

Vysoká škola ekonomická v Praze

Fakulta informatiky a statistiky



Využití TMMi ke zlepšování procesů a praktik testování v agilních projektech

DIPLOMOVÁ PRÁCE

Studijní program: Aplikovaná informatika

Studijní obor: Informační systémy a technologie

Autor: Bc. Dominik Veselý

Vedoucí diplomové práce: Ing. et Ing. Michal Doležel, Ph.D.

Praha, duben 2019

Prohlášení

Prohlašuji, že jsem diplomovou práci s názvem Využití TMMi ke zlepšování procesů a praktik testování v agilních projektech vypracoval samostatně za použití v práci uvedených pramenů a literatury.

V Praze dne 28. dubna 2019

.....

Bc. Dominik Veselý

Poděkování

Děkuji tímto Ing. et Ing. Michalu Doleželovi, Ph.D. za vstřícnost a poskytnutí cenných rad při vedení této práce. Děkuji také kolegům z vývojového týmu za čas věnovaný posuzování výstupů práce. Na závěr děkuji Bc. Michaele Grussově a Martině Černé, DiS za pomoc s korekcí textu.

ABSTRAKT

Tato diplomová práce se zabývá procesy testování softwaru a možnostmi jejich zlepšování. Pozornost je zejména věnována zlepšování těchto procesů prostřednictvím modelů k tomu určených. Práce se zaměřuje na model TMMi (Test Maturity Model Integration), který představuje robustní rámec pro komplexní zlepšování a zvyšování zralosti procesů testování. TMMi byl prvotně určen pro organizace, které se řídí tradičními metodami vývoje. V současné době je však velkým trendem přecházet na agilní metody, na což TMMi Foundation reaguje a snaží se organizacím se svým modelem vyjít vstříc.

Hlavním cílem praktické části je neformální hodnocení současného stavu procesů testování ve společnosti Glogster, a.s. s využitím modelu TMMi. Zjištěné nedostatky jsou předmětem návrhu řešení pro zlepšení procesů testování tak, aby bylo možné dosáhnout druhé úrovně modelu TMMi. K analýze je třeba vybrat vhodnou metodu hodnocení. Tuto metodu ovšem TMMi veřejně nenabízí. Je tedy třeba vybrat vhodnou neoficiální metodu, jejíž tvorba byla např. předmětem jiných diplomových prací. Vzniklý návrh je diskutován se členy vývojového týmu. Přínosem práce je tedy identifikace nedostatků procesů testování softwaru v konkrétní společnosti a na míru vytvořený návrh, který daná společnost může využít pro své účely vylepšování těchto procesů. Dalším přínosem práce je objasnění použitelnosti modelu TMMi v malých a středně velkých agilních týmech.

KLÍČOVÁ SLOVA

Kvalita softwaru, TMMi, Agilní testování, Zlepšování kvality procesů, Zlepšování procesů testování

JEL KLASIFIKACE

M15, L86

ABSTRACT

This thesis deals with the processes of software testing and possibilities of their improvement. Particular attention is paid to improve testing processes through dedicated models. The thesis focuses on the TMMi (Test Maturity Model Integration), which provides a robust framework for the comprehensive improvement of test processes maturity. TMMi was primarily designed for organizations that follow traditional development methods. However, it is a major trend nowadays to switch to agile method, whereupon TMMi Foundation responds and tries to accommodate organizations with their model.

The main goal of the practical part is an informal assessment of the current state of testing processes in the company Glogster, a.s. using the model TMMi. Identified deficiencies are the subject of a proposal to improve testing processes so that the second level of the TMMi can be achieved. An appropriate method of evaluation should be selected for analysis. However, TMMi model does not contain a practical assessment method. It is therefore necessary to select an appropriate unofficial method, whose work was, for example, the subject of other dissertations. Proposed results are discussed with members of the development team. The contribution of this work is to identify deficiencies in software testing processes in a particular company and tailor-made design that a company can use to improve these processes. Another contribution of the work is to clarify the usability of the TMMi model in small and medium-sized agile teams.

KEYWORDS

Software quality, TMMi, Agile testing, Process quality improvement, Test process improvement

JEL CLASSIFICATION

M15, L86

Obsah

Úvod	11
Cíle práce	11
Metody dosažení cíle.....	12
Struktura práce	14
Očekávané přínosy.....	14
Rešerše.....	15
Kvalifikační práce	16
Odborné publikace.....	17
Internetové zdroje.....	19
1. Řízení kvality softwaru	20
1.1. Kvalita softwaru	20
1.2. Řízení/kontrola kvality softwaru	22
1.3. Testování softwaru.....	23
1.3.1. Techniky testování a členění testů	23
1.3.2. Agilní testování a role testera	27
1.3.3. Procesy testování.....	28
2. Zlepšování kvality softwaru.....	30
2.1. CMMI (Capability Maturity Model Integration)	31
3. Zlepšování kvality procesů testování.....	33
3.1. Test Process Improvement (TPI).....	35
3.2. TestSPICE	36
4. Test Maturity Model Integration.....	39
4.1. Základní informace.....	39
4.2. Úrovně zralosti dle TMMi.....	39
4.2.1. První úroveň – Initial („Počáteční“).....	41
4.2.2. Druhá úroveň – Managed („Řízená“).....	41
4.2.3. Třetí úroveň – Defined („Definovaná“).....	42
4.2.4. Čtvrtá úroveň – Measured („Měřená“).....	43

4.2.5.	Pátá úroveň – Optimization („Optimalizovaná“)	43
4.3.	Komponenty TMMi	44
4.4.	Hodnocení pro získání certifikace TMMi	46
4.5.	Využití TMMi v agilním vývoji	47
4.6.	Shrnutí modelu	50
5.	Hodnocená společnost	51
5.1.	Představení hodnocené společnosti	51
5.2.	Organizační struktura společnosti	52
5.3.	Současná metodika vývoje a testování	53
5.4.	Zvolená metoda pro hodnocení společnosti dle TMMi	55
5.4.1.	Popis zvolené hodnotící metody	56
6.	Hodnocení současného stavu procesů testování	60
6.1.	Plán hodnocení	60
6.2.	Porovnání současného stavu proti druhé úrovni TMMi	61
6.2.1.	PA: Test Policy and Strategy	61
6.2.2.	PA: Test Planning	66
6.2.3.	PA: Test Monitoring and Control	72
6.2.4.	PA: Test Design and Execution	78
6.2.5.	PA: Test Environment	84
6.3.	Vyhodnocení dosažené úrovně	87
7.	Návrh nového řešení	89
7.1.	Návrh na zlepšení PA: Test Policy and Strategy	89
7.2.	Návrh na zlepšení PA: Test Planning	91
7.3.	Návrh na zlepšení PA: Test Environment	92
7.4.	Posouzení návrhu	93
7.4.1.	Odpovědi členů vývojového týmu společnosti Glogster	94
7.5.	Vyhodnocení odpovědí	97
Závěr		99
Seznam literatury		101

Přílohy	105
Příloha A: Kanban tabule využívaná společností Glogster	105
Příloha B: Detail karty položky v produktovém backlogu	106

Seznam obrázků

Obrázek 1 - Vývoj modelů TMA / TPI a jejich vztahů [13]	34
Obrázek 2 - Popis jednotlivých úrovní zralosti klíčových oblastí [36]	35
Obrázek 3 - Matice zralosti TPI [7]	36
Obrázek 4 - Úrovně zralosti a procesní oblasti modelu TMMi[3].....	40
Obrázek 5 - Struktura a komponenty modelu TMMI, Upravená verze z [3]	45
Obrázek 6 - Šablona hodnocení [7]	57
Obrázek 7 - Příklad dokumentu Politika testování [53].....	90
Obrázek 8 - Příklad posouzení rizik [6].....	92

Seznam tabulek

Tabulka 1 - výskyt klíčových slov	16
Tabulka 2 - Stupnice hodnocení atributů procesů (SPICE) [35]	37
Tabulka 3 - Aplikovatelné a neaplikovatelné komponenty TMMi v agilním vývoji [5]	49
Tabulka 4 - Stupnice hodnocení TMMi [7]	58
Tabulka 5 – Šablona tabulky hodnocení [7].....	58
Tabulka 6 - Hodnocení – PA2.1.SG1 - Establish a Test Policy	62
Tabulka 7 - Hodnocení – PA2.1.SG2 - Establish a Test Strategy	63
Tabulka 8 - Hodnocení – PA2.1.SG3 - Establish Test Performance Indicators.....	66
Tabulka 9 - Hodnocení – PA2.2.SG1 - Perform Risk Assessment	67
Tabulka 10 - Hodnocení – PA2.2.SG2 - Establish a Test Approach	68
Tabulka 11 - Hodnocení – PA2.2.SG3 - Establish Test Estimates.....	70
Tabulka 12 - Hodnocení – PA2.2.SG4 - Develop a Test Plan.....	71
Tabulka 13 - Hodnocení – PA2.2.SG5 - Obtain commitment to the Test Plan.....	72
Tabulka 14 - Hodnocení – PA2.3.SG1 - Monitor Test Progress Against Plan.....	74
Tabulka 15 - Hodnocení – PA2.3.SG2 - Monitor Product Quality Against Plan and Expectations	76
Tabulka 16 - Hodnocení – PA2.3.SG3 - Manage Corrective Actions to Closure.....	77
Tabulka 17 - Hodnocení – PA2.4.SG1 - Perform Test Analysis and Design using Test Design Techniques.....	80
Tabulka 18 - Hodnocení – PA2.4.SG2 - Perform Test Implementation	80
Tabulka 19 - Hodnocení – PA2.4.SG3 - Perform Test Execution	82
Tabulka 20 - Hodnocení – PA2.4.SG4 - Manage Test Incidents to Closure.....	83
Tabulka 21 - Hodnocení – PA2.5.SG1 - Develop Test Environment Requirement.....	85
Tabulka 22 - Hodnocení – PA2.5.SG2 - Perform Test Environment Implementation.....	86
Tabulka 23 - Hodnocení – PA2.5.SG3 - Manage and Control Test Environment	87
Tabulka 24 - Výsledek hodnocení - 2. úroveň TMMi	88
Tabulka 25 - Kanban společnosti Glogster.....	105

Úvod

V poslední době softwarový průmysl vynakládá značné úsilí na zlepšování kvality svých produktů. Komplikovanost tohoto úsilí se odvíjí od vytváření stále složitějšího a rozsáhlejšího softwaru, přičemž požadavky zákazníků jsou čím dál více náročné. Navzdory povzbudivým výsledkům s různými přístupy ke zlepšení kvality je softwarový průmysl stále daleko od bezchybnosti. Pro zlepšení kvality produktů se softwarový průmysl často zaměřuje na zlepšování svých vývojových procesů. Na základě teoretických poznatků a zkušeností z praxe začalo vznikat hojné množství modelů, rámců a metodik. Ty mají za úkol předat soubor osvědčených postupů (best practices) a dopomoci k systematickému zavádění a zlepšování procesů v organizaci.

Jedním z modelů, který se hojně využívá ke zdokonalení vývojových procesů, je model CMMI [1]. Model zralosti (CMM) a jeho nástupce Capability Maturity Model Integration (CMMI) jsou často považovány za průmyslový standard pro zlepšování softwarových procesů [2]. Navzdory skutečnosti, že testování často představuje minimálně 30-40 % celkových nákladů projektu, se modely zlepšování softwarových procesů, jako je CMM a CMMI, věnují testování jen omezeně. Jako odpověď testovací komunita vytvořila a stále vyvíjí vlastní modely pro zlepšování kvality a zvyšování zralosti procesů testování. Tato práce se věnuje využití modelu testovací zralosti TMMi (Test Maturity Model Integration). TMMi je rozsáhlý model pro zvyšování zralosti testovacích procesů a může sloužit jako doplněk k CMMI, nebo jako samostatný model. [3]

Současně s vysokými nároky na kvalitu produktů se klade důraz i na rychlost jeho vývoje a schopnost průběžně reagovat na změny požadavků. Tyto nároky jsou hlavní doménou agilních metodik vývoje. Tyto metodiky nabírají na oblibě především díky své flexibilitě a možnostem přizpůsobování se potřebám týmu. Agilní metodiky jsou založeny na iterativním přírůstkovém vývoji. Testovací procesy jsou vykonávány průběžně napříč všemi jeho etapami. Proto je testování v agilním vývoji velmi důležitým prvkem a jeho přirozenou součástí. Z toho důvodu je hlavním smyslem této práce uplatnění modelu TMMi pro zvyšování zralosti procesů testování v agilním prostředí.

Cíle práce

Hlavní cíl diplomové práce představuje ohodnocení současných procesů testování společnosti Glogster a na jeho základě vytvoření návrhu na zlepšení procesů testování tak,

aby bylo možné dosáhnout druhé úrovně („definovaná“) modelu TMMi s ohledem na praktikované metody této společnosti.

Pro splnění hlavního cíle byly definovány následující dílčí cíle:

- DC1 – Ucelit teoretické poznatky o řízení kvality, testování softwaru a zlepšování kvality softwaru. Dále objasnit problematiku zlepšování procesů testování prostřednictvím modelů a rámců, které jsou k tomu využívány.
- DC2 – Představit model TMMi pro zvyšování zralosti testovacích procesů s možnostmi jeho uplatnění v agilním prostředí.
- DC3 – Zvolit, případně upravit hodnotící metodu TMMi pro následné provedení neformálního hodnocení společnosti.
- DC4 – Provést neformální hodnocení konkrétní společnosti proti druhé úrovni zralosti modelu TMMi.
- DC5 – Na základě výsledků hodnocení vytvořit návrh na zvýšení zralosti procesů testování, který umožní dosáhnout druhé úrovně modelu TMMi.
- DC6 – Provést interview se členy vývojového týmu s cílem evaluovat vypracovaný návrh, provedené hodnocení a vhodnost využití modelu TMMi ve společnosti.

Metody dosažení cíle

Pro splnění stanovených cílů této práce bylo využito metodiky Výzkum a návrh (angl. *DSR* – *Design Science Research*). Tato metodika se skládá z šesti fází [4]:

- Identifikace problému a motivace (angl. *Identify Problem & Motivate*)
- Definice cílů řešení (angl. *Define Objectives of a Solution*)
- Návrh a vývoj (angl. *Design & Development*)
- Uvedení (angl. *Demonstration*)
- Vyhodnocení (angl. *Evaluation*)
- Komunikace (angl. *Communication*)

Identifikace problému a motivace

Společnosti zaměřující se na vývoj softwaru jsou v současnosti pod velkým tlakem konkurence. Zároveň jsou na jejich softwarové produkty kladeny stále větší nároky především na jejich kvalitu a rychlost dodání. Aby byly společnosti schopné tyto požadavky naplnit, vzniká zde snaha o optimalizaci a zlepšování kvality zavedených podnikových procesů. Mezi ty spadají i testovací procesy.

