FRI-ONLINE-1-CCT1-02

COMPARATIVE ANALYSIS OF TEST-DRIVEN DEVELOPMENT AND ACCEPTANCE TEST-DRIVEN DEVELOPMENT[2]

Assist. Prof. Lachezar Yordanov, PhD
Department of Computer Systems and Technologies,
University of Ruse
Phone: +359 82 888 859
E-mail: Liordanov@ecs.uni-ruse.bg

M. Eng. Teodora Yordanova
Junior Software Developer
Parallel Bulgaria Ltd, Ruse
Phone: +359 882 820 695
E-mail: tyordanova@parallel-bg.com

Abstract: *The report represents a comparative analysis of two techniques for software development and the implementation of (automatic) tests – test-driven development and acceptance test-driven development. The first technique aims to check whether the code is valid (emphasis on functional tests), and the second one - whether the code is working as expected (non-functional tests). The report describes the main points in the development of the applications and the creation of the accompanying tests; the advantages and disadvantages of both techniques. Analyzed are the necessary resources (development time, number and type of specialists, knowledge in the field of the software product) and the saved losses from detected in time errors (in the presentation of information, in the main modules of the application, in the product specification, in the client's assignment). The formed conclusions provide guidelines in which situations test-driven development and acceptance test-driven development can be used and which of the two techniques is more suitable.*

Keywords: *Test-Driven Development, Acceptance Test-Driven Development, Automatic Testing, Product Specification*

JEL Codes: *I20, C88,*

ВЪВЕДЕНИЕ

Докладът представлява сравнителен анализ на две техники за разработване на софтуер и реализирането на (автоматични) тестове – разработване водено от тестове и разработване водено от тестове за възприемане. Първата техника има за цел да провери дали програмата работи правилно (набляга на функционалните тестове), а втората – дали работи според очакванията (нефункционални тестове) (Jash Unadkat, J., 2019). В доклада са описани основните стъпки, предимства и недостатъци на всяка техника. Предоставени са насоки кога и къде могат да бъдат използвани.

ИЗЛОЖЕНИЕ

При създаването на софтуер и последвалото тестване могат да се разграничат две основни техники – тестване след реализация (Testing Post Creation, TPC) и разработване водено от тестове (Test-Driven Development, TDD). Първата техника е най-старата и при нея тестовете се създават след реализирането на дадена програма или модул от нея. При втората създаването на тестовете изпреварва това на софтуера, като предварително подготвените тестове се използват по време на целия процес – при реализирането, междинното и окончателното тестване на продукта.

[2] Докладът е представен на сесия FRI-ONLINE-1-CCT1 на 13 ноември 2020 с оригинално заглавие на български език: СРАВНИТЕЛЕН АНАЛИЗ НА РАЗРАБОТВАНЕТО ВОДЕНО ОТ ТЕСТОВЕ И РАЗРАБОТВАНЕТО ВОДЕНО ОТ ТЕСТОВЕ ЗА ВЪЗПРИЕМАНЕ

Разработването водено от тестове намира приложение при изграждането на системи с множество функционалности и/или при липса на опит на екипа в сферата на работа на крайния продукт. Създаването на тестовете преди реализирането на софтуера позволява бързото откриване на грешки и тяхното отстраняване още в началото. Това от своя страна намалява времето за разработка и повишава качеството на продукта. За създаването на предварителните тестовете е необходимо осъществяването на анализ както на спецификацията на продукта, така и на средата и сферата на работа на крайния продукт. Това от своя страна налага включването на анализатори и разработчици на тестове (QA специалисти). Разработването водено от тестове насърчава съвместната работа между различните специалисти в екипа и обменянето на знания. Това от своя страна води до реализирането на структуриран, подреден и четим код.

В последните години започва да навлиза още една техника, която се явява подразделение на разработването водено от тестове – разработване водено от тестове за възприемане (Acceptance Test-Driven Development, ATDD). Основната разлика е, че ако при първоначалния подход се съсредоточаваме върху функционалните тестове на софтуера, то тук основен акцент са нефункционалните тестове.

Разработване водено от тестове

Класическото разработване водено от тестове има за цел да отговори на въпроса дали кодът е правилен (Is the code valid?). Казано по друг начин създава тестове от гледна точка на програмиста.

