BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF INTELLIGENT SYSTEMS
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

CONTINUOUS INTEGRATION AND AUTOMATED CODE REVIEW IN OPEN SOURCE PROJECTS

PRŮBĚŽNÁ INTEGRACE A AUTOMATIZOVANÁ KONTROLA KÓDU V PROJEKTECH S OTEVŘENÝM ZDROJOVÝM KÓDEM

BACHELOR'S THESIS
BAKALÁŘSKÁ PRÁCE

AUTHOR **ADRIÁN TÓTH**
AUTOR PRÁCE

SUPERVISOR **Ing. LENKA TUROŇOVÁ,**
VEDOUCÍ PRÁCE

BRNO 2017

Abstract

Do tohoto odstavce bude zapsán výtah (abstrakt) práce v anglickém jazyce.

Abstrakt

Do tohoto odstavce bude zapsán výtah (abstrakt) práce v českém (slovenském) jazyce.

Keywords

Sem budou zapsána jednotlivá klíčová slova v anglickém jazyce, oddělená čárkami.

Klíčová slova

Sem budou zapsána jednotlivá klíčová slova v českém (slovenském) jazyce, oddělená čárkami.

Reference

TÓTH, Adrián. *Continuous Integration and Automated Code Review in Open Source Projects*. Brno, 2017. Bachelor's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor Ing. Lenka Turoňová,

Continuous Integration and Automated Code Review in Open Source Projects

Declaration

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana X... Další informace mi poskytli... Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Adrián Tóth
November 3, 2017

Acknowledgements

V této sekci je možno uvést poděkování vedoucímu práce a těm, kteří poskytli odbornou pomoc (externí zadavatel, konzultant, apod.).

Contents

Chapter 1

Introduction

In these days, Continuous Integration (CI) is more often used in larger projects, where multiple developers are working on one and the same software product. This process ensures fast software development, called eXtreme Programming (XP), known as agile software development methodology. The methodology is mainly used to accelerate the development, nevertheless, development of software may be disrupted in various other ways. Nowadays, although this type of software development has many disadvantages, it is still much more often used on larger projects. The progress of the development may not be reached with a continuous integration which guarantees less disorder and failures. You may also know that the continuous integration is a part of the following open-source projects e.g. Facebook, Twitter, Mozilla. These projects use one of many famous continuous integration service Travis CI. Excluding Travis CI, there are plenty of other continuous integration services you may heard about, such as Jenkins, TeamCity, CircleCI, GitLab CI, Codeship and so on.

The software development process requires many code checking tools after every single code change in the source code. For as much as with every single change of code, there is a possibility to add, fix, derange or deteriorate any parts of the software product. These tools provide an automated code review and they afford a quick feedback by which they try to prevent these code impairments. Feedback about his adjustment is sent to the developer, who has made the change in the code. The automated process which provides the code review does not bother with executing a huge amount of tests. Above mentioned process is conducted via continuous integration server, which compiles the code, runs scripts and tests. The results are aggregated and the feedback is given to the developer who has made this code change. Continuous integration server is invoked every single time after any change is fetched in the source code and he had to execute the stated acts which are predefined. In next chapters we describe in details how does this workflow work and what steps are required to run.

The essence of this work is about the basics of continuous integration and its fundamentals. This thesis attempts to explain how the fundamentals of continuous integration and automated code review work. It describes how it is integrated to the software development, and how it works on an extensive project nowadays. Examples will be based on open-source project e.g. ManageIQ, which is a cloud manager founded by RedHat. The development process of the ManageIQ rests in agility and stability of the progress. These main factors of the development process could not be reached without a quick feedback to the developers working on project about their changes, that are submitted to the software product.

Chapter 2

Continuous Integration & Automated Code Review

Despite the fact that continuous integration and automated code review are used in a lot of projects, it is still an unknown part of software development. There are only a few researches describing this part of development how is it deployed, managed and used. Development analysts are not giving an adequate attention to this part of software development. They are usually describing it as a common part of development in a software development process. This part is concerning to extreme programming due to fast code change deployment. This development technique is very adaptive and still more and more open source projects are using it. There are many of developers relying to this type of software development which helps them rapidly. This chapter will give you a detailed point of view above the modern in-use software development methodology which is still in evolution.