Práce se zaměřuje na model TMMi, který slouží ke zlepšování a postupnému zvyšování zralosti testovacích procesů. TMMi Foundation, mezinárodní nezisková organizace se zaměřením na podporu organizací ve zlepšování kvality softwaru, v roce 2018 vydala dokument, který umožňuje aplikovat model v agilním prostředí. Dokument však poskytuje velkou benevolenci k plnění konkrétních cílů modelu a umožňuje společností nebo vývojovým týmům, aby si je mohly upravovat pro své potřeby. Je tedy třeba jasně navrhnout cíle procesních oblastí tak, aby byly v souladu jak s agilními praktikami, tak s modelem TMMi a zároveň byly přizpůsobeny potřebám konkrétní společnosti. Takto navržené cíle umožní hodnotit testovací procesy v agilním prostředí.

Definice cílů řešení

Cílem práce je ohodnotit stav testovacích procesů proti stanoveným cílům procesních oblastí modelu TMMi ve společnosti Glogster, a.s. Zmíněná společnost se skládá z malého vývojového týmu, který se řídí čistě agilními principy s použitím metodiky SCRUM, která zde však nefiguruje ve své základní podobě, ale je přizpůsobená potřebám týmu (více informací o společnosti viz Kapitola 5. Hodnocená společnost). Následně, na základě zjištěných nedostatků z hodnocení, je cílem vytvořit návrh na zlepšení těchto procesů, aby bylo možné dosáhnout druhé úrovně zralosti v souladu s tímto modelem.

Návrh a vývoj

Pro vypracování teoretické a praktické části je čerpáno z odborných publikací a internetových zdrojů, které jsou uvedeny v rešerši diplomové práce a v kapitole použitých zdrojů. Pro navržení cílů procesních oblastí je využito syntézy informací z dokumentu *TMMi in the Agile world* [5] a odborné publikace *Agile Testing: A Practical Guide for Testers and Agile Teams* [6]. Samotné hodnocení (viz kapitola 6) je založeno na autorových zkušenostech a poznatcích nabytých během působení v hodnocené společnosti. Vzniklý návrh se zakládá na informacích, které jsou podloženy vhodnými zdroji.

Uvedení

Výsledky práce byly předány k posouzení klíčovým členům vývojového týmu. Následně byly těmto členům kladeny otázky s cílem získat názory ohledně využitelnosti modelu TMMi ve společnosti. Získané informace slouží k závěrečnému vyhodnocení. Tato část se nachází v podkapitole 7.4 - Posouzení návrhu.

Vyhodnocení

Vyhodnocení výsledků práce je obsaženo v podkapitole 7.5 - Vyhodnocení. Tato část shrnuje informace získané z posudků klíčových členů a autorovy poznámky.

Komunikace

Tato část je naplněna zveřejněním diplomové práce.

Struktura práce

V kapitolách 1 až 3 se nacházejí teoretické poznatky o řízení kvality, testování softwaru a zlepšování kvality softwaru. Tyto poznatky jsou využity pro zpracování praktické části. Pozornost je zejména věnována zlepšování procesů testování prostřednictvím modelů a rámců k tomu určených (viz kapitola 3). Podstatou první části je seznámení se s modelem TMMi, z toho důvodu je model řešen v samostatné kapitole (viz kapitola 4).

V druhé části je představena hodnocená společnost. Jelikož TMMi Foundation veřejně nenabízí oficiální metodu hodnocení zralosti testovacích procesů, tak musí být vybrána vhodná neoficiální metoda, která vyhovuje potřebám hodnocené organizace (viz podkapitola 5.4). Vybraná metoda slouží k ohodnocení současného stavu testovacích procesů dané společnosti (viz kapitola 6). Na základě hodnocení jsou v této části zjištěné nedostatky předmětem návrhu řešení na zlepšení řízení kvality procesu testování tak, aby bylo možné dosáhnout druhé úrovně modelu TMMi (viz kapitola 7). Evaluační vypracovaného návrhu je ověřena prostřednictvím rozhovorů s členy vývojového týmu společnosti. Cílem rozhovorů je ověření uskutečnitelnosti, náročnosti, omezení, přínosů realizace zmiňovaného návrhu (viz podkapitola 7.4).

Očekávané přínosy

Vzniklý návrh na zlepšení současných procesů testování společnosti Glogster a samotné hodnocení společnosti s použitím modelu TMMi je především očekávaným přínosem pro hodnocenou společnost. Výsledky hodnocení mohou společnosti sloužit jako dokumentace zavedených testovacích procesů a praktik. Realizace vzniklého návrhu je předpokladem pro zvýšení zralosti testovacích procesů v hodnocené společnosti a dosažení druhé úrovně zralosti testovacích procesů podle modelu TMMi.

Výsledky práce mohou také sloužit malým a středně velkým agilním týmům, kteří mají zájem o využití modelu TMMi pro zlepšování kvality a zvyšování zralosti svých testovacích procesů. Zejména hodnotící tabulky s uvedeným plněním jednotlivých cílů může pro takovéto společnosti sloužit jako vzor pro jejich vlastní hodnocení.

Rešerše

Cílem této kapitoly je uvést a okomentovat hlavní informační zdroje, které se zabývají tématem na zlepšování procesů testování, jejich standardizaci s využitím modelu TMMi a jeho aplikace v agilním vývoji.

Pro účel vyhledávání adekvátních zdrojů byla určena klíčová slova a vyhledávací řetězce v anglickém a českém jazyce:

- TMMi
- Test Maturity Model Integration
- TMMi and agile
- TMMi a agilní přístup
- Test maturity

Tyto vyhledávací řetězce byly zadány do vyhledávačů známých informačních zdrojů, mezi které patří:

- Scholar.google.cz – vyhledávač odborné literatury od společnosti Google. Umožňuje vyhledávat odborné články, knihy, abstrakty od různých akademických vydavatelů, univerzit, nebo jiných webových stránek. Také umí vyhledávat související díla, citace apod.
- Vse.summon.serialssolutions.com – vyhledávač informačních zdrojů, který poskytuje VŠE (Vysoká škola ekonomická). Lze skrze něj vyhledat téměř všechny dostupné placené elektronické informační zdroje VŠE, katalog knihovny VŠE a také obsahuje databázi vysokoškolských kvalifikačních prací (bakalářské, diplomové a disertační práce). Mimo to jsou indexovány odborné volně dostupné (open access) databáze a vědecké tituly časopisů.
- Theses.cz - systém pro odhalování plagiátů mezi závěrečnými pracemi, který je vyvíjen a provozován Masarykovou univerzitou. Slouží vysokým školám a univerzitám (nejen v ČR) jako národní registr závěrečných prací.

V následné tabulce je uveden počet výskytů klíčových slov ve vyhledávacích informačních zdrojů, které jsou uvedeny výše. Klíčová slova uvedená v uvozovkách byla vyhledávána jako celistvý text. To znamená, že každý výsledek musí obsahovat kompletní řetězec slov uvedený v závorkách. U výrazů (především víceslovných) uvedených bez uvozovek nemusí být vyhledávání stoprocentně relevantní. Pro hledání zdrojů na téma ve spojení s agilním

přístupem je využit vyhledávací operátor „AND“, který zaručí vypsání všech zdrojů, ve kterých se nachází obě slova zároveň.

Informační zdroj / Klíčové slovo	Google Scholar	VŠE zdroje – Summon	Theses.cz
TMMi	2 940	8 619	36
“Test Maturity Model Integration“	271	26	19
TMMi AND agile	259	31	16
TMMi AND agilní	3	0	15
„Test maturity“	1020	202	20

Tabulka 1 - výskyt klíčových slov

Ze získaných výčtů zdrojů byly vybrány ty, jenž jsou pro tuto diplomovou práci klíčové. Tyto zdroje jsou spolu se stručným popisem uvedeny níže.

Kvalifikační práce

Diplomová práce *Modely zlepšování procesů testování softwaru a zajištění kvality* (2012) [7] je zaměřená zejména na model Test Maturity Model Integration (TMMi), který důkladně analyzuje, hledá možnosti a omezení jeho využití a shrnuje existující zkušenosti s modelem. Protože TMMi nedefinuje praktický postup (metodu) pro ohodnocení procesů dle modelu, v této práci je taková metoda navržena. Navržená metoda slouží jako předloha hodnotící metody v této práci.

Diplomová práce *Řízení kvality softwaru ve společnosti XY* (2015) [8] obsahuje procesní analýzu současného stavu procesu testování softwaru ve společnosti XY. Identifikované podprocesy jsou v práci posléze vymodelovány a zachyceny do tzv. Karet procesů. Zjištěné nedostatky jsou předmětem návrhu řešení na zlepšení řízení kvality procesu testování softwaru. Návrh řešení je poté implementován na současný stav procesu testování softwaru s cílem dosáhnout druhé úrovně zralosti procesů testování softwaru ohodnocenou oproti modelu TMMi.

Diplomová práce *Procesní přístup k testování software* (2017) [9] – v této práci je aplikován procesní přístup na procesy metodiky testování vybrané společnosti. Je zde provedena procesní analýza stávajícího stavu procesů této metodiky a zhodnocena jejich zralost oproti modelu TMMi. Procesní analýza a zhodnocení zralosti jsou následně podkladem pro hlavní cíl diplomové práce, kterým je optimalizace procesů s cílem dosáhnout druhé úrovně modelu TMMi.

Diplomová práce *Zvyšovanie vyspelosti procesov testovania v agilných projektoch – kvalitatívny empirický výskum* (2017) [10] analyzuje vývoj procesu testování z pohledu členů agilních týmů. V práci je popsáno agilní testování. Autor práce se dále zaměřuje na vyspělost agilního testování. Je zde objasněno, že zvyšování vyspělosti procesů testování v agilním vývoji nelze provádět porovnáváním s jinými týmy či striktním dodržováním standardizovaných postupů, ale je třeba aktivně hledat vhodné řešení, které je vyhovující pro jednotlivé členy konkrétního týmu. Práce také obsahuje rozhovory s jednotlivými členy několika agilních týmů ohledně vyspělosti testování. Z diskuzí autor vybral společné problémy v procesech testování a na jejich základě předkládá vytvořený návrh na zlepšení.

Zvyšování zralosti testovacích procesů v IT společnosti (2009) [11] je diplomová práce, která se zabývá zvyšováním zralosti testovacích procesů s využitím modelu CMMI, který se primárně zaměřuje na celou oblast podnikových procesů. CMMI pokrývá široké spektrum oblastí, které zahrnují i procesy testování. Autor práce se zaměřil pouze na oblasti a činnosti které se týkají testování. Na jejich základě provedl hodnocení současného stavu testovacích procesů konkrétní společnosti a vypracoval návrh na jejich zlepšení s možností dosáhnout až třetí úrovně modelu CMMI. Autor v závěru zmiňuje model TMMi, který se podrobně a výhradně zabývá oblastí testování a uvádí, že jeho využití by mohlo být vhodným tématem pro jiné diplomové práce.

Odborné publikace

Dokument *Test Maturity Model integration (TMMi)* (2012) [3] představuje hlavní dokument od TMMi Foundation, který obsahuje veškeré informace ohledně modelu TMMi. Také obsahuje referenční model s požadavky na plnění jednotlivých komponent modelu TMMi, který je bohužel svým charakterem vhodný pouze pro tradiční přístup vývoje.

Dokument *TMMi in the Agile world* (2018) [5] vytvořený přímo nadací TMMi popisuje, jak zkombinovat agilní přístup pro vývoj softwaru s modelem zlepšování procesů TMMi a jak může být model TMMi použit v agilním prostředí.

Kniha *Agile Testing: A Practical Guide for Testers and Agile Teams* (2009) [6] definuje agilní testování a ilustruje role testerů na příkladech konkrétních agilních týmů. Vysvětluje rozdíly mezi tradičním a agilním testováním. Objasňuje, jak používat agilní testovací kvadranty, které napomáhají k určení, jaké testy jsou zapotřebí, kdo je má provádět a jaké nástroje využít. Především obsahuje konkrétní praktiky testování, které jsou využity pro vytvoření vhodného modelu hodnocení testovacích procesů.

Publikace *Towards Agile Implementation of Test Maturity Model Integration (TMMi) Level 2 using Scrum Practices* (2015) [12] si klade za cíl zlepšit proces testování pomocí podrobného mapování postupů TMMi a metodiky Scrum. Ověřuje toto mapování se studií, která poskytuje empirické důkazy o získaných výsledcích. Výzkum byl vyhodnocen na vzorku dvou velkých společností, které jsou certifikovány TMMi Foundation. Na závěr experimentální výsledky poukazují na účinnost použití tohoto integrovaného přístupu ve srovnání s jinými existujícími přístupy.

Software test maturity assessment and test process improvement: A multivocal literature review (2017) [13] Autoři tvrdí, že postupy v oblasti testování softwaru nejsou v mnoha společnostech dostatečně vyspělé a obvykle se provádějí nahodile. Takové nevyspělé postupy vedou k různým negativním výsledkům, např. k neúčinnosti testovacích postupů při zjišťování všech vad a k překročení nákladů a plánů testovacích činností. Za účelem systematického posuzování zralosti testování a zlepšování procesů testování, byly navrženy různé přístupy a rámce. Aby byly zjištěny nejmodernější a nejpraktičtější postupy v této oblasti, autoři této práce vypracovali „multivokální“ literární přehled (systematický přehled z různých zdrojů) odborné literatury a také šedé literatury odborníků (např. blogové příspěvky). Díky tomu zjistili 58 různých modelů testovací zralosti a velké množství zdrojů s různými stupni empirických důkazů na toto téma. Práce také obsahuje stručné porovnání nejpoužívanějších modelů z této oblasti.

Kniha *Software quality assurance: From Theory to Implementation* (2004) [14] se zaměřuje na kvalitu softwaru, její zajišťování a oblast testování. Zaměřuje se také na

¹ Multivokální literární přehled – systematický přehled informací z různých typů zdrojů a pohledů od různých autorů a skupin (akademici, novináři, studenti atd.).

certifikace a standardy týkající se uvedených oblastí. Čtenáři může vhodně posloužit jako ucelený zdroj informací ohledně zajišťování kvality softwaru. Tento zdroj byl využit pro účely teoretické části této práce.

Kniha *Zlepšování procesů při budování informačních systémů* (2018) [15] čtenáři představuje normy, standardy a metodiky, které jsou vhodné pro zlepšování procesů při budování IS. Jsou zde přehledně popsány současné metodiky vývoje. Práce se podrobněji zabývá modelem CMMI, ze kterého model TMMi vychází.

Internetové zdroje

Internetové zdroje nejčastěji využívané v této práci, jsou ve většině případech zastoupeny webovými prezentacemi jednotlivých společností, které poskytují služby v oblastech zajišťování kvality softwaru. Na těchto stránkách jsou často publikovány nejnovější informace týkající se problematiky, kterou se daná společnost zabývá.

Většina potřebných zdrojů pro účely této práce se nachází na stránkách TMMi Foundation <https://www.tmmi.org/>. Nacházejí se zde všechny nejaktuálnější dokumenty týkající se stavu a vývoje modelu TMMi.

