvéru. Je predsa len lepšie venovať trochu viac času výberu, než potom zistiť, že zvolený pomocný softvér nie je vôbec pomocný, ale práve naopak.

Nesmieme však zabúdať, že ani takýto softvér nikdy nemôže eliminovať tú najdôležitejšiu vec - komunikáciu. Tá stále ostáva základom pri vyvíjaní softvéru viacerými ľuďmi.

Použitá literatúra

1. Alwis B., Sillito J.: Why are software projects moving from centralized to decentralized version control systems? In: *Proceedings of the 2009 ICSE Workshop on Cooperative and Human Aspects on Software Engineering (CHASE '09)*. IEEE Computer Society, Washington (2009), DC, USA, 36-39.
2. Kuhn, D.: Distributed Version Control Systems. [Online] [Dátum: 25. November 2011.] http://www.ayuna.de/content/publications/DistributedVCS_Daniel_Kuhn_2010.pdf.
3. Otte, S.: Version Control Systems. [Online] [Dátum: 25. November 2011.] http://cst.mi.fu-berlin.de/teaching/WS0809/19565-PS-TI/reports/otte09version.pdf.

Annotation

Choosing the right way in the software version management

This paper explains the procedure for selecting the most suitable software for version control. For a few selected divisions of software projects is shown the applicability of each approach. It points to possible problems in the selection of inappropriate access of versioning system. In some cases, is expressed the author's own stance on issues and problems arising. The aim is to convince the reader about using software version control and the proper selection depending of the type of software project.