AUTOMATIZOVAŤ A ČI NEAUTOMATIZOVAŤ, TAK ZNIE OTÁZKA

Prečo robiť niečo ručne ak to za Vás môže urobiť stroj a bohužiaľ aj v lepšej kvalite?

Jozef Krajčovič

Slovenská technická univerzita
Fakulta informatiky a informačných technológií
Ilkovičova 3, 842 16 Bratislava
Xkrajcovicj3[zavináč]stuba[.]sk

Abstrakt. *S prudkým rozvojom informačných a komunikačných technológií sa stal vývoj softvéru čoraz komplexnejšou záležitosťou, ktorá v sebe zahŕňa veľké množstvo rôznorodých procesov a požiadaviek. To podnietilo firmy k stimulu dôslednej reorganizácie manažmentu dostupných zdrojov a k efektívnejšiemu využitiu podporných prostriedkov, ktoré sú použité v procese vývoja softvéru. Jedným s možných spôsobov ako zefektívniť proces vývoja softvéru je zautomatizovať rutinné úlohy, ktoré sú vykonávané väčšinou ručne, čo do istej miery ovplyvňuje celkovú efektivitu práce a tým pádom taktiež kvalitu produktu. V tejto eseji sa zaoberám nasledujúcimi otázkami. Môže automatizácia zostavovania a nasadzovania softvéru zvýšiť efektivitu vývojového tímu? Je využitie automatizácie prínosom aj v malých tímoch? Pri odpovediach na tieto otázky vychádzam z vlastného subjektívneho názoru na základe mojich skúseností s podpornými prostriedkami.*

Kľúčové slová: *manažment podpory vývoja, podporné prostriedky, automatizované zostavovanie a nasadzovanie softvéru*

Úvod

Vývoj softvéru je dynamicky sa rozvíjajúcou oblasťou, ktorá vyžaduje stále viac a viac nárokov na kvalitu produktu, preto je potreba tieto procesy čo možno najefektívnejšie riadiť.

Jednou z možností ako toho dosiahnuť je použitie podporných nástrojov v rámci procesu vývoja softvéru, aby sme efektívnejšie využili všetky dostupné zdroje, ktoré máme k dispozícii a vo výraznej miere ovplyvnili kvalitu produktu.

Podporné prostriedky v procese vývoja softvéru

V súčasnej dobe každý vývojový tím, malý alebo veľký využíva podporné prostriedky v procese vývoja softvéru veď bez toho to ani nejde, že?

Dnes máme k dispozícii veľa nástrojov pre podporu vývoja či už to komerčných alebo open-source, ktoré pomáhajú manažérom, ale aj ostatným členom projektového tímu efektívnejšie riadiť, zlepšovať a vykonávať jednotlivé úlohy v rámci jednotlivých fáz projektu od plánovania, analýzy, implementácie, testovania až po nasadenie výsledného produktu do prevádzky. Pod pojmom podporné prostriedky mám na mysli nástroje ako napríklad integrované vývojové prostredie, manažment verzií súborov, zostavovanie a nasadzovanie softvéru a tak ďalej [1].

Dôležitým faktom každej úspešnej firmy je samozrejme správny výber podporných nástrojov, ktoré chce firma integrovať do svojho vývojového procesu. Vzhľadom na to, že na trhu je k dispozícii veľa podporných nástrojov je dôležité si najskôr objasniť, čo od podporných nástrojov očakávame a nezabúdať nato, že každý nástroj má svoje určité obmedzenia.

Pri výbere vhodných podporných nástrojov treba brať do úvahy niekoľko faktorov ako napríklad či sa jedná o veľký tím alebo malý tím, platforma pre ktorú sa vyvíja produkt a samozrejme tým najdôležitejším je dostupnosť finančných prostriedkov firmy.

Čo by však podľa mňa mal spĺňať každý dobrý podporný nástroj je aby vedel odbremeniť členov tímu od rutinných úloh, ktoré je potreba vykonávať ručne a umožnil tieto úlohy čiastočne zautomatizovať. Ďalej zefektívnil komunikáciu tímovej spolupráce a zrozumiteľným spôsobom prezentoval informácie, ktoré by boli jednoducho dostupné pre každého člena tímu. Upozorňoval nás na kritické situácie a umožnil by nám včas a adekvátne reagovať. Hlavne aby tieto nástroje neboli typu ad-hoc t.j. použiteľné len pre jeden typ projektu.

Môžeme mať síce kvalitné podporné nástroje, ale ak nie sú správne použité v procese vývoja tak sa znižuje ich efektívnosť. Preto treba vybrať také nástroje, ktoré čo najlepšie aproximujú k interným procesom danej firmy.