2.1 Continuous Integration

Continuous integration has a key role in the software development process consisting of a few certain unavoidable steps which will be described later in next subsections. Everything begins at the moment, when one developer who has made changes in a source code of the software product is trying to commit them into the software product. The process of continuous integration has begun at this point and lasts until feedback is sent back to the developer. These stages of continuous integrations are proceed every time after the CI server has detected a change in a version control repository. This automation has a lot of benefits which are necessary to keep the software product without any kind of defects. Many of them are detected in time and reported back to the developer as a corrupted source code. Not a few developers may think that the continuous integration is only about compiling a source code and launching tests. In the next subsections, we will present the steps of the continuous integration and describe these individual phases in detail.

[IMAGE - CI book page 5]

2.1.1 Demands of continuous integration

The minimal requirements for good software development of a project where multiple developers are working on a same project are a version control repository and a continuous

integration server. The version control system guarantees a software configuration management which is required for the continuous integration. The continuous integration could not be deployed without a version control system. The meaning of the version control system is very important. You cannot manage changes that developers had made in the source code without a version control system. The version control system has a very positive impact on the developing project. The system offers a history of changes which may be highly useful if a rollback is desired. Besides the history of changes, this system may save more other information about the source code, e.g. who did the change, when was the change created, etc. In addition, the version control system is represents a primary source for the project source codes. This type of project setup is much more often used these days rather than in the past. Nearly every project has its own version control system which is provided by a repository hosting service.

A continuous integration server has a huge advantage. This is a reason why it is highly recommended. It depends on the developer, how does he deploy the continuous integration server. With the continuous integration server, he does not have to bother with such many scripts for the automation. Nevertheless, as he decides how the continuous integration server will be established, the system must contain these features. To facilitate the process of continuous integration, the system must support services as polling version control system, retention of build history, launching predefined steps such as scripts and tests. Furthermore, the system should offer an opportunity to send a feedback back to the developers. This server executes a series of actions or steps taken in order to achieve a particular end of continuous integration. The next subsection will determine and state these fundamental steps of the continuous integration scenario, describe and illustrate them in detail.

2.1.2 Stages of continuous integration

The stages of continuous integration insure code inspection and code integration. Before we begin, we need to clarify certain concepts which will be used later. To understand these steps, we need to understand what is the difference between **a build**, **a private build** and **an integration build**.

Definition 1 *A build may refer to a set of activities performed to generate, test, inspect, and deploy software.[1]*

Definition 2 *A private build define a process in which a software developer runs the build on his local machine to ensure that the changes he made work, before he commits them into a version control repository.*

Definition 3 *An integration build is the act of combining software components (programs and files) into a software system.[1]*

1. The change.
   =[REWRITE]The first step starts at the moment of committing the change by developer to a version control repository.
2. The reaction.
3. The feedback.
4. The waiting.

2.2 Automated Code Review

Abcd.

Chapter 3

Continuous Integration in open source projects

Asdf.

3.1 Open source

Asdf.

3.2 Github

Asdf.

3.3 TravisCI

Asdf.

Chapter 4

Conclusion

Asdf.

Bibliography

[1] Paul M. Duvall, A. G., Steve Matyas: *Continuous Integration - Improving Software Quality and Reducing Risk*. Pearson Education, Inc.. 2007. ISBN 0-321-33638-0.

Appendix A

Jak pracovat s touto šablonou

V této kapitole je uveden popis jednotlivých částí šablony, po kterém následuje stručný návod, jak s touto šablonou pracovat.

Jedná se o přechodnou verzi šablony. Nová verze bude zveřejněna do konce roku 2017 a bude navíc obsahovat nové pokyny ke správnému využití šablony, závazné pokyny k vypracování bakalářských a diplomových prací (rekapitulace pokynů, které jsou dostupné na webu) a nezávazná doporučení od vybraných vedoucích, která již teď najdete na webu (viz odkazy v souboru s literaturou). Jediné soubory, které se v nové verzi změní, budou `projekt-01-kapitoly-chapters.tex` a `projekt-30-prilohy-appendices.tex`, jejichž obsah každý student vymaže a nahradí vlastním. Šablonu lze tedy bez problémů využít i v současné verzi.