Основните стъпки при тази техника са:

- анализ на изискванията;
- създаване на спецификация;
- създаване на тестове или сценарии за тях;
- разработване на приложението и тестване чрез подготвените тестове;
- провеждане на допълнителни тестове;
- внедряване на продукта.

Този подход повишава производителността на програмистите, защото грешките се откриват в ранните етапи на разработка, което от своя страна води до по-малки промени по структурата и основните функции на програмата. Вследствие на това се улеснява създаването на софтуер с множество функционалности, с гъвкава и лесна за поддръжка структура. Освен това тази техника дава определена сигурност при голяма промяна на част от архитектурата на приложението (дори след внедряване на продукта).

Основен недостатък на разработването водено от тестове е непредвиждането на грешки в спецификацията на софтуера. Това налага включването на специалист, който има опит в работата с или изграждането на такъв тип системи.

Разработване водено от тестове за възприемане

Разработването водено от тестове за възприемане има за цел да отговори на въпроса дали кодът работи според очакванията (Is the code working as expected?), т.е. създава се тест за приемане от гледна точка на потребителя.

Основните стъпки са:

• изясняване на изискванията, създаване на тестове или сценарии за тях и създаване на спецификация;

- разработване на приложението и тестване чрез подготвените тестове;
- провеждане на допълнителни тестове;
- внедряване на продукта.

При тази техника изискванията са много ясно анализирани още в началото. Поради тази причина в ранните етапи се разчита до голяма степен на сътрудничеството между клиентите, QA специалистите и разработчиците. Съответно това води до повишаване на разходите (време, специалисти, пари) за първоначален анализ и значителното им намаляване за тестване и най-вече за същинска реализация. Много важно предимство на разработването водено от тестове за възприемане е това, че тестът за приемане служи като ръководство за целия процес на разработка.

Основни разлики между двете техники

Първата разлика между двете техники е моментът, в който се осъществява анализирането на изискванията, създаването на спецификацията и реализирането на основните тестове и сценарии за тях. При класическия вариант се осъществяват последователно (в реда на изброяването им). При разработване водено от тестове за възприемане тези процеси са паралелни и взаимно свързани, като често крайният вариант на спецификацията е готов след създаването на тестовете.

Друга важна разлика е необходимите познания за този тип системи. При разработването водено от тестове за възприемане екипът може да няма опит с такъв тип системи и/ или познания за сферата на работа на крайния продукт, защото при тази техника се разчита на съвместната работа с клиента. При другата техника е задължително включването на специалист с опит в дадената сфера.

Последната, но не и по важност, разлика е видът на предварително създадените тестове. При TDD се набляга на функционалните тестове, а при ATTD – на нефункционалните. Съответно другият тип тестове се създават и провеждат след реализирането на системата.

ЗАКЛЮЧЕНИЕ

Класическото разработване водено от тестове е подходящо, когато:

- голяма част от функционалностите се основава на закон, правило или формула;
- може да се използва съществуващ софтуер като образец;
- екипът или част от него имат опит с такива системи или в сферата на работа на крайния продукт.

Съответно разработването водено от тестове за възприемане е подходящо да се използва, когато:

- приложението ще замени две или повече системи, които се разминават в използваната терминология или в процеса на работа;
- липсват примери за такъв тип системи;
- екипът няма опит в сферата на работа на крайния продукт.

От направения анализ се вижда, че скоро двете техники – класическото разработване водено от тестове (TDD) и разработването водено от тестове за възприемане (ATTD), ще се изравнят като употреба. Основна причина за това е внедряването на все повече компютризирани системи в процеса на работа на различни фирми и организации и трупането на опит на екипите разработващи съответните продукти. Вероятно в бъдеще често ще се използва хибриден вариант, като за изграждане на цялостната система ще се стъпва на ATTD, а за модулите с бизнес логиката – TTD.

ACKNOWLEDGEMENT

This paper is supported by project 20-FEEA-01 “Methods and tools for multimedia content analysis and automated document and big data processing”, funded by the Research Fund of the “Angel Kanchev” University of Ruse

REFERENCES

Jash Unadkat, J. (2019). BDD vs TDD vs ATDD : Key Differences, Technical Content Writer at BrowserStack - October 4, 2019, www.browserstack.com/guide/tdd-vs-bdd-vs-atdd.