Vo väčšine prípadov firmy nevyužívajú všetky výhody, ktoré im podporné prostriedky ponúkajú. Jednou z nich je práve zautomatizovanie rutinných činnosti, ktoré vykonávajú členovia tímu ručne. Pri aplikovaný automatizácie do procesu vývoja by sa mohla zvýšiť efektivita práce až o 50 % čo sa odrazí na kvalite produktu. Pod rutinnými činnosťami mám na mysli automatizované zostavovanie a nasadzovanie softvéru, automatizované testy, automatizované generovanie dokumentácie zo zdrojového kódu a automatizovaná vizualizácia výsledkov [2].

Práve automatizáciu väčšina tímov podceňuje a neprikladá jej veľký význam. Vo väčšine prípadov sa jedná o menšie tímy v rozsahu cca 6-10 členov, ktorý nepovažujú zautomatizovať svoje činnosti za prioritné čo podľa môjho názoru nie je vhodné riešenie v dnešnej dobe ak chce byť firma úspešná na trhu a produkovať kvalitný produkt.

Automatizované zostavovanie a nasadzovanie softvéru

Ako som už v predošlej kapitole naznačil automatizácia vo vývojovom procese podľa mňa hra veľmi dôležitú úlohu a má nezanedbateľný vplyv na zvyšovanie efektivity práce či už v malých alebo veľkých tímoch.

V tejto kapitole popisujem podľa mňa najdôležitejšiu časť procesu vývoja softvéru, ktorú by mal každý tím jednoznačné integrovať. A je to automatizované zostavovanie a nasadzovanie softvéru.

Pod pojmom zostavovanie a nasadzovanie softvéru si môžeme predstaviť nejaký sled činností. Týmito činnosťami sú kompilácia rôznych častí zdrojového kódu do binárneho formátu, linkovanie t.j. spojovanie jednotlivých binárnych súborov a ďalšie kroky potrebné k vytvoreniu použiteľnej aplikácie, ako je zostavenie inštalačného programu a umiestnenie výsledného produktu na zdieľaný server.

Až potom čo zostavia všetky častí aplikácie, bude možne povedať či aplikácia funguje správne a či robí to, čo sa od nej očakáva. Zostavovanie jednotlivých častí aplikácie je pomerne zložitý proces, časovo náročný a náchylný na chyby.

Avšak väčšina firiem nevyužíva možnosti automatizovaného zostavovania a nasadzovania softvéru, čo je podľa mňa veľmi veľká chýba. Tieto firmy využívajú klasický scenár, kedy jeden alebo dvaja členovia tímu sú zodpovedný za zostavovanie a nasadzovanie softvéru, ktoré vykonávajú ručne. Nevýhoda tohto scenára je vysoká závislosť na týchto zamestnancoch, slabá transparentnosť vývoja, slabá komunikácia s ostatnými členmi tímu. Pri používaní tohto klasického scenára je pravdepodobnosť výskytu chýb veľmi vysoká, čo samozrejme zvyšuje náklady na vývoj.

Predstavte si príklad že by ste urobili zmeny v zdrojovom kóde nejakej aplikácie, ktorú dáte zostaviť v svojom *IDE*. Všetko beží na vašej pracovnej stanici v poriadku a žiadne chyby sa nevyskytli. Po čase zistite, že je váš kód nefunkčný po zmenách, ktoré vykonali vaši kolegovia v tíme a prehlásite legendárnu vetu *"Ahoj, ale na mojom stroji to funguje!"*. A nastáva ta nepríjemná situácia a vy musíte hľadať príčiny výskytu chýb a pritom už pracujete na novej funkcionalite. To asi nie je príjemná situácia, že?

Práve preto je vhodné do toho procesu zaviesť používanie automatizovaného zostavovania a nasadzovania softvéru, ktoré má obrovské výhody:

1. Zlepšenie kvality výrobkov (šetrenie času a peňazí)
2. Eliminácia a rýchlejšia identifikácia chýb
3. Eliminácia závislosti na kľúčových zamestnancoch
4. Maximalizácia toku informácií medzi členmi tímu
5. Rýchla a promptná reakcia v prípade vzniku problému

Zavedením automatizácie do svojho procesu vývoja budete schopný integrovať včas a častejšie a rýchlejšie testovať svoj produkt, čím sa znižuje frekvencia výskytu chýb a nákladov na vývoj.