Popis částí šablony

Po rozbalení šablony naleznete následující soubory a adresáře:

bib-styles Styly literatury (viz níže).

obrazky-figures Adresář pro Vaše obrázky. Nyní obsahuje placeholder.pdf (tzv. TODO obrázek, který lze použít jako pomůcku při tvorbě technické zprávy), který se s prací neodevzdává. Název adresáře je vhodné zkrátit, aby byl jen ve zvoleném jazyce.

template-fig Obrázky šablony (znak VUT).

fitthesis.cls Šablona (definice vzhledu).

Makefile Makefile pro překlad, počítání normostran, sbalení apod. (viz níže).

projekt-01-kapitoly-chapters.tex Soubor pro Váš text (obsah nahraďte).

projekt-20-literatura-bibliography.bib Seznam literatury (viz níže).

projekt-30-prilohy-appendices.tex Soubor pro přílohy (obsah nahraďte).

projekt.tex Hlavní soubor práce – definice formálních částí.

Výchozí styl literatury (czechiso) je od Ing. Martínka, přičemž anglická verze (englishiso) je jeho překladem s drobnými modifikacemi. Oproti normě jsou v něm určité

odlišnosti, ale na FIT je dlouhodobě akceptován. Alternativně můžete využít styl od Ing. Radima Loskota nebo od Ing. Radka Pyšného[1]. Alternativní styly obsahují určitá vylepšení, ale zatím nebyly řádně otestovány větším množstvím uživatelů. Lze je považovat za beta verze pro zájemce, kteří svoji práci chtějí mít dokonalou do detailů a neváhají si nastudovat detaily správného formátování citací, aby si mohli ověřit, že je vysázený výsledek v pořádku.

Makefile kromě překladu do PDF nabízí i další funkce:

- přejmenování souborů (viz níže),
- počítání normostran,
- spuštění vlny pro doplnění nezlomitelných mezer,
- sbalení výsledku pro odeslání vedoucímu ke kontrole (zkontrolujte, zda sbalí všechny Vámi přidané soubory, a případně doplňte).

Nezapomeňte, že vlna neřeší všechny nezlomitelné mezery. Vždy je třeba manuální kontrola, zda na konci řádku nezůstalo něco nevhodného – viz Internetová jazyková příručka[2].

Pozor na číslování stránek! Pokud má obsah 2 strany a na 2. jsou jen "Přílohy" a "Seznam příloh" (ale žádná příloha tam není), z nějakého důvodu se posune číslování stránek o 1 (obsah "nesedí"). Stejný efekt má, když je na 2. či 3. stránce obsahu jen "Literatura" a je možné, že tohoto problému lze dosáhnout i jinak. Řešení je několik (od úpravy obsahu, přes nastavení počítadla až po sofistikovanější metody). **Před odevzdáním proto vždy překontrolujte číslování stran!**

Doporučený postup práce se šablonou

1. **Zkontrolujte, zda máte aktuální verzi šablony.** Máte-li šablonu z předchozího roku, na stránkách fakulty již může být novější verze šablony s aktualizovanými informacemi, opravenými chybami apod.

2. **Zvolte si jazyk**, ve kterém budete psát svoji technickou zprávu (česky, slovensky nebo anglicky) a svoji volbu konzultujte s vedoucím práce (nebyla-li dohodnuta předem). Pokud Vámi zvoleným jazykem technické zprávy není čeština, nastavte příslušný parametr šablony v souboru projekt.tex (např.: `documentclass[english]{fitthesis}` a přeložte prohlášení a poděkování do angličtiny či slovenštiny.