Na webu <https://qacomplete.com/> lze nalézt přehledně situované zdroje, které se věnují současným trendům, nástrojům a metodikám v oblasti testování.

1. Řízení kvality softwaru

Cílem této kapitoly je ucelit teoretické poznatky v oblastech: kvalita softwaru, řízení kvality softwaru a testování softwaru.

V úvodu této kapitoly je nutno podotknout, že řízení kvality softwaru je jen jednou oblastí komplexního rámce, který se nazývá management kvality softwaru (SQM – Software Quality Management) [16]. Tento rámec má zjednodušeně za úkol zajistit kvalitu výsledného softwarového produktu. Zajišťuje, plánuje a spravuje veškeré procesy organizace, které mají na kvalitu vliv. Tato práce se však zaměřuje pouze na dvě podoblasti SQM: Řízení kvality softwaru (SQC – Software Quality Control) a zlepšování kvality softwaru (SQI – Software Quality Improvement).

Tyto podoblasti jsou úzce spjaty s problematikou diplomové práce, jelikož hlavní činnost SQC představuje testování softwaru. To se skládá z jednotlivých procesů testování, jejichž kvalitu a zralost napomáhá určit a zvyšovat model TMMi. Zlepšování postupů a procesů však spadá pod oblast SQI (viz kapitola 2). Pro účely této práce je tedy zapotřebí objasnit tyto oblasti.

Pro ucelený vstup do problematiky diplomové práce je důležité objasnit základní pojmy a oblasti. V této kapitole jsou vysvětleny následující pojmy:

- Kvalita softwaru
- Řízení kvality softwaru (SQC)
- Testování softwaru

Pro oblast zlepšování kvality softwaru byla vymezena samostatná kapitola (viz kapitola 2).

1.1. Kvalita softwaru

Pojem *software* je téměř identicky definovaný mezinárodní organizací pro normalizaci (ISO) a Institutem pro elektrotechnické a elektronické inženýrství (IEEE) jako „*sada počítačových programů, postupů a případně související dokumentace a data týkající se provozu počítačového systému*“ [14]. Zjednodušeně se dá také říci, že software je v rámci IT vše, co není hardware. To je však velmi zjednodušená definice.

Historicky v rámci softwarového inženýrství byla kvalita jedním z několika důležitých faktorů. Radíme do nich náklady, plán a funkčnost softwarového produktu [17]. Tyto faktory

určují úspěch nebo selhání softwarového produktu na rozvíjejícím se trhu. Je třeba také zohlednit časové období a pro jaký tržní segment je software vyvíjen [17]. J. D. Musa a W. W. Everett ve své práci vydané IEEE [17] využili těchto faktorů k rozdělení softwarového inženýrství na čtyři oblasti:

1. Funkční oblast se soustřeďuje na poskytování automatizovaných funkcí, které nahrazují to, co bylo dříve prováděno manuálně.
2. Oblast plánování klade důraz na včasné, řádné zavádění důležitých prvků a nových systémů pro uspokojení naléhavých potřeb uživatelů.
3. Oblast nákladů se zaměřuje na takové snížení ceny, aby zůstala konkurenceschopná.
4. Oblast spolehlivosti zaměřuje svou pozornost na řízení uživatelských očekávání v rámci zvýšené závislosti na softwaru a vysoké náklady nebo vážné škody spojené se selháním softwaru [18].

Usilováním o definování kvality softwaru a jejích atributů vzniklo několik různých modelů v rámci mezinárodních norem. Mezi ně patří např. normy řady ISO/IEC 9126 (Softwarové inženýrství – jakost produktu) [19] a norma ISO/IEC 12119 (Softwarové balíky – Požadavky na jakost a zkoušení) [20]. Z důvodu značných nedostatků uvedených norem, mezi které patří především zastaralost a nekonzistence, vznikl jednotný systém norem ISO/IEC 25000-25099 v rámci projektu SQuaRe (Software Quality Requirements and Evaluation – Požadavky na kvalitu softwaru a její hodnocení). Jedná se o projekt společného technického podvýboru (ISO/IEC JTC1/SC 7) organizací ISO a IEC. [15]

Jednou z těchto vydaných norem je norma ISO/IEC 25010 [21], podle které kvalita softwaru představuje, do jaké míry softwarový produkt splňuje stanovené a implicitní potřeby, je-li používán za stanovených podmínek. Norma definuje model kvality produktu a model kvality užití. Kvalita softwarového produktu je zde určována na základě osmi charakteristik:

- Funkčnost
- Účinnost
- Kompatibilita
- Použitelnost
- Bezporuchovost
- Udržovatelnost
- Přenositelnost

Jednotlivé charakteristiky obsahují další podcharakteristiky, které jsou předmětem měření. Jednou z měř je i testování, které slouží jako hlavní zprostředkovatel informací o kvalitě

produktu. [22] Testování také představuje jednu z hlavních činností řízení/kontroly kvality (SQC).

1.2. Řízení/kontrola kvality softwaru

Řízení nebo také kontrola kvality softwaru (SQC) je podoblast spadající do oblasti managementu kvality softwaru (SQM). Má za úkol kontrolovat a řídit dodržování kvality softwarového produktu a hledat její nedostatky. Cílem SQC není pouze nacházet možné nedostatky vyvíjeného produktu, ale hlavně se snaží identifikovat jejich příčiny. Kontroluje, zda jsou naplněny všechny stanovené požadavky na softwarový produkt. Mezi hlavní činnosti SQC patří testování, revize a inspekce. Pojem SQC je v praxi často zaměňován za jiný pojem a tím je SQA (Software Quality Assurance) neboli zajišťování kvality softwaru. Hlavním rozdílem je skutečnost, že SQC se zaměřuje především na kontrolu a řízení kvality produktu jako takového, oproti tomu SQA se soustředí na procesy, postupy a standardy, které jsou v organizaci využívány a mají vliv na kvalitu výsledného produktu. Oblast zajišťování kvality obsahuje řadu aktivit, které zajišťují to, že nastavené procesy, postupy a standardy jsou pro danou společnost vhodné a správně naimplementovány. Oblasti SQC a SQA na sebe navazují a jsou vzájemně propojeny. Jejich postavení v organizaci záleží na tom, jak konkrétní organizace tyto oblasti vnímají. SQC tak může být podoblastí SQA, nebo mohou tyto dvě oblasti zastávat pozici na stejné úrovni „vedle sebe“. [23]

Hlavním hnacím motorem pro řízení a zajišťování kvality na současně pestrém a v některých odvětvích přehlceném trhu je konkurenceschopnost. Dostatečná kvalita vede ke spokojenosti zákazníka a vyšší produktivitě. V některých průmyslových a podnikových oblastech je však kvalita nutností, protože nekvalita softwarového produktu by mohla dokonce ohrožovat lidské životy. Z toho důvodu je využívání velmi propracovaných metodik a dodržování norem nezbytné. Zejména u automobilového, leteckého, chemického a jaderného průmyslu může nedodržení kvality zapříčinit fatální následky. Takovým příkladem může být také systém pro evidenci pacientů v nemocnici, který obsahuje veškeré informace o anamnéze, diagnóze pacienta, dávkování konkrétních léků, či plánované operace. I zde musí být systém naprosto spolehlivý, aby nebyl ohrožen život pacienta. [24]

Výslednou kvalitu vyvíjeného softwaru ovlivňují nejen procesy, které se týkají vývoje, ale i ty co zajišťují jeho komplexní životní cyklus. Jedná se tedy zejména i o následné aktualizace a další údržbu, jenž se provádí po dobu, kdy je již daný software v provozu u zákazníka. Tyto procesy nejsou samostatně oddělitelné, ale jsou propojené s dalšími podpůrnými procesy organizace. Z toho důvodu jsou v určité míře závislé na procesech, které se na první dojem

netýkají pouze vývoje. Z toho vyplývá, že kvalita produktu závisí i na celkovém procesním prostředí organizace. Proto se v organizaci zavádí zajištění kvality, které řeší správnost integrace procesů a principů s cílem poskytnout požadovanou úroveň kvality výsledného produktu. [9]

1.3. Testování softwaru

Testování softwaru představuje hlavní aktivitu, která spadá pod řízení kvality softwaru. Protože se diplomová práce zaměřuje na zvyšování zralosti procesů testování v agilních projektech, je třeba čtenáře nejprve seznámit s procesy testování a obecně testováním v agilním prostředí.

V první řadě je nutné vysvětlit, co pojem „testování softwaru“ vůbec znamená. Význam pojmu testování softwaru můžeme nalézt v několika možných definicích. Např. Institut pro elektrotechnické a elektronické inženýrství IEEE popisuje pojem testování ve dvou definicích. První z nich říká, že *„testování je proces provozování systému nebo jeho komponent za určitých podmínek, sledování nebo zaznamenávání výsledků a vyhodnocování sledovaného systému z nějakého hlediska.“* Ve druhé definici popisuje testování softwaru jako *„proces analyzování softwaru za účelem rozpoznávání rozdílů (defektů) mezi současným a požadovaným stavem a následným vyhodnocením zkoumaného softwaru“.* [14]

Zjednodušeně lze říci, že se jedná o proces sběru a analýzy informací o kvalitě softwarového produktu, jehož cílem je zjistit, zda produkt splňuje konkrétní či zjevné potřeby uživatelů. Testování má za úkol minimalizovat náklady na opravy případných chyb, protože čím dříve je chyba nalezena, tím méně nákladná je její oprava. V dnešní době se software netestuje pouze kvůli odhalení chyb. Cílem je, aby byl především zákazník s produktem spokojen. Testování řeší problémy dříve, než jsou odhaleny uživateli, což zvyšuje důvěru v samotnou aplikaci, ale i ve společnost, která aplikaci vyvíjí.

1.3.1. Techniky testování a členění testů

Existuje několik testovacích technik, které je nutné zvolit v závislosti na konkrétní testované aplikaci. Jejich volba je prováděna v rámci plánování testů, ve kterém se určí testovací techniky a dále pak podrobněji popisuje jejich provádění. V následující části je uvedeno základní členění těchto technik.

Statické a dynamické testování

V první řadě je testování rozlišováno podle potřeby funkčnosti daného softwaru. Přesněji tedy, zda je k otestování nutné jeho spuštění, či nikoliv. Tyto techniky se dělí na statické a dynamické testování. [25]

Statické testy nevyžadují běh softwaru. Může se tedy začít s testováním již v rané části vývoje daného softwaru, kdy zatím neexistuje jeho spustitelná verze. Statické testy jsou především prováděny pro kontrolu specifikací, požadavků a projektové dokumentace, což vede např. k přesnějšímu posouzení časové a nákladové náročnosti projektu. Nedílnou součástí je také tzv. „code review“ neboli kontrola kódu, zkoumání programového kódu, jeho syntaxe, přezkoumávání algoritmu. Statické testování tak může nalézt chyby již v rané fázi projektu a minimalizovat tak náklady na jejich opravu.

Naopak dynamické testy vyžadují spustitelnou verzi aplikace. Testování tak probíhá v pozdější fázi vývoje než statické. Od toho se odvíjí také vyšší složitost a náklady na opravu případně nalezených chyb. Provádí se na základě předem připravených testovacích scénářů, které obsahují testovací skripty (testovací případy, které tvoří skupinu logicky navazujících kroků). Pro dynamické testování je využíváno testů černé, šedé a bílé skřínky. Mohou být prováděny jak manuálně, tak i automatizovaně. [25]

Testování černé, bílé a šedé skřínky

V případě testování černé skřínky (také Black-box test) má tester přístup k programu jen přes stejné rozhraní jako zákazník nebo uživatel. Nemá tedy přístup ke zdrojovému kódu a testuje jen na základě uživatelských/testovacích dokumentací nebo podle vlastní intuice. Nezná, jak přesně systém pracuje s daty. Sleduje se, jaký výstup vznikne po zadání vstupních dat. Naopak u testů bílé skřínky (také White-box test) má tester přístup ke zdrojovému kódu programu a zná vnitřní datové a programové struktury. Je pak možné otestovat všechny průchody zdrojovým kódem, zadání neočekávaných vstupních hodnot a použít další testy, které se zakládají na kontrole zdrojového kódu. Šedá skříňka je kombinací černé a bílé skřínky. Tester tak má přístup ke zdrojovému kódu, ale testuje primárně přes uživatelské rozhraní. Přístup ke zdrojovému kódu využívá např. tak, že provedené operace přes uživatelské rozhraní následně kontroluje pomocí dotazů do databáze. [26]

Manuální a automatizované testování

Testování se dále dělí na manuální a automatizované. Ve většině případů mají stejnou váhu a vzájemně se tyto dvě techniky doplňují. Jak už název napovídá je tato testovací technika

prováděna manuálně, kde tester tzv. „proklikává“ a testuje daný produkt. Při testování pracovník využívá svých zkušeností, fantazie a intuice nebo předem připravených testovacích scénářů. Manuální testování má řadu výhod a nevýhod. Na rozdíl od automatizovaných testů může tester reportovat i přehlednost uživatelského rozhraní a dojmy z práce s produktem. Díky jeho vynalézavosti může přijít i na zatím nezjištěné scénáře, se kterými by se uživatel mohl setkat. Hlavní nevýhodou je ovšem časová náročnost a z toho vyplývající i vyšší náklady na zdroje. Manuální testování se časem pro testera stává velmi stereotypní, díky čemuž pak může docházet k poklesu efektivity. Tento problém řeší automatizované testování, kde nedochází k poklesu efektivity. Automatizované testování s sebou nese výrazné úspory prostředků a zvýšení kvality výsledného produktu. Je využíváno zejména na testování jednoduchých nebo dobře algoritmizovatelných operací ve velkém objemu. Funguje i jako rychlá zpětná vazba pro vývojáře. Vývojář tak nemusí čekat po každém commitu na testera, až dotestuje, ale může si předpřipravené testy spustit sám a okamžitě vidí výsledek jednotlivých testů. Testy se mohou i plánovaně automaticky spouštět, díky čemuž se může zajistit stálá kontrola nad běžící aplikací. Automatizace je také užitečná pro testování výkonu, kdy se na aplikaci např. pustí několik testů najednou a zkoumá se, zda aplikace zvládá dané zatížení. [27]

Zaměření testů podle dimenzí kvality softwarového produktu

Testy lze také dělit podle toho, na jakou oblast softwaru se zaměřují. Tyto oblasti (dimenze) jsou podrobně popsány v modelu s názvem FURPS [22]. Název FURPS je akronymem jednotlivých znaků kvality softwarového produktu, na které se tento model zaměřuje. Jsou jimi funkčnost (funkcionality), použitelnost (usability), spolehlivost (reliability), výkonnost (performance), rozšiřitelnost (supportability). Později bylo za potřebí vidět řešení z více dimenzí, což dalo za vznik rozšířené verzi FURPS+. Toto rozšíření se zaměřilo na klasifikaci dalších typů nefunkčních požadavků.