Automatizované zostavovanie môžeme rozdeliť do nasledujúcich kategórií podľa toho kedy sa zostavovanie uskutočňuje [3]:

1. Kontinuálna integrácia – umožňuje zostavovať softvér pri každom nahlásení zmien (*check-in*) do systému riadenia verzií (*git, svn*)

2. Dávkové zostavovanie – ide o podobný princíp ako pri kontinuálnej integrácií ale s tým rozdielom, že zostavovanie softvéru začne po nejakom stanovenom čase napríklad po 1. hodine od odovzdania zmien
3. Denné zostavovanie – zostavovanie prebieha každý deň v presne stanovený čas, vo väčšine prípadov ide o nočné zostavovanie napríklad o 22:00 hodine

Vo väčšine prípadov s ktorými som sa doteraz stretol to vyzerá tak, že firmy využívajú najčastejšie denné zostavovanie, na druhom mieste kontinuálnu integráciu, dávkové zostavovanie sa vyskytuje vo firmách iba zriedka.

Typický priebeh zostavovania softvéru

Ako so spomenul v predošlej kapitole najčastejší spôsob zostavovania softvéru je denné zostavovanie. Vo väčšine firiem prebieha denné zostavovanie softvéru vo večerných hodinách napríklad o 22:00 hodine, kedy sa spustí automatizovaný skript.

Tento skript zistí zmeny, ktoré boli vykonané v repozitároch jednotlivých členov tímu v systéme pre riadenie verzií. Ak teda zistí nejaké zmeny, nasleduje proces zostavovania zdrojových kódov do binárneho tvaru. Keď je produkt zostavený, nasledujú overovacie testy (*BVT – build verification tests)*, ktoré odhalia chyby, ktoré sa zaznamenajú v systéme a taktiež sa automatický odosielajú e-mailové notifikácie členom tímu, ktorý sú zodpovedný za výskyt danej chyby. Ak sa nevyskytujú chyby v procese zostavovania nasleduje automatické vygenerovanie dokumentácie zo zdrojového kódu. A nakoniec keď je zostavená funkčná verzia softvéru je inkrementálne vytvorená nová vetva v systéme pre riadenie verzií

Na druhý deň keď členovia tímu prídu do práce, prezrú si výsledky denného respektíve nočného zostavenia v danom systéme, ktorý poskytuje reporty. Ak je zaznamenané v systéme, že sa vyskytli chyby t.j. nastalo takzvané „*rozbité zostavenie*“, členovia tímu a hlavne ten kto môže za rozbité zostavenie vykoná príslušné nápravy.

V softvérovom priemysle existuje tradícia, že ten člen tímu, ktorý rozbil zostavenie je patričné potrestaný ako napríklad nosenie čapice hanby alebo mu kolegovia umiestňujú posmešné ceduľky na dvere a podobne. Všetko samozrejme v rámci humoru, hlavne ak ide o syntaktické chyby, nedodržanie štandardu kódovania (*syntaktická analýza*) a podobné chyby, ktoré boli spôsobené v rámci nepozornosti.

Automatizované zostavovanie softvéru podľa môjho subjektívneho názoru nie lenže umožňuje automatizovať rutinné činnosti, ktoré by sme museli inak vykonávať ručne a zdĺhavo, ale na základe reportov a záznamov môžeme predikovať určité metriky. Tieto metriky môžeme použiť pri zlepšovaní procesov ako napríklad určovanie vplyvu refactoringu kódu alebo iných aspektov, ktoré môžu zefektívniť vývojový proces.

Záver

Využitie podporných prostriedkov pre automatizáciu zostavovania a nasadzovania softvéru by mala určite integrovať každá firma, ktorá chce zefektívniť svoj proces vývoja a tým pádom dodávať kvalitnejší produkt, čím si zaručí úspech a postavenie na trhu. Určite to neplatí iba pre veľké firmy ale aj pre malé firmy, ktoré sa možno zavedením

automatizácie do svojho vývojového procesu stanú konkurencie schopnými proti veľkým firmám.

Použitá literatúra

1. Bashar N.: Meta CASE Support for Method-Based Software. In: *Proc. of 1st Int. Congress on Meta-CASE*, 5-6th January 1999, Sunderland, UK Development Dostupné na: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.51.9116>.
2. José M.: The Software Factory: Integrating CASE Technologies to Improve Productivity. Dostupné na:
   <http://dspace.mit.edu/bitstream/handle/1721.1/1658/ 96_02.pdf?sequence=2>.
3. Mickey G. et al: Profesionálne riadenie životného cyklu aplikácií, Indiana: Wiley Publishing Inc., 2010. 412 s. ISBN 978-80-7413-102-8.

Annotation

Automation or no automation, this is the question?

With the rapid development of information and communication technology, software development has become increasingly more complex matter, that includes a large number of disparate processes and requirements. This has prompted the company to reorganize management of available resources and efficient use of support tools that are used in the software development process. One with possible ways how to efficient the software development process is to automate routine tasks, which are mede mostly by hand, what influences to quality of the product. In this essay will address the following questions. Can automation build of the software to increase the efficiency of the development team? Use of automation is also good for small teams? Answers on these questions in my subjective opinion and based on my experience with the supporting tools.