3. **Přejmenujte soubory.** Po rozbalení je v šabloně soubor `projekt.tex`. Pokud jej přeložíte, vznikne PDF s technickou zprávou pojmenované `projekt.pdf`. Když vedoucímu více studentů pošle `projekt.pdf` ke kontrole, musí je pracně přejmenovávat. Proto je vždy vhodné tento soubor přejmenovat tak, aby obsahoval Váš login a (případně zkrácené) téma práce. Vyhněte se však použití mezer, diakritiky a speciálních znaků. Vhodný název může být např.: "`xlogin00-Cisteni-a-extrakce-textu.tex`". K přejmenování můžete využít i přiložený Makefile:

   ```
   make rename NAME=xlogin00-Cisteni-a-extrakce-textu
   ```

[1] BP Ing. Radka Pyšného http://www.fit.vutbr.cz/study/DP/BP.php?id=7848
[2] Internetová jazyková příručka http://prirucka.ujc.cas.cz/?id=880

4. Vyplňte požadované položky v souboru, který byl původně pojmenován `projekt.tex`, tedy typ, rok (odevzdání), název práce, svoje jméno, ústav (dle zadání), tituly a jméno vedoucího, abstrakt, klíčová slova a další formální náležitosti.

5. Rozšířený abstrakt v češtině lze v šabloně povolit odkomentováním příslušných 2 bloků v souboru `fitthesis.cls`.

6. Nahraďte obsah souborů s kapitolami práce, literaturou a přílohami obsahem svojí technické zprávy. Jednotlivé přílohy či kapitoly práce může být výhodné uložit do samostatných souborů – rozhodnete-li se pro toto řešení, je doporučeno zachovat konvenci pro názvy souborů, přičemž za číslem bude následovat název kapitoly.

7. Nepotřebujete-li přílohy, zakomentujte příslušnou část v `projekt.tex` a příslušný soubor vyprázdněte či smažte. Nesnažte se prosím vymyslet nějakou neúčelnou přílohu jen proto, aby daný soubor bylo čím naplnit. Vhodnou přílohou může být obsah přiloženého paměťového média.

8. Nascanované zadání uložte do souboru `zadani.pdf` a povolte jeho vložení do práce parametrem šablony v projekt.tex (`documentclass[zadani]{fitthesis}`).

9. Nechcete-li odkazy tisknout barevně (tedy červený obsah – bez konzultace s vedoucím nedoporučuji), budete pro tisk vytvářet druhé PDF s tím, že nastavíte parametr šablony pro tisk: (`documentclass[zadani,print]{fitthesis}`). Barevné logo se nesmí tisknout černobíle!

10. Vzor desek, do kterých bude práce vyvázána, si vygenerujte v informačním systému fakulty u zadání. Pro disertační práci lze zapnout parametrem v šabloně (více naleznete v souboru fitthesis.cls).

11. Nezapomeňte, že zdrojové soubory i (obě verze) PDF musíte odevzdat na CD či jiném médiu přiloženém k technické zprávě.

Obsah práce se generuje standardním příkazem `\tableofcontents` (zahrnut v šabloně). Přílohy jsou v něm uvedeny úmyslně.

Pokyny pro oboustranný tisk

- **Oboustranný tisk je doporučeno konzultovat s vedoucím práce.**
- Je-li práce tištěna oboustranně a její tloušťka je menší než tloušťka desek, nevypadá to dobře.
- Zapíná se parametrem šablony: `\documentclass[twoside]{fitthesis}`
- Po vytištění oboustranného listu zkontrolujte, zda je při prosvícení sazební obrazec na obou stranách na stejné pozici. Méně kvalitní tiskárny s duplexní jednotkou mají často posun o 1–3 mm. Toto může být u některých tiskáren řešitelné tak, že vytisknete nejprve liché stránky, pak je dáte do stejného zásobníku a vytisknete sudé.
- Za titulním listem, obsahem, literaturou, úvodním listem příloh, seznamem příloh a případnými dalšími seznamy je třeba nechat volnou stránku, aby následující část začínala na liché stránce (\cleardoublepage).
- Konečný výsledek je nutné pečlivě překontrolovat.