Model FURPS popisuje tyto dimenze [28]:

- Functionality (funkčnost) – představuje hlavní funkce softwarového produktu. Může také zahrnovat licencování, lokalizaci, online nápovědu, tisk, reportování, bezpečnost, správu systému nebo pracovní postup.
- Usability (použitelnost) – zde se zejména řeší vnímání uživatelského prostředí člověkem. Zahrnuje přístupnost, estetiku rozhraní a konzistenci v uživatelském rozhraní.
- Reliability (spolehlivost) – zahrnuje např. četnost selhání, zotavení při výpadku proudu, obnovitelnost systému po vypnutí atd.

- Performance (výkonnost) – výkon zahrnuje např. rychlost průchodu informací prostřednictvím systému, doba odezvy systému (která se také týká použitelnosti), doby zotavení a doby spuštění.
- Supportability (rozšiřitelnost/podporovatelnost) – do této dimenze řadíme testovatelnost, adaptabilitu, udržitelnost, kompatibilitu, konfigurovatelnost, instalovatelnost, škálovatelnost, lokalizovatelnost a podobně. Můžeme sem zahrnout také manuály a jiné uživatelské dokumentace.

V současnosti je možné se setkávat s rozšířenou verzí FURPS+. Ono “+” rozšiřuje FURPS o čtyři další dimenze [28]:

- Design constraints (omezení návrhu) – omezení návrhu, jak naznačuje název, omezuje návrh. Je to například požadavek na relační databázi na stanovení přístupu, který používáme při vývoji systému.
- Implementation constraints (požadavky na implementaci) – obsahuje vymezení použitých programovacích jazyků, standardů, omezení zdrojů apod.
- Interface constraints (požadavky na rozhraní) – u systémů je běžné, že komunikují s jinými systémy. Tato dimenze definuje potřebná omezení pro tuto komunikaci (formát vstupů a výstupů, čas atd.).
- Physical constraints (požadavky na fyzické vlastnosti) – fyzická omezení ovlivňují hardware použitý k uložení systému – například tvar, velikost a hmotnost.

Úrovně testů

Testy jsou rozlišovány podle konkrétní fáze, ve které se vývoj softwaru nachází. Výstupy těchto fází jsou následně testovány na úrovni testů, která náleží danému výstupu a oblasti, kterou je třeba otestovat. Testy dělíme následovně:

- Jednotkové testy – tento typ testů ve valné většině provádí programátoři. Ověřují si tím funkčnost jednotlivých částí (jednotek) napsaného kódu.
- Integrační testy – testuje se správnost propojení a komunikace mezi jednotlivými komponentami (jednotkami, moduly a většími celky aplikace).
- Systémové testy – probíhá zde celkové otestování výsledného produktu předtím, než se předá odběrateli k akceptačnímu testování. Testy jsou prováděny napříč všemi oblastmi, které popisuje model FURPS.
- Akceptační testy – tyto testy se provádějí na straně odběratele. Ověřují, zda jsou dodrženy všechny akceptační kritéria aplikace. Pokud jsou dodrženy, aplikace je přijata a nasazena k používání. [22]

1.3.2. Agilní testování a role testera

Tato práce se věnuje zlepšování procesů testování v agilním prostředí. Z toho důvodu je nutné objasnit, v čem agilní testování spočívá.

Pojem *agilní* lze vyložit několika způsoby. Ať již obecně nebo v souvislosti s IT světem je tento výraz spojován s dynamickým vývojem událostí. Termínem agilní můžeme označit jakýkoliv přístup k řízení projektů, který se v první řadě snaží naučit týmy principy spolupráce, flexibility, jednoduchosti, transparentnosti a schopnosti reagovat na zpětnou vazbu v průběhu celého procesu vývoje nového programu nebo produktu. [29]

Jak již bylo zmíněno v úvodní kapitole, agilní metodiky vývoje představují současný trend hlavně díky rychlosti vývoje a schopnostem flexibilně reagovat na změny požadavků. Vývoj v agilních metodikách je založen na iterativním a inkrementálním modelu životního cyklu. Je zde kladen důraz na zákazníka, který často bývá součástí týmu. Díky tomu zákazník získává absolutní přehled o vývoji produktu a může do něj průběžně zasahovat [15]. Na poli agilního vývoje se vyskytuje velké množství agilních metodik, které bývají průběžně modifikovány nebo kombinovány. Mezi nejznámější agilní metodiky zcela jistě patří SCRUM, XP, Kanban, TDD atd. Jejich podrobný popis však není cílem této diplomové práce už jen kvůli tomu, že bylo toto téma mnohokrát zpracováno. Přehledné informace lze např. získat z publikace „*Zlepšování procesů při budování informačních systémů*“ [15]. Pro cíle diplomové práce je však důležité čtenáři poskytnout přehled o agilním testování.

Pojmem *agilní testování* jsou označovány techniky a praktiky testování softwaru, které využívají hodnot a principů sepsaných v „*Manifestu agilního vývoje softwaru*“ [30]. Osoba, která pomocí agilního testování tvoří a provádí testy vyvíjeného softwaru v agilně řízeném projektu, se nazývá agilní tester. [6]

Na rozdíl od tradičních metodik se chápání role testera v agilním přístupu o hodně liší. Především díky tomu, že tester již není posledním článkem vývojového procesu a stává se důležitou součástí celého procesu vývoje. Agilní tester má v mnoha případech největší přehled o vyvíjeném produktu. Dalo by se tedy říct, že se stává klíčovým uživatelem, proto se počítá s jeho názory a nápady na úpravy řešení. Nejen, že se tester stal důležitým členem týmu, ale jsou na něj kladeny i rozsáhlejší požadavky na jejich schopnosti a znalosti, které byly v tradičních metodikách méně potřebné.

Agilní tester by měl být schopen psát a spouštět automatizované testy. Měl by také ovládat různé testovací techniky jako např. testování černé, šedé a bílé skřínky, tedy jak se znalostí,

tak i s neznalostí zdrojového kódu. Protože je na agilního testera kladeno více požadavků, bývají také více technicky zdatní než standardní testeři v tradičních metodikách. [6]

V agilních metodikách se od testerů také očekává velký důraz na spolupráci a komunikaci s celým týmem. U agilního testera je nutné, aby byl schopen plánovat a řídit svou práci, jelikož v agilním přístupu se spoléhá na samostatnost každého člena týmu. Zároveň je důležité, aby se tester rychle přizpůsoboval změnám, které je potřeba následně zahrnout při tvorbě a úpravě testů.

Dalo by se tedy říci, že v agilním přístupu mají testeři daleko více práce zejména z důvodu, že procesy testování probíhají po celou dobu trvání projektu, nikoliv až v poslední fázi vývoje systému, jako je tomu u tradičních metodik. Důležitá je komunikace s týmem, který odpovídá za konečný produkt jako celek. [6]

1.3.3. Procesy testování

Tato práce se zabývá zvyšováním zralosti procesů testování. Je tedy vhodné vymezit, o které procesy se jedná a jak je v této práci proces chápán. V oblasti vývoje softwaru je proces chápán jako „*Množina aktivit, praktik, metod a transformací, které lidé používají k vývoji a údržbě SW a souvisejících artefaktů (plány, dokumentace, ...)*“. [31] V práci se také často zmiňuje termín *praktika*. Praktika je zde chápána, jako osvědčený způsob/postup provádění nějaké činnosti/aktivity.

Procesy testování v agilním prostředí se od tradičních liší především dobou, kdy jsou jednotlivé procesy prováděny. V tradičních metodách je časové ukotvení těchto procesů jasně vymezeno a ve většině případech jsou procesy testování prováděny v pozdějších etapách vývojového cyklu. V případě agilního vývoje je provádění procesů testování takřka kontinuální záležitost. Procesy testování jsou prováděny prakticky od započatí vývoje. Většina těchto procesů probíhá opakovaně v každé vývojové iteraci s tím, že mohou být prováděny paralelně. [6]

Bez ohledu na to, jaké metodiky vývoje jsou ve společnosti využívány, testování se běžně zakládá na těchto základních procesech testování/procesních oblastech[32]:

- **Plánování testů** – v této části se určují rozsah, cíle a rizika testování. Zavádějí se zkušební metody. Vytváří se testovací politika a strategie.

- **Příprava a návrh testů** – dochází k přípravě všeho potřebného pro úspěšné zahájení provádění testů. Přípravuje se testovací prostředí, navrhuje se testovací případy, vytvářejí se automatizované testy, určují se priority testování např. na základě stanovených rizik atd.
- **Provádění testů** – zde jsou prováděny manuální i automatizované testy v souladu s testovací politikou, strategií a plánem. Při nalezení defektu se jeho charakteristika zaznamenává se všemi náležitostmi do předem určeného nástroje. Po opravě defektů dochází k opakovanému otestování dané části.
- **Monitorování a řízení testů** – průběh testů je průběžně monitorován a řízen. Kontroluje se např. čas vynaložený na provádění testů, zda je vše v souladu s plánem apod.
- **Vyhodnocení testů** – posuzují se výsledky testování, zda bylo vše dostatečně otestováno a může být proces ukončen. V této fázi mohou nastat tyto skutečnosti: konec testování, pokračování v testování, opakování testování. [32]

Tyto procesy se dále větví na další podprocesy a podpůrné procesy. Dokonce i tyto jednotlivé procesy mohou být pojaty jako podprocesy, nebo procesní oblasti v procesu testování. Pro objasnění kontextu procesů testování je uvedený popis dostačující. Mimo to se těmito procesy detailněji autor zabývá v kapitole 6.2 *Porovnání současného stavu proti druhé úrovni TMMi*. Podrobnější informace a popis procesů testování lze také dohledat v normě ISO/IEC/IEEE 29119 nebo dokumentech od ISTQB (*International Software Testing Qualifications Board*) [32]. Zvyšování zralosti a zlepšování těchto procesů je hlavní doménou této práce a tomuto tématu se věnují následující kapitoly.

2. Zlepšování kvality softwaru

Tato krátká kapitola má za cíl uvést a vymezit oblast zlepšování při vývoji softwarového produktu. Kapitola zároveň slouží jako úvod do následné užší oblasti *Zvyšování/Zlepšování zralosti procesů testování*, kterou se podrobněji zabývají další kapitoly.

Zlepšování kvality softwaru (SQI – software quality improvement) [33] představuje zjednodušeně snahu o kontinuální zlepšování, které vede k zefektivnění procesů, snížení nákladů, zvýšení produktivity, větší konkurenceschopnosti a zvyšující se kvalitě softwarového produktu. Snaží se o neustálé zlepšování např. vývojových procesů s cílem dosáhnout co nejlepších výsledků. Proces zlepšování často zasahuje do většiny podnikových procesů. V případě zlepšování procesů se častěji vyskytuje označení *Zlepšování softwarového procesu* (SPI – software process improvement). V oblasti vývoje softwaru je proces chápán jako „*Množina aktivit, praktik, metod a transformací, které lidé používají k vývoji a údržbě SW a souvisejících artefaktů (plány, dokumentace, ...)*“.[31]

Zlepšování je pro mnoho podniků působících na rychle se rozvíjícím trhu informačních technologií nepostradatelnou součástí. Pokud společnost začne ve zlepšování stagnovat, tak riskuje, že ji konkurence uteče. Proto je zlepšování nedílnou a často přirozenou součástí úspěšných společností a pracovních týmů, které působí na trhu informačních technologií.

Je známo, že kvalita procesů (zejména vývoje) velkou měrou ovlivňuje kvalitu produktu. V případě agilního vývoje však toto tvrzení není tak jednoznačné a je mnohdy rozporováno zastánci agilních přístupů. Podle jejich názoru se kvalita finálního softwarového produktu odvíjí především od schopností a zkušeností jednotlivých pracovníků a týmu jako celku [34]. V agilním vývoji se především klade důraz na komunikaci, flexibilní reakce na změny a zaměření se na zákazníka [34]. To však neznamená, že není třeba zlepšovat procesy a praktiky, které jsou v těchto procesech obsaženy. Podle autorova názoru je tedy v nejlepším případě vhodné se v oblasti zlepšování zaměřit na tým jako takový a zároveň na zlepšování procesů. Je na konkrétním agilním týmu, aby si sám rozhodl, zda je potřeba věnovat se zlepšování zralosti daného týmu, nebo procesů, které již jsou anebo by měly být ve společnosti zavedeny.

Pro zlepšování kvality softwaru existuje mnoho nástrojů a modelů, které se především zaměřují na kvalitu procesů, určení jejich zralosti a pomoc s jejich zlepšováním.

Mezi nejznámější modely a nástroje, které slouží k posuzování zralosti procesů za účelem jejich zlepšování, patří např. norma ISO/IEC 15504 [35], která je také známa pod označením SPICE a Integrační model zralosti CMMI [15]. Model CMMI se posléze stal základem ke vzniku modelu TMMi, který může sloužit buď jako jeho doplněk nebo samostatný rámec pro zvyšování zralosti procesů testování. Protože se práce věnuje především modelu TMMi, je třeba alespoň stručně popsat model ze kterého TMMi vychází (viz následující text).

2.1. CMMI (Capability Maturity Model Integration)

Existuje celá řada procesně orientovaných modelů. Ty si jsou často velmi podobné, liší se především sadami nástrojů, které uživatelům nabízí. Jedním z nich je i CMMI. Capability Maturity Model Integration může být do češtiny přeložen jako Integrační model zralosti [15]. CMMI v1.3 je dostupný zdarma, což představuje nezanedbatelnou výhodu oproti jiným modelům. U nové verze CMMI v2.0², která byla vydána v březnu 2018, tomu tak již není a za získání této verze je třeba zaplatit. Autorem modelu CMMI je tým pracující při Carnegie Mellon University, konkrétně jejich instituce SEI-CMU (Software Engineering Institute – Carnegie Mellon University). Zásady, na kterých je CMMI postaven, mají velmi blízko k jiným metodologiím, jako jsou např. ITIL a ISO. Oproti nim CMMI definuje pět úrovní zralosti, a tak tým může svou procesní zralost postupně zdokonalovat. [1]

Model CMMI se v průběhu let značně rozšířil. V současné době pokrývá tři klíčové zájmové oblasti:

- CMMI-DEV (Product and service development) – zaměřuje se na vývoj služeb a produktů.
- CMMI-SVC (Service establishment, management) – pokrývá zavádění služeb a jejich řízení.
- CMMI-ACQ (CMMI for Acquisition) - řeší ověřené postupy nákupu produktů a služeb.

CMMI vznikl spojením dvou podobných modelů z řady CMM (Capability Maturity Model – vyvíjen do roku 1997): SE-CMM (pro technické týmy) a SW-CMM (pro softwarové týmy). Slovo *integration* znamená integraci několika standardů dohromady.

² <https://cmmiinstitute.com/products/cmmi/cmmi-v2-products>

Hlavní podstatu modelu představuje přístup k procesní zralosti. CMMI je rozdělen na pět úrovní, přičemž dosahovat vyšších úrovní lze postupně. Díky tomu může společnost model zavádět přiměřeně ke svým schopnostem a předejít tak negativním dopadům na fungování firmy kvůli zavádění modelu s naddimenzovanými požadavky. Tyto úrovně jsou následující [15]:

- Úroveň 1 – Počáteční (Initial)
 - Procesy probíhají nahodile, nejsou definovány. Vše se řídí prakticky na základě předchozích zkušeností. Výstupy procesů jsou většinou v rukou konkrétních jedinců.
- Úroveň 2 – Řízená (Managed)
 - Společnost má plánované činnosti a stanovené řízení projektů (základy projektového managementu). Rostoucí společnosti této úrovně často dosahují samovolně na základě přirozené potřeby organizovat věci.
- Úroveň 3 – Definovaná (Defined)
 - Společnost má zásadní procesy jasně formulované, zdokumentované a řízené. Existují zavedené standardy, sady nástrojů, postupů.
- Úroveň 4 – Kvantitativně řízená (Quantitatively Managed)
 - Produkty i procesy jsou řízeny kvantitativně. To znamená, že je měřena výkonnost procesů a výstupy jsou tak předem předvídatelné.
- Úroveň 5 – Optimalizující (Optimized)
 - Společnost neustále zlepšuje/optimalizuje své procesy.

„CMMI vám nezakazuje věci udělat po svém, ale CMMI vám nabízí, jak to udělat tak, aby to bylo užitečné a udržitelné.“[1]

CMMI patří mezi nejznámější modely využívané k určování a zlepšování procesní zralosti organizace. Procesy týkající se oblasti testování softwaru jsou v tomto modelu zastoupeny jen částečně a nedá se tedy využít jako samostatný model pro zlepšování procesů testování. Na to však reaguje TMMi (Test Maturity Model Integration), který je na tuto oblast přímo zaměřen.

3. Zlepšování kvality procesů testování

Cílem této kapitoly je pochopit podstatu, význam a obsah problematiky zlepšování kvality/zvyšování zralosti procesů testování. Kapitola se také zaměřuje na modely, které jsou v této oblasti využívány. Práce se především věnuje modelu TMMi. V této kapitole jsou uvedeny a blíže popsány dvě hojně využívané alternativy k tomuto modelu, které slouží ke stejnému účelu.

Z předchozí kapitoly vyplývá, že procesy zavedené v organizaci, která vyvíjí software, mohou (viz kontroverze mezi zaměřením se na kvalitu procesů proti schopnostem členů týmu – Kapitola 2) mít dopad na finální kvalitu samotného produktu. Významný podíl má na kvalitě produktu oblast testování. Tato oblast přispívá k dosažení takové úrovně kvality softwarového produktu, na které jsou splněny všechny stanovené požadavky. Představuje významnou aktivitu pro hodnocení a zlepšování kvality produktu a hraje důležitou roli v řízení kvality (viz podkapitola 1.3).

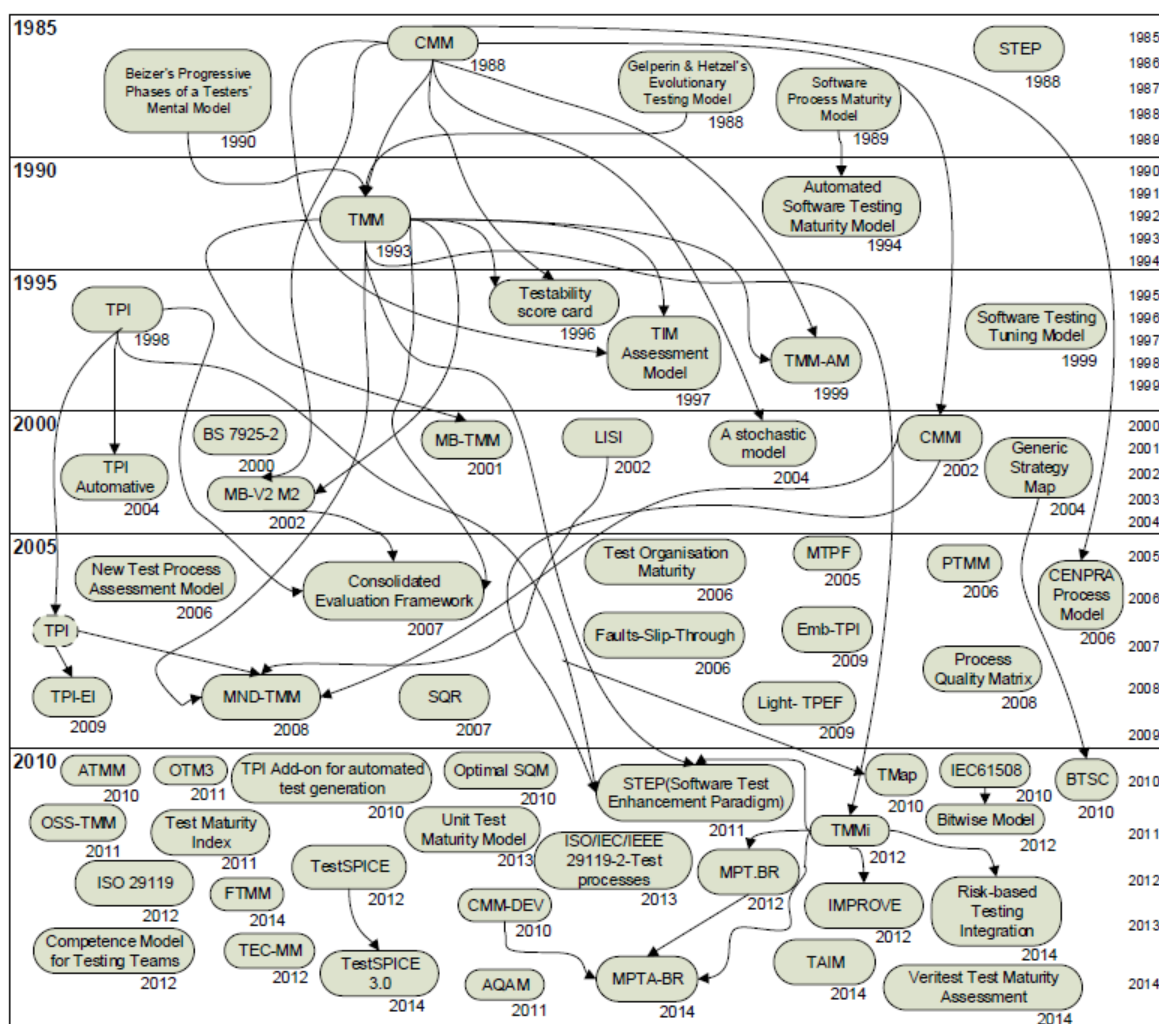
Testování představuje velmi rozsáhlou oblast procesů a aktivit/praktik, a proto je velmi náročné zajistit efektivnost a úspěšnost této oblasti bez jasně vymezených systematických postupů. Řešení této problematiky mohou představovat modely zlepšování procesů testování (TPI – Test Process Improvement) a modely hodnocení zralosti testování (TMA – Test Maturity Assessment). Modely představují soubory požadavků, které by procesy testování měly splňovat. Tyto požadavky jsou sestaveny na základě teorie a praktických zkušeností. V zásadě se jedná o soubor osvědčených postupů (best practices) v oblasti testování, které vedou ke zlepšení a zefektivnění procesů testování. [7]

Metody a modely uvedené v předchozí kapitole se zaměřují na komplexní kvalitu vývoje a ve většině případů testování berou jako samozřejmou součást. Na procesy testování a jejich zlepšování se však přímo nesoustřeďují, a proto vznikají modely, které se na řízení a zlepšování procesů testování přímo specializují. Modely jsou převážně stavěny na stupňovité struktuře. Díky té je umožněno její plynulé a postupné nasazování v rámci celé organizace nebo konkrétního projektu. Společnost si na základě svých potřeb a možností může vytyčit úroveň daného modelu, které chce dosáhnout s možností pozdějšího dosahování dalších úrovní. Modely na zlepšování kvality procesů testování patří mezi další nástroje SPI potažmo SQL.

Specialisté na softwarové inženýrství Vahid Garousi, Michael Felderer a Tuna Hacaloğlu ve svém průzkumu *Software test maturity assessment and test process improvement: A*

multivocal literature review [13], který je založen na multivokálním přehledu literatury, zjistili, že od devadesátých let konstantně roste vznik a nasazování modelů zaměřujících se na vyspělost a zlepšování procesů testování (TMA/TPI models). Podařilo se jim zmapovat 58 modelů. Toto číslo překvapilo i samotné autory průzkumu. Autoři těchto modelů se od sebe navzájem mnohdy inspirují. Modely se liší především zaměřením na specifické typy vývoje, na konkrétní typy testů, nebo pro jaké odvětví průmyslu je software vyvíjen. Vývoj těchto modelů i jejich vzájemné návaznosti znázorňuje diagram níže (viz Obrázek 1).

Z tohoto průzkumu se dalo odvodit, že mezi nejnovější a nejpopulárnější současné modely patří Test Maturity Model Integration (TMMi), Test process improvement (TPI) a TestSPICE. TMMi je hlavním tématem této diplomové práce a z toho důvodu je mu věnována značná pozornost v následujících kapitolách. V následující části této kapitoly jsou pro možnost srovnání stručně popsány dva zmiňované modely.



Obrázek 1 - Vývoj modelů TMA / TPI a jejich vztahů [13]

3.1. Test Process Improvement (TPI)

TPI model [36] pro hodnocení a zlepšování procesu testování patří mezi nejznámější modely v této oblasti. O její vývoj se zasloužila francouzská společnost SOGETI, která na trhu zastupuje roli poskytovatele testovacích služeb. Hlavní výhodou představuje jeho plynulá reprezentace a flexibilní nasazování. TPI nezávisle vychází z modelu T-Map (Test Management Approach) a aplikuje jeho myšlenky. [36]

Model obsahuje 20 klíčových oblastí u kterých se hodnotí jejich stupeň zralosti. Existují čtyři stupně zralosti, které jsou označeny písmeny A, B, C, D (A je nižší než D). Společnost si může podle vlastních priorit a požadavků určit pro každou klíčovou oblast cílovou úroveň zralosti. [7]

Každá úroveň se skládá z určitých požadavků na konkrétní klíčovou oblast. Požadavky určité úrovně rovněž obsahují požadavky na nižší úrovně. Tzn. proces testování na úrovni B splňuje požadavky obou úrovní A i B. V následující tabulce je uveden popis jednotlivých úrovní zralosti v závislosti na klíčových oblastech (viz Obrázek 2).

Levels	A	B	C	D
Key area				
Test strategy	Strategy for single high-level test	Combined strategy for high-level tests	Combined strategy for high-level tests plus low-level tests or evaluation	Combined strategy for all test and evaluation levels
Life-cycle model	Planning, Specification, Execution	Planning, Preparation, Specification, Execution, Completion		
Moment of involvement	Completion of test basis	Start of test basis	Start of requirements definition	Project initiation
Estimating and planning	Substantiated estimating and planning	Statistically substantiated estimating and planning		
Test specification techniques	Informal techniques	Formal techniques		
Static test techniques	Inspection of test basis	Checklists		
Metrics	Project metrics (product)	Project metrics (process)	System metrics	Organisation metrics (>1 system)
Test automation	Use of tools	Managed test automation	Optimal test automation	
Test environment	Managed and controlled environment	Testing in most suitable environment	Environment on call	
Office environment	Adequate and timely office environment			
Commitment and motivation	Assignment of budget and time	Testing integrated in project organisation	Test-engineering	
Test functions and training	Test manager and testers	(Formal) Methodical, technical and functional support, management	Formal internal Quality Assurance	
Scope of methodology	Project specific	Organisation generic	Organisation optimising (R&D)	
Communication	Internal communication	Project communication (defects, change control)	Communication within the organisation about the quality of the test processes	
Reporting	Defects	Progress (status of tests and products), activities (costs and time, milestones), defects with priorities	Risks and recommendations, substantiated with metrics	Recommendations have a Software Process Improvement character
Defect management	Internal defect management	Extended defect management with flexible reporting facilities	Project defect management	
Testware management	Internal testware management	External management of test basis and test object	Reusable testware	Traceability system requirements to test cases
Test process management	Planning and execution	Planning, execution, monitoring, and adjusting	Monitoring and adjusting within organisation	
Evaluation	Evaluation techniques	Evaluation strategy		
Low-level testing	Low-level test life-cycle: planning	White-box techniques	Low-level test strategy	

Obrázek 2 - Popis jednotlivých úrovní zralosti klíčových oblastí [36]

Hodnocení zralosti procesu testování probíhá formou kontrolních bodů pro každou klíčovou oblast. Kontrolní body prezentují jednotlivé otázky, na které se odpovídá buď ANO, NE případně n/a (neaplikovatelné). Úroveň zralosti dané klíčové oblasti je tedy splněna v případě, že na všechny otázky kontrolních bodů je odpovězeno kladně nebo n/a. [36]

Mezi jednotlivými úrovněmi klíčových oblastí vznikají závislosti, na které je třeba pamatovat. Např. aby bylo možné realizovat druhou úroveň konkrétní klíčové oblasti, je třeba mít zavedenou první úroveň jiné klíčové oblasti apod. Z toho důvodu TPI poskytuje matici zralosti (viz Obrázek 3), která obsahuje doporučení postupu zavádění jednotlivých úrovní pro konkrétní klíčové oblasti.

Vertikální osa matice označuje klíčové oblasti, horizontální osa zobrazuje stupnici zralosti. V matici každá úroveň souvisí s určitým stupněm testovací zralosti. Výsledkem je 13 stupňů testovací zralosti. Nevyplněné buňky mezi různými úrovněmi nemají samy o sobě žádný význam, ale naznačují, že dosažení vyšší zralosti v klíčové oblasti souvisí se zralostí ostatních klíčových oblastí. [36]

ID Klíčová oblast	0	1	2	3	4	5	6	7	8	9	10	11	12	13
	Ad Hoc	Řízené				Efektivní				Optimalizující				
1 Strategie testování		A					B				C		D	
2 Model životního cyklu		A			B									
3 Rannost testování			A				B				C		D	
4 Odhady a plánování				A							B			
5 Techniky specifikace testů		A		B										
6 Techniky statického testování					A		B							
7 Metriky						A			B			C		D
8 Nástroje testování					A			B			C			
9 Testovací prostředí				A				B						C
10 Pracovní prostředí				A										
11 Motivace a zapojení		A				B						C		
12 Pozice a testování				A			B			C				
13 Rozsah metodologie					A						B			C
14 Komunikace			A		B							C		
15 Reporting		A			B		C					D		
16 Řízení chyb		A				B		C						
17 Řízení nástrojů testování			A			B				C				D
18 Řízení procesu testování		A		B								C		
19 Evaluae							A			B				
20 Testování nízké úrovně					A		B		C					

Obrázek 3 - Matice zralosti TPI [7]

Oproti modelu TMMi je TPI více obecný a v popisu jednotlivých praktik pro dosažení vyšší úrovně méně detailní. Velikou výhodou oproti TMMi představuje hodnotící metoda, která je modelem TPI veřejně poskytována. Tato metoda zároveň pokrývá hodnocení procesů a zavádění změn. [7]

3.2. TestSPICE

Model TestSpice [37] vznikl prostřednictvím německé společnosti SQS na základě normy ISO / IEC 15504-5, která se zabývá posuzováním/hodnocením zralosti či stavu již

zavedených procesů a je historicky známa pod názvem SPICE (Software Process Improvement and Capability dEtermination). Model je založený na osvědčených postupech přejatých z ISTQB (*International Software Testing Qualifications Board*), modelů TPI a TMMi. Struktura samotného modelu byla přejata z ověřených referenčních procesních modelů ISO / IEC 15504. Od této normy se model prakticky liší jen v zaměření na vývoj specifických procesních referenčních modelů (PRM) a modelů hodnocení procesů, který v tomto případě vede ke zlepšení efektivity procesů testování. Zjednodušeně lze říci, že procesy obsažené ve SPICE byly jen namapovány na procesy testování např. jen změnou názvu procesu či úpravou některých požadavků. [38]

Na každý proces se váže sada požadavků. Tyto požadavky pak popisují a určují, čeho se musí dosáhnout, aby proces dosáhl kýžené úrovně. Každé vyšší úrovně lze dosáhnout pouze v případě, že byla dosáhnuta úroveň o jeden stupeň nižší.

Hodnocení zralosti procesů je zde definováno šesti úrovněmi:

1. Nekompletní – proces není schopen dosáhnout svého účelu, není implementován.
2. Vykonávaná – proces je realizovaný a dosahuje svého účelu.
3. Řízená – proces je realizovaný a řízený. Procesy na této úrovni jsou plánované, monitorované. Výstupy procesů jsou dále kontrolovány.
4. Zavedená – proces je definovaný, jsou jasně popsány interakce s jinými procesy, stanoveny kompetentní role pro provádění procesů. Jsou stanoveny metody pro sledování účinnosti procesu.
5. Předvídatelná – proces je měřený. Na základě měření se dá předvídat míra dosažení výkonnostních cílů.
6. Optimalizovaná – procesy jsou na základě měření analyzovány s cílem inovace a optimalizace jednotlivých procesů. [35]

K samotnému hodnocení konkrétních procesů se využívá čtyř bodová stupnice, která určuje míru splnění požadavků pro konkrétní úroveň (viz Tabulka 2).

N	Not achieved (nedosaženo)	0–15 % dosaženo
P	Partially achieved (částečně dosaženo)	>15 % - 50 % dosaženo
L	Largely achieved (z velké části dosaženo)	>50 % - 85 % dosaženo
F	Fully achieved (plně dosaženo)	>85 % - 100 % dosaženo

Tabulka 2 - Stupnice hodnocení atributů procesů (SPICE) [35]

Nasazení modelu TestSPICE je velmi pohodlné pro organizace, které již používají model SPICE, jenž je dokumentován v normě ISO / IEC 15504. Organizace se zavedeným modelem SPICE mohou bez obtíží přidat TestSPICE již k zavedenému modelu hodnocení procesů a získají tak rozšíření se zaměřením pro hodnocení procesů testování. Organizace, které nevyužívají SPICE nebo jiné modely hodnocení procesů, musí nasadit kompletní model hodnocení procesů, který pokrývá všechny aspekty hodnocení procesů testování. [38]

Model TestSPICE je možné stejně jako model TPI nasazovat plynule. Což znamená, že úroveň zralosti se hodnotí u každého procesu zvlášť, a proto si organizace může lépe nastavit zavádění modelu podle svých priorit. Model také podporuje nasazení na agilní projekty. [37]

4. Test Maturity Model Integration

Model TMMi a jeho možnost uplatnění v agilním prostředí představuje klíčovou část této diplomové práce. Z toho důvodu je tomuto modelu věnována celá kapitola. Cílem této kapitoly je představit model TMMi, který slouží ke zvyšování zralosti procesů testování. V této kapitole je model detailně popsán. Jsou zde vymezeny všechny jeho úrovně, komponenty a informace o hodnocení procesů. Důležitou část také představuje podkapitola týkající se použitelnosti modelu v agilním prostředí. Informace o modelu jsou převážně čerpány z hlavního dokumentu *Test Maturity Model integration (TMMi)* [3], který poskytuje TMMi Foundation.

4.1. Základní informace

Model známý pod názvem Test Maturity Model Integration, byl vyvinut nadací TMMi (TMMi Foundation) jako orientační a referenční rámec pro zlepšování procesů testování. Jedná se o doplňkový model k CMMI (konkrétně verze 1.2) [3], který se zabývá otázkami důležitými pro test manažery, test inženýry a odborníky na kvalitu softwaru. Testování definované v TMMi je pojato v co nejširším smyslu, aby zahrnovalo veškeré aktivity spojené s kvalitou softwarových produktů. Stejně jako CMMI tak i TMMi využívá koncept úrovní zralosti pro hodnocení a zlepšování procesů. Dále identifikuje procesní oblasti, cíle a postupy. Kritéria zralosti podle TMMi mají za úkol zlepšit proces testování a pozitivně ovlivnit kvalitu softwarového produktu. V rámci TMMi se testování postupně vyvíjí z chaotického, špatně definovaného, s nedostatkem zdrojů a nástrojů k vyspělému a kontrolovanému procesu, který má jako hlavní cíl prevenci defektů, a nejen jejich detekci. [3]

TMMi řeší všechny úrovně testů (včetně statických testů). Pokud jde o dynamické testování, spadají do oblasti působnosti TMMi jak nižší úrovně testů (např. unit testy, integrační testy), tak vyšší úrovně testů (např. systémové testy, akceptační testy). TMMi poskytuje kompletní rámec, který má být používán jako referenční model během zlepšování procesu testování na všech úrovních. [3]

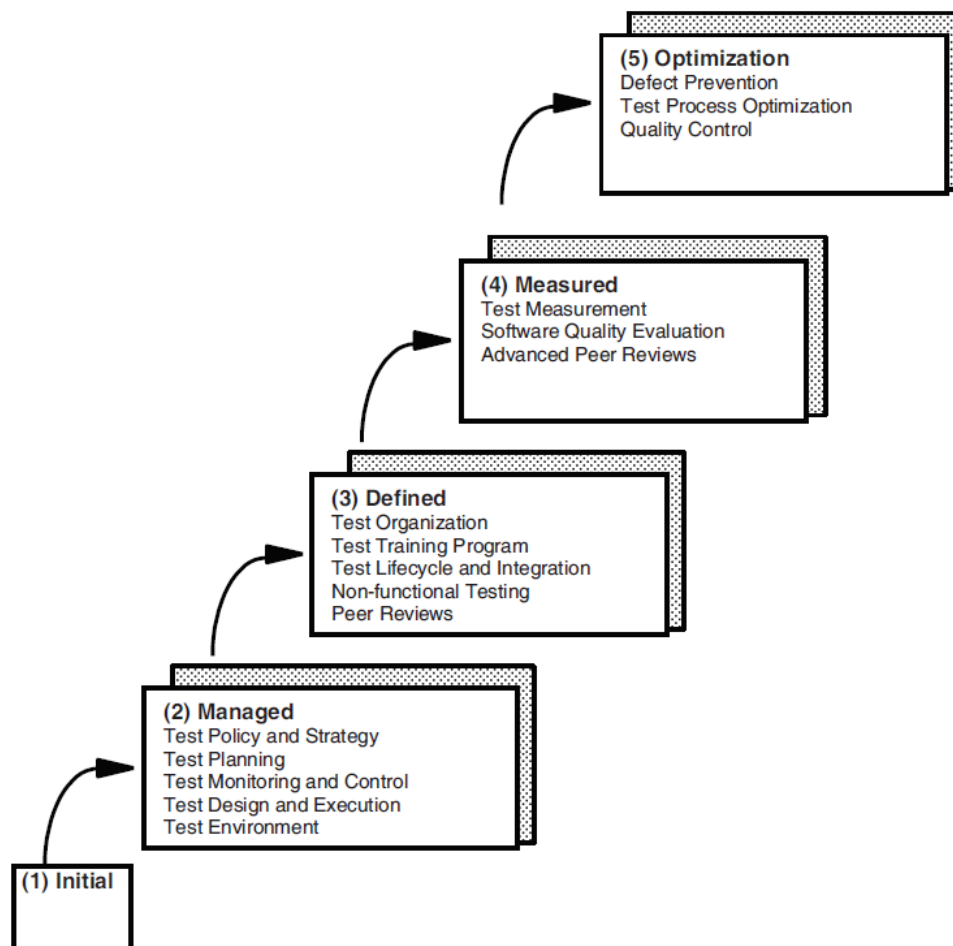
4.2. Úrovně zralosti dle TMMi

Model TMMi má odstupňovanou architekturu pro postupné zlepšování procesů testování. Dosažením každé úrovně se zajišťuje odpovídající zlepšení, které může sloužit jako základ pro další úroveň. Struktura modelu obsahuje 5 úrovní, přičemž každá úroveň představuje stupeň zralosti procesů testování. Vyšší úroveň vždy představuje evoluční cestu za

postupným zlepšováním. Každá úroveň obsahuje soubor procesních oblastí, které organizace musí implementovat, aby dosáhla zralosti na dané úrovni. Jelikož každá nižší úroveň tvoří základ pro úroveň vyšší, je kontraproduktivní úrovně přeskakovat. Úrovně, které jsou blíže popsány v další části této kapitoly jsou uvedeny pod těmito názvy [3]:

1. Úroveň – Initial („Počáteční“)
2. Úroveň – Managed („Řízená“)
3. Úroveň – Defined („Definovaná“)
4. Úroveň – Measured („Měřená“)
5. Úroveň – Optimization („Optimalizovaná“)

Na následujícím obrázku je graficky znázorněna struktura modelu TMMi se všemi pěti úrovněmi a oblastmi, které jednotlivé úrovně řeší (viz Obrázek 4).



Obrázek 4 - Úrovně zralosti a procesní oblasti modelu TMMi[3]

4.2.1. První úroveň – Initial („Počáteční“)

Počáteční úroveň zralosti představuje procesy testování jako neuspořádané a nedefinované. Protože organizace na této úrovni obvykle neposkytuje stabilní prostředí na podporu procesů testování, tak úspěch projektů často závisí na kompetenci a hrdinství lidí v organizaci, nikoliv na použití osvědčených procesů. Testy jsou vytvářeny ad hoc po dokončení programování. Cílem testování na této úrovni je ukázat, že software běží bez závažných defektů. Produkty jsou vydávány bez dostatečného ověření kvality a potenciálních rizik. Často se stává, že takovýto produkt, který byl uveden do provozu, nesplňuje všechny požadavky, je nestabilní, nebo má velkou odezvu (je pomalý). Při testování se využívá nedostatek prostředků, nástrojů a dostatečně kompetentních pracovníků. V této úrovni TMMi nejsou definovány žádné procesní oblasti jako je tomu u vyšších úrovní. Organizace na této úrovni mají tendence k opouštění zaběhnutých procesů v době krize a nejsou schopné reprodukovat jejich úspěchy. Charakteristické pro ně také bývá nedodržování smluvních termínů na dodávku produktu, rozpočty bývají překračovány a kvalita dodávek často nebývá podle očekávání. [3]

4.2.2. Druhá úroveň – Managed („Řízená“)

Na druhé úrovni modelu TMMi se testování stává již řízeným procesem. Tato úroveň napomáhá k udržení stávajících postupů i při stresových situacích, které se při vývoji vyskytují. Pro zkvalitnění procesů testování je testovací strategie zavedena do celé organizace nebo projektu. Vytváří se zde testovací plán, jehož součástí je i podrobný systematický přístup k testování. Přístup k testování je založen na vyhodnocení rizik produktu, která vyplývají ze zdokumentovaných požadavků na daný produkt.

Testovací plán definuje, jaké testování je požadováno, kdy, jak a kým. Tvorbu plánu a přístupu mají na starost zainteresované strany (dodavatel, zákazník) a mohou je podle potřeby upravovat. Testování je monitorováno a řízeno tak, aby bylo zajištěno, že vše probíhá podle plánu. Díky monitoringu jsou zjistitelné odchylky od plánu, na které se dá v krátké době reagovat přijmutím různých opatření. Management má díky tomu skvělý přehled o celém průběhu procesu testování.

Druhá úroveň pokrývá všechny hlavní úrovně testování: Unit, Integrační, Systémové a Akceptační testy. Pro jednotlivé úrovně testu jsou určeny cíle, které se pak promítají do organizační nebo projektové strategie.

Pro splnění 2. úrovně TMMi musí být zavedeny tyto procesní oblasti:

- Test Policy and Strategy („Politika a strategie testování“)
- Test Planning („Plánování testů“)
- Test Monitoring and Control („Monitorování a kontrola testů“)
- Test Design and Execution („Návrh a provedení testů“)
- Test Environment („Testovací prostředí“)

Druhá úroveň modelu TMMi si klade za cíl ověření, že produkt splňuje veškeré stanovené požadavky. Stále se zde však počítá s testováním až po naprogramování softwaru. Daleko vhodnější je začít s testováním již v rannějších fázích životního cyklu softwaru např. při návrhu nebo během fáze programování. Z toho důvodu nalezené chyby často pramení z rannějších fází životního cyklu např. špatně definované požadavky nebo samotný návrh softwaru. [3]

4.2.3. Třetí úroveň – Defined („Definovaná“)

Třetí úroveň Definovaná již integruje proces testování do všech etap životního cyklu vývoje softwaru. Testy se plánují v počáteční fázi vývoje již během definování požadavků. Vzniká zde tzv. master test plan, ten sjednocuje ostatní testovací plány a pokrývá všechny úrovně testování. V této úrovni jsou zavedeny standardizované procesy testování, role pokrývající testování jsou vedeny jako plnohodnotná profese a v zájmu organizace probíhají školení těchto zaměstnanců.

Návrhy testů v druhé úrovni TMMi se převážně zaměřují na funkcionální testy, přičemž na třetí úrovni modelu TMMi se testy a zkušební techniky rozšiřují tak, aby zahrnovaly i nefunkcionální testování, které zahrnují např. testování použitelnosti, spolehlivost a bezpečnosti v závislosti na obchodních cílech. Do nefunkcionálního testování spadá také revize dokumentace a požadavků, kterou provádějí profesionální odborníci již ve fázi návrhu softwaru.

Hlavní rozdíl mezi druhou a třetí úrovní zralosti představuje podrobnější dokumentace a popis jednotlivých procesů a postupů. Procesy testování jsou zde přizpůsobeny a standardizovány pro určité typy projektů v rámci celé organizace. Díky tomu jsou procesy konzistentnější. Oproti tomu na druhé úrovni se mohou na konkrétních projektech jednotlivé procesy zcela lišit.

Pro splnění 3. úrovně TMMi musí být zavedeny tyto procesní oblasti:

- Test Organization („Organizace testování“)
- Test Training Program („Školení zaměstnanců“)
- Test Lifecycle and Integration („Testovací životní cyklus a integrace“)
- Non-functional Testing („Nefunkcionální testování“)
- Peer Reviews („Revize“) [3]

4.2.4. Čtvrtá úroveň – Measured („Měřená“)

Díky dosažení dvou předchozích úrovní TMMi se testování může stát plně měřeným procesem, což velkou měrou dopomáhá ke zkvalitňování procesů v dané oblasti.

Pro proces testování se využívá specifický nástroj pro monitorování a správu testování (např. HP ALM). Takovýto nástroj poskytuje vyhodnocování kvality procesu testování, k posuzování produktivity a sledování zlepšení. Každá měření jsou ukládána na uložistích a slouží jako podpora pro další rozhodování. Prostřednictvím nástroje lze také zjistit výkon testů a předpovědi týkající se nákladů.

Přítomnost měřicího nástroje umožňuje organizaci provádět proces hodnocení kvality výrobků tím, že definuje kvalitativní potřeby, atributy kvality a metriky kvality. Produkt čelí hodnocení pomocí kvantitativních kritérií pro kvalitativní atributy, jako je spolehlivost, použitelnost a udržitelnost. Kvalita produktu se chápe kvantitativně a řídí se definovanými cíli po celý životní cyklus.

Revize a inspekce se považují za součást procesu testování. Používají se k měření kvality výrobku již v rané fázi životního cyklu a kladou si za cíl optimalizaci a zefektivnění testování.

Pro splnění 4. úrovně TMMi musí být zavedeny tyto procesní oblasti:

- Test Measurement („Měření testů“)
- Product Quality Evaluation („Hodnocení kvality produktu“)
- Advanced Peer Reviews („Pokročilé revize“) [3]

4.2.5. Pátá úroveň – Optimization („Optimalizovaná“)

Dosažením všech předchozích úrovní modelu TMMi je proces testování plně definovaný a měřený. Na páté úrovni je organizace schopna kontinuálně zlepšovat testování na základě

kvantitativního chápání statisticky řízených procesů. K postupnému zlepšování dochází např. za pomoci využívání nových technologií a optimalizací procesů testování.

Důležitou procesní oblast zde představuje prevence chyb. Nejde tedy pouze o hledání a reportování chyb, ale na základě identifikace a analýzy příčin vad v celém vývojovém životním cyklu se proces prevence snaží těmto chybám předejít. Předcházení defektům, kontrola kvality a optimalizace procesů testování slouží pro průběžné zlepšování procesů. Díky optimalizaci jsou procesy testování znovupoužitelné. [3]

Pro splnění 5. úrovně TMMi musí být zavedeny tyto procesní oblasti:

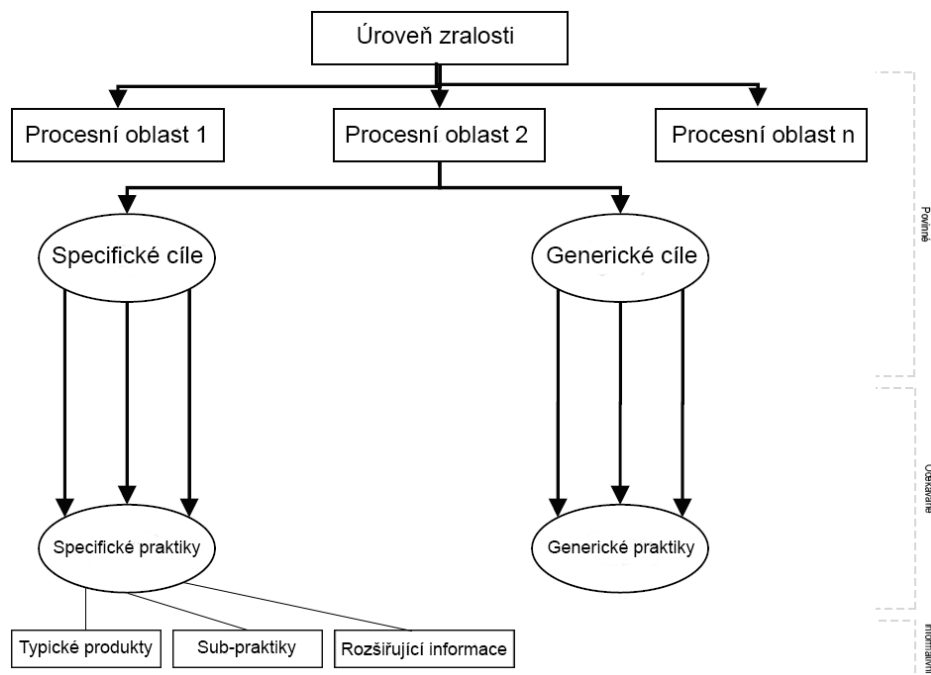
- Defect Prevention („Prevence defektů“)
- Quality Control („Kontrola kvality“)
- Test Process Optimization („Optimalizace procesů testování“)

Organizace, která dosáhla páté a poslední úrovně modelu TMMi, má plně optimalizované procesy. Pro tuto úroveň podle TMMi platí následující charakteristiky procesů:

- Řízené, definované, měřené, účinné
- Statisticky řízené a předvídatelné
- Zaměřené na prevenci defektů
- Podpora automatizace je považována za efektivní využití zdrojů
- Schopnost podporovat přenos technologií z průmyslu do organizace
- Schopnost znovupoužitelnosti
- Zaměřuje se na změny procesu s cílem dosáhnout neustálého zlepšování. [3]

4.3. Komponenty TMMi

Pro dosažení konkrétní úrovně zralosti musí mít organizace zavedené procesní oblasti, které jsou pro danou úroveň přesně definovány. Každá procesní oblast musí splnit určitá kritéria, která jsou zastoupena tzv. komponenty. Pro lepší představu je níže graficky znázorněna struktura modelu TMMi s obsaženými komponenty (viz Obrázek 5).



Obrázek 5 - Struktura a komponenty modelu TMMi, Upravená verze z [3]

Procesní oblasti se skládají ze tří typů komponent:

- **Povinné komponenty (Required Components)** – popisují, čeho musí organizace dosáhnout, aby uspokojila procesní oblast pro danou úroveň TMMi. Do těchto komponent patří:
 - **Specifické cíle (Specific goals)** – specificky charakterizované cíle, které musejí být bezpodmínečně splněny, aby byla procesní oblast dodržena.
 - **Generické cíle (Generic goals)** – tyto cíle se objevují ve více procesních oblastech, proto se nazývají generické. Popisují charakteristiky, které musí být splněny pro institucionalizaci procesů.
- **Očekávané komponenty (Expected Components)** – jsou využívány k dosažení povinných komponent. V případě, že nejsou využity definované praktiky, musí být nalezena jejich vhodná alternativa. Očekávané komponenty zahrnují:
 - **Specifické praktiky (Specific practices)** – popis aktivit, které vedou k dosažení specifických cílů.
 - **Generické praktiky (Generic practices)** - popis aktivit, které vedou k dosažení generických cílů.
- **Informativní komponenty (Informative Components)** - informativní komponenty poskytují informace, které pomáhají organizacím začít uvažovat o tom, jak přistupovat k požadovaným a očekávaným komponentům. Tato komponenta obsahuje:

- **Sub-praktiky** – podrobný popis, který poskytuje pokyny pro interpretaci a implementaci konkrétní praktiky. Slouží jako informativní složka pro poskytnutí nápadů na zlepšení procesu testování.
- **Typické produkty** – jedná se o ukázkové příklady z praxe pro inspiraci.
- **Rozšiřující informace** – další podpůrné informace např. poznámky, příklady, reference apod. [3]

4.4. Hodnocení pro získání certifikace TMMi

Pro nasazení modelu TMMi slouží hlavní dokument, obsahující popis modelu a požadavky na procesy testování, pod názvem Test Maturity Model Integration (TMMi) s označením poslední vydané verze Release 1.0. Dosažení určité úrovně zralosti musí znamenat totéž pro různé hodnocené organizace. Pravidla pro dosažení této konzistence jsou obsažena v požadavcích na metody hodnocení procesů testování, které popisuje dokument TMMi Assessment Method Application Requirements (TAMAR). Tento dokument nabízí dvě možnosti způsobu hodnocení. [3]

Hodnocení může být prováděno formálním a neformálním způsobem. Formální způsob hodnocení je prováděn pod vedením akreditovaného hodnotitele. Akreditace vedoucího hodnotitele musí být dosaženo prostřednictvím nadace TMMi. Společnost, která chce být formálně certifikována, musí nechat vyškolit hodnotitele, který získá k dispozici akreditovanou metodu hodnocení, nebo může požádat externí společnost, která takovouto metodou hodnocení disponuje [39]. Poslední zmíněnou možnost do jisté míry omezuje skutečnost, že v současnosti disponuje akreditovanou metodou pro formální hodnocení pouze devět společností na světě [40]. Další možnost představuje žádost o akreditaci vlastní metody, která ovšem musí splňovat náročné požadavky uvedené v dokumentu TAMAR. Žádost je nutné odeslat organizaci TMMi s detailní dokumentací. Zpracování požadavku je zpoplatněno částkou 5000 USD a celková doba celého procesu akreditace i s připomínkami od organizace se odhaduje na dva měsíce [41]. Hlavním cílem formálního hodnocení je získání certifikace TMMi, která zaručuje dosaženou úroveň zralosti a zároveň splnění normy ISO/IEC 15504. Formální hodnocení vyžaduje vyšší úroveň důkazů a potvrzení pro dosažení specifických, obecných postupů a cílů procesních oblastí a úrovní zralosti v rámci TMMi oproti neformálnímu hodnocení. Pokud již dříve hodnocená společnost dosáhla další úrovně zralosti, tak musí dojít k novému hodnocení, kde při formálním způsobu hodnocení musí být posouzeno, zdali společnost stále splňuje všechny požadavky z úrovně předchozí [42].

Neformální hodnocení oproti formálnímu by mělo být vedeno zkušenými hodnotiteli, který však nemusí mít akreditaci k výkonu této role. Tato úroveň hodnocení je navržena jako počáteční indikativní pohled a rychlá kontrola, aby se zjistil stav momentálních procesů testování. Tato forma hodnocení umožňuje organizaci vytvořit si orientační názor na to, zda jsou její stávající postupy v souladu s TMMi. [42]

Podrobný popis hodnotící metody bohužel TMMi volně neposkytuje. Její získání je podmíněno absolvováním odborného školení, kde školený získá licenční balíček „TAM“ (TAM je zdokumentovaná akreditovaná formální metoda pro provádění hodnocení procesů proti TMMi), který obsahuje podpůrné artefakty a nástroje, které umožňují provádět formální a neformální hodnocení s důvěrou nadace TMMi. [39] Z tohoto důvodu je v praktické části využito neformální metody, která je blíže popsána v praktické části práce [viz kap. 5.4.]

4.5. Využití TMMi v agilním vývoji

TMMi bylo pro svou rozsáhlost a velké nároky na dokumentaci předurčováno především pro integraci do projektů, které jsou vyvíjeny pomocí tradičních metod. To TMMi Foundation vyvrací dokumentem „*TMMi in the Agile world*“ [5]. V dokumentu je popsáno nasazení modelu do agilního prostředí s cílem dosáhnout druhé úrovně zralosti. V roce 2018 vydal Eric van Veenandaal (místopředseda představenstva TMMi) knihu, která z tohoto dokumentu vychází. Tato kniha, která je k sehnání jako e-book pod názvem „*TMMi® in the Agile Era*“ [43], oproti původnímu dokumentu navíc obsahuje více teorie. Je tedy zcela patrné, že TMMi se vyvíjí s dobou a reaguje na nejnovější trendy, mezi které patří i popularita agilního vývoje.

Vývoj softwaru v agilním prostředí sebou nese řadu výhod, kterými jsou např. rychlá reakce na měnící se priority, rychlost uvolnění produktu na trh, zvýšení produktivity týmu a procesy testování jsou součástí celého životního cyklu projektu (ne tedy až na jeho konci, jak je zvykem u tradičních metodik) atd. Avšak stejně jako u většiny projektů se i agilně řízené projekty potýkají s kvalitou svých produktů, která se ve velké míře odvíjí od vyspělosti procesů testování. TMMi se snaží být obecným modelem použitelným pro různé modely životního cyklu a různá prostředí. [44]

Autor zmíněného dokumentu uvádí, že model TMMi a agilní vývoj rozhodně nejsou v rozporu a úspěšná integrace modelu přinese takovým projektům řadu výhod, které model TMMi nabízí. Použití modelu TMMi v agilním vývoji přináší systematičnost do testovacích

procesů/procesních oblastí a upomíná na kritické části, které bývají často opomíjeny – v agilním prostředí se jedná zejména o nefunkční testování.

Při implementaci TMMi je třeba vzít v úvahu, že záměrem modelu TMMi není striktní zavedení souboru praktik do organizace. TMMi napomáhá k nalezení takových testovacích oblastí, kde změna/zlepšení může přinést požadovanou hodnotu s ohledem na podnikatelské cíle organizace. Praktiky TMMi jsou pouze očekávanou komponentou (viz Podkapitola 4.3). Povinnou komponentu představují cíle jednotlivých procesních oblastí, kterých by dodržением těchto praktik mělo být dosaženo. Je tedy povoleno použít „alternativní“ praktiku s ohledem na definovanou praktiku TMMi tak, aby bylo dosaženo definovaného cíle a zároveň byla zachována smysluplnost využití takovéto praktiky pro konkrétní organizaci. Často se tedy může stát, že daná praktika je již splněna pomocí alternativní verze, kterou si organizace sama nastolila a vede ke splnění cíle k dané procesní oblasti. [43]

Aby hodnocení mohlo být konzistentní u všech hodnocených organizací, tak TMMi poskytuje hodnotitelům přehled použitelnosti procesních oblastí, specifických cílů a specifických postupů v agilním kontextu. Komponentám, které jsou v agilním prostředí neaplikovatelné se při hodnocení přiřadí známka NA (neaplikovatelné). Výčet komponent TMMi pro druhou úroveň a jejich aplikovatelnost v agilním prostředí je následující (viz Tabulka 3) [5]. Aby nedošlo ke špatné interpretaci názvů jednotlivých komponent modelu, jsou tyto názvy uvedeny v originálním jazyce.