Styl odstavců

Odstavce se zarovnávají do bloku a pro jejich formátování existuje více metod. U papírové literatury je častá metoda s použitím odstavcové zarážky, kdy se u jednotlivých odstavců textu odsazuje první řádek odstavce asi o jeden až dva čtverčíky (vždy o stejnou, předem zvolenou hodnotu), tedy přibližně o dvě šířky velkého písmene M základního textu. Poslední řádek předchozího odstavce a první řádek následujícího odstavce se v takovém případě neoddělují svislou mezerou. Proklad mezi těmito řádky je stejný jako proklad mezi řádky uvnitř odstavce. [**?**] Další metodou je odsazení odstavců, které je časté u elektronické sazby textů. První řádek odstavce se při této metodě neodsazuje a mezi odstavce se vkládá vertikální mezera o velikosti 1/2 řádku. Obě metody lze v kvalifikační práci použít, nicméně často je vhodnější druhá z uvedených metod. Metody není vhodné kombinovat.

Jeden z výše uvedených způsobů je v šabloně nastaven jako výchozí, druhý můžete zvolit parametrem šablony "`odsaz`".

Užitečné nástroje

Následující seznam není výčtem všech využitelných nástrojů. Máte-li vyzkoušený osvědčený nástroj, neváhejte jej využít. Pokud však nevíte, který nástroj si zvolit, můžete zvážit některý z následujících:

MikTeX LaTeX pro Windows – distribuce s jednoduchou instalací a vynikající automatizací stahování balíčků.

TeXstudio Přenositelné opensource GUI pro LaTeX. Ctrl+klik umožňuje přepínat mezi zdrojovým textem a PDF. Má integrovanou kontrolu pravopisu, zvýraznění syntaxe apod. Pro jeho využití je nejprve potřeba nainstalovat MikTeX.

WinEdt Ve Windows je dobrá kombinace WinEdt + MiKTeX. WinEdt je GUI pro Windows, pro jehož využití je nejprve potřeba nainstalovat MikTeX či TeX Live.

Kile Editor pro desktopové prostředí KDE (Linux). Umožňuje živé zobrazení náhledu. Pro jeho využití je potřeba mít nainstalovaný TeX Live a Okular.

JabRef Pěkný a jednoduchý program v Javě pro správu souborů s bibliografií (literaturou). Není potřeba se nic učit – poskytuje jednoduché okno a formulář pro editaci položek.

InkScape Přenositelný opensource editor vektorové grafiky (SVG i PDF). Vynikající nástroj pro tvorbu obrázků do odborného textu. Jeho ovládnutí je obtížnější, ale výsledky stojí za to.

GIT Vynikající pro týmovou spolupráci na projektech, ale může výrazně pomoci i jednomu autorovi. Umožňuje jednoduché verzování, zálohování a přenášení mezi více počítači.

Overleaf Online nástroj pro LaTeX. Přímo zobrazuje náhled a umožňuje jednoduchou spolupráci (vedoucí může průběžně sledovat psaní práce), vyhledávání ve zdrojovém textu kliknutím do PDF, kontrolu pravopisu apod. Zdarma jej však lze využít pouze s určitými omezeními (někomu stačí na disertaci, jiný na ně může narazit i při psaní bakalářské práce) a pro dlouhé texty je pomalejší.

Pozn.: Overleaf nepoužívá Makefile v šabloně – aby překlad fungoval, je nutné kliknout pravým tlačítkem na `projekt.tex` a zvolit "Set as Main File".

Užitečné balíčky pro LaTeX

Studenti při sazbě textu často řeší stejné problémy. Některé z nich lze vyřešit následujícími balíčky pro LaTeX:

- `amsmath` – rozšířené možnosti sazby rovnic,
- `float, afterpage, placeins` – úprava umístění obrázků,
- `fancyvrb, alltt` – úpravy vlastností prostředí Verbatim,
- `makecell` – rozšíření možností tabulek,
- `pdflscape, rotating` – natočení stránky o 90 stupňů (pro obrázek či tabulku),
- `hyphenat` – úpravy dělení slov,
- `picture, epic, eepic` – přímé kreslení obrázků.

Některé balíčky jsou využity přímo v šabloně (v dolní části souboru fitthesis.cls). Nahlédnutí do jejich dokumentace může být rovněž užitečné.

Sloupec tabulky zarovnaný vlevo s pevnou šířkou je v šabloně definovaný "L" (používá se jako "p").