Komponenty TMMi			Aplikovatelnost v agilním vývoji
PA1	Test Policy and Strategy		
	SG1	Establish a Test Policy	Aplikovatelný
	SG2	Establish a Test Strategy	Aplikovatelný
	SG3	Establish Test Performance Indicators	Aplikovatelný
PA2	Test Planning		
	SG1	Perform Risk Assessment	Aplikovatelný
	SG2	Establish a Test Approach	Částečně aplikovatelný
		SP2.3 Define entry criteria	NA
		SP2.5 Define suspension and resumption criteria	NA
	SG3	Establish Test Estimates	Částečně aplikovatelný

Komponenty TMMi			Aplikovatelnost v agilním vývoji
		SP3.2 Define test lifecycle	NA
	SG4	Develop a Test Plan	Aplikovatelný
	SG5	Obtain commitment to the Test Plan	Částečně aplikovatelný
		SP5.2 Reconcile work and resource levels	NA
PA3	Test Monitoring and Control		
	SG1	Monitor Test Progress Against Plan	Aplikovatelný
	SG2	Monitor Product Quality Against Plan and Expectations	Částečně Aplikovatelný
		SP2.3 Monitor entry criteria	NA
		SP2.5 Monitor suspension and resumption criteria	NA
	SG3	Manage Corrective Actions to Closure	Aplikovatelný
PA4	Test Design and Execution		
	SG1	Perform Test Analysis and Design using Test Design Techniques	Aplikovatelný
	SG2	Perform Test Implementation	Částečně Aplikovatelný
		SP2.3 Specify intake test	NA
		SP2.4 Develop test execution schedule	NA
	SG3	Perform Test Execution	Částečně Aplikovatelný
		SP3.1 Perform Intake Test	NA
	SG4	Manage Test Incidents to Closure	Aplikovatelný
PA5	Test Environment		
	SG1	Develop Test Environment Requirement	Aplikovatelný
	SG2	Perform Test Environment Implementation	Aplikovatelný
	SG3	Manage and Control Test Environments	Aplikovatelný

Tabulka 3 - Aplikovatelné a neaplikovatelné komponenty TMMi v agilním vývoji [5]

Jak už napovídá jeden z agilních principů „*Fungující software před vyčerpávající dokumentací*“ [30], tak agilní metodiky si na dokumentaci příliš nezakládají. Dokumentace je zde tvořena pouze tehdy, když je to nezbytné a je jí jednoznačně potřeba. Zde by mohlo dojít k nabytí dojmu, že TMMi není vhodná volba. Opak je však pravdou. Ač by se mohlo zdát, že TMMi si zakládá na podrobné dokumentaci, tak tomu tak není. TMMi má pro principy agilních metodik pochopení a nevyžaduje striktně tvoření podrobné dokumentace, jako tomu je u tradičních metodik. Například u specifického cíle druhé úrovně modelu

TMMi s názvem „*Vytváření testovacího plánu*“ (v originále „*SG 4 Develop a Test Plan*“) [3] se nachází praktika „*Založení testovacího plánu*“ (v originále „*SP 4.5 Establish the test plan*“) [3]. Tato praktika spočívá ve vytvoření dokumentu, který obsahuje obsáhlé informace ohledně naplánovaného testování. Konkrétně má tento dokument obsahovat identifikátor, vlastnosti aplikace, které se budou a nebudou testovat, harmonogram testování, požadavky na zdroje, odpovědnost apod. V agilním prostředí se takto podrobný dokument velmi často nevytváří, a když už, tak např. ve formě zápisků z pravidelných diskuzí, nebo prostřednictvím myšlenkové mapy. TMMi s tímto počítá a ve svém dokumentu „*TMMi in the Agile world*“ [5] uvádí, že tato agilní praktika je plně v souladu se specifickou praktikou „*Založení testovacího plánu*“ (v originále „*SP 4.5 Establish the test plan*“) [3]. Z tohoto příkladu lze odvodit, že TMMi opravdu toleruje praktiky agilních metodik a staví se k nim s větší benevolentností, než by se mohlo původně zdát. Vždy je však potřeba, aby využití praktiky vedlo ke splnění povinných komponent, jimiž jsou konkrétní cíle daných procesních oblastí.

4.6. Shrnutí modelu

Model TMMi, už jen díky množstvím všech požadavků, se řadí mezi nejrobustnější a nejobsáhlejší modely zaměřené na zlepšování procesů testování. Hlavní překážku, kvůli které by mohl od nasazení odradit, představuje jeho stupňovitá povaha. To znamená, že pro dosažení každé další úrovně je potřeba splnit veškeré požadavky úrovně předchozí. Společnosti si tedy nemohou vybrat jen konkrétní části, které považují pro své specifické potřeby za důležité. V takovém případě je více nasnadě zvolit jiný model např. TPI nebo TestSPICE. Tvůrci TMMi se svým modelem snaží zaručit komplexní pokrytí veškerých možných potřeb procesu testování. Model je strategicky navržen tak, aby při jeho kvalitním nasazení došlo k nepopíratelnému zlepšení, které sebou nese výrazné úspory a kvalitní výsledky procesů testování.

V době vzniku této diplomové práce na světovém trhu figuruje celkem 56 společností s certifikací TMMi. Z toho 6 společností s certifikací pro 2. úroveň zralosti (řízená), 31 společností s certifikací pro 3. úroveň zralosti (definovaná), 5 společností s certifikací pro 4. úroveň zralosti (měřená) a 14 společností s certifikací pro 5. úroveň zralosti (optimalizovaná). [45]

5. Hodnocená společnost

Před samotným hodnocením společnosti je nutné tuto společnost blíže specifikovat a zvolit takovou metodu hodnocení, která bude jejím potřebám vyhovovat. V této kapitole je představena společnost, která byla podrobena neformálnímu hodnocení procesů testování podle modelu TMMi a jeho druhé úrovni zralosti. Je zde popsán produkt, který je společností vyvíjen, její organizační struktura a současně využívaná metodika vývoje. K tomu, aby hodnocení bylo možné, musí být určena metoda hodnocení. Popis zvolené metody je také obsahem této kapitoly.

5.1. Představení hodnocené společnosti

Akciová společnost Glogster, a.s. vznikla v roce 2007 se sídlem v Praze a Bostonu jako startupový projekt. Zpočátku se jednalo o vizuální sociální síť, která umožňovala uživatelům vyjadřovat se prostřednictvím multimediálních nástěnek tzv. „glogů“. Tuto platformu si následně oblíbila celá řada pedagogů a začlenili práci s „glogy“ do své výuky. Na tuto skutečnost společnost zareagovala a v roce 2009 spustila novou službu, která je určena pro vzdělávací účely zejména základních škol. Služba GlogsterEdu je v současné době primárním produktem a slouží jako cloudová prezentační platforma pro interaktivní výuku. GlogsterEdu poskytuje uživatelům nástroj pro tvorbu „glogů“ a přístup do digitální knihovny s edukativním obsahem. Formát „glogu“ umožňuje kombinovat různé typy médií – videa, hudbu, text, grafické elementy, obrázky, 3D modely atd. na jedné stránce. Služba je běžně užívána jako prezentační nástroj ve výuce i mimo ni. Glogster nabízí k interaktivní spolupráci mezi pedagogy a žáky, rozvíjí digitální gramotnost, vztahy mezi učiteli a žáky. Přispívá také k prohloubení zájmu o danou výuku. Glogster poskytuje několik typů licencí. Ty se od sebe liší počtem a typem možných uživatelských účtů (školy, učitelé, studenti). Největší licenci představuje tzv. district administrátor, která umožňuje správu a vytváření účtů pro jednotlivé školy. Tyto školy následně spravují učitelské a studentské účty. Učitelé mohou využívat těchto funkcí: řadit své studenty do jednotlivých tříd, zadávat studentům úkoly ve formě projektů, známkovat tyto projekty, prohlížet studentské glogy, vytvářet studijní materiály apod. Uživatel si podle svých požadavků může vybrat, jak rozsáhlou licenci potřebuje.

Co se týče technologické části, Glogster je tvořen prostřednictvím PHP (skriptovací programovací jazyk), MySQL (databázový systém) HTML (HyperText Markup Language), JavaScript (skriptovací jazyk) a Flashe (grafický vektorový program pro tvorbu interaktivních webových aplikací od společnosti Adobe). Glogster je webová aplikace

přístupná z většiny moderních webových prohlížečů. Od roku 2014 je k dispozici také mobilní aplikace pro zařízení s operačním systémem iOS a od roku 2015 pro operační systém Android. V současné době podpora Flashe u webových prohlížečů postupně ustupuje (časté bezpečnostní problémy, vysoké nároky na systém) a na rok 2020 je naplánované jeho definitivní ukončení. [46] Z toho důvodu Glogster v současnosti řeší přechod všech flashových prvků aplikace na modernější technologii HTML5. Nejčerstvější novinkou je nyní zcela nový HTML5 editor, který kromě jiného podporuje umístění „živých“ 3D objektů. Dále umožňuje vkládání multimédií přetažením přímo z adresáře počítače či z otevřené webové stránky. Komplexní projekt Glogster je založen na kontinuálním vývoji, který reaguje na evoluci technologií, požadavky uživatelů a současné trendy.

5.2. Organizační struktura společnosti

Struktura společnosti se člení na organizační útvary. Organizační útvary jsou zde reprezentovány jednotlivými divizemi. Vyjma externích akcionářů se ve společnosti nachází přibližně 11 aktivních pracovníků, kteří pokrývají níže uvedené divize:

- CEO
- Péče o zákazníky
- Obchod (Sales)
- Marketing
- Finance
- IT
 - Projektový manažer
 - PHP Programátor
 - HTML5 Programátor
 - Frontend Programátor
 - Tester

Divize IT má na starost veškerý vývoj a správu aplikace, zodpovídá za kvalitu, provádí analýzy a reporting pro vedení společnosti a marketingové účely. Dále komunikuje s divizí péče o zákazníky, poskytuje jí technickou podporu a využívá tuto divizi pro zapracování nových podnětů od uživatelů. Autor této práce působí ve společnosti jako jediný tester, zároveň se podílí na návrhu a vedení konkrétních částí projektu, úzce spolupracuje s projektovým manažerem a členy vývojového týmu.

Projektový manažer se často i s částí vývojového týmu podílí na jiných projektech mimo společnost Glogster. Díky tomu si členové týmu do těchto projektů odnášejí osvědčené praktiky a metody vývoje, které postupně přizpůsobují a vylepšují napříč těmito projekty.

Proto by hodnocení a návrhy na zvýšení zralosti procesů testování mohly mít pozitivní vliv i na další projekty, které jsou řízeny obdobným způsobem jako ve společnosti Glogster.

5.3. Současná metodika vývoje a testování

Ještě před samotným hodnocením současných procesů testování dle modelu TMMi je pro lepší představivost nutné čtenáře stručně seznámit s využívanou metodikou vývoje.

Procesy testování jsou úzce spjaté s metodikou vývoje softwaru. Z toho důvodu je nutné nejprve vymezit kompletní proces vývoje ve společnosti Glogster, do kterého plynule vstupují i procesy testování.

Jak už z tématu diplomové práce vyplývá, společnost si prosazuje agilní praktiky a principy. S uvedením konkrétní agilní metodiky je to však složitější. Původně byla prosazována a nějakou dobu dodržována metodika SCRUM. V průběhu cca čtyř let došlo však k přizpůsobení této metodiky potřebám týmu. Dalo by se říci, že se v současné době využívá uvolněnější modifikace metodiky SCRUM. Modifikace spočívá např. v nahodilém sledu a trvání jednotlivých sprintů³. Jednotlivé sprinty nejsou detailněji plánovány. To znamená, že ve většině případech se nestanovuje přesný termín dokončení sprintu a požadavky týkající se jeho obsahu se mohou průběžně měnit.

Pro plánování a řízení vývoje je velmi důležitý produktový backlog⁴ (angl. product backlog), kde se zároveň nachází požadavky na nové funkcionality a zaznamenané defekty čekající na opravu. Funkcionality jsou ve většině případech zapisovány formou tzv. user stories⁵, nebo use cases⁶ - ty zároveň mnohdy obsahují i grafické náčrtky a obrázky. Pravidla zápisu těchto položek nejsou ve společnosti nijak formalizovány. Záleží pouze na potřebě, jak podrobně je nutné danou položku popsat, aby byla správně pochopena všemi, co s danou položkou přijdou do kontaktu.

³ Sprint – časový úsek iterativního vývoje. Smyslem sprintu je vydání zvolené množiny užitečných vlastností. [15]

⁴ Produktový backlog – seznam požadavků a funkcí softwaru. Ty se v seznamu většinou vyskytují ve formě User Stories nebo Use Cases. [6]

⁵ User Story – představuje popis konkrétní uživatelské potřeby. Specifikuje, kdo jí chce naplnit a proč. Je psaná ve srozumitelném neformálním jazyce, aby jí mohl každý rozumět. [6]

⁶ Use Case – obsahuje posloupnost kroků, která definuje interakce mezi uživatelem a systémem, nebo jen mezi systémy. [6]

Při plánování sprintu se z produktového backlogu vybírají jednotlivé položky s ohledem na jejich prioritu, závislost na dalších položkách či rychlost možného vyřešení. Během plánování sprintu má hlavní slovo projektový manažer, který však zohledňuje veškeré poznatky a připomínky všech členů týmu. V průběhu sprintu se mnohdy stává, že některé položky jsou problematičtější na vývoj, než se předpokládalo. Aby tyto položky zbytečně nebránily nasazení dříve vyvinutým funkcionalitám, tak se často mění pořadí vyvíjených položek, které jsou následně uvolňovány do testovacího prostředí a odtud jsou po odsouhlasení zveřejněny uživatelům. Může se tedy stát, že se úspěšně nasadí všechny zvolené položky hromadně, nebo dojde k rozpadnutí obsahu sprintu na menší části – některé položky se přesunou do nadcházejícího sprintu. Na základě těchto informací lze tvrdit, že podoba sprintů se průběžně mění, doba jejich trvání není jasně stanovena a odvíjí se od průběhu samotného vývoje.

Komunikace v týmu není nijak organizována např. formou denního scrumu (stand-up meeting), retrospektivy apod. Komunikace často probíhá mezi jednotlivci, nebo skupinově když je potřeba (tzv. ad-hoc).

Během vývoje se klade značný důraz na testování, které hraje velkou roli na kvalitě výstupu jednotlivých sprintů a posléze celého finálního produktu. Scrum obecně nedefinuje konkrétní procesy testování a techniky. Je tedy na týmu, jaké procesy a techniky testování si osvojí.

Ve společnosti je testování prováděno na všech úrovních. Úroveň jednotkových, integračních, systémových a značná část akceptačních testů je silně podpořena automatizací. Tyto testy se spouští prostřednictvím automatizačního nástroje Jenkins při každém nasazování nové funkcionality na testovací prostředí. Při neúspěšném provedení některého z testů dojde automaticky k zastavení celého buildu. Tímto se předchází nasazování nefunkčního kódu na testovací prostředí, které je určeno pro akceptační testování.

Akceptační testy, které nejsou vhodné pro automatizaci (např. testování UI), jsou prováděny manuálně po nasazení nové funkcionality na testovací prostředí. Pro akceptační testování zde není tvořena rozsáhlá dokumentace, jak je zvykem u tradičních metodik, ale spoléhá se především na schopnosti a důslednost testera. Tester nemá k dispozici detailně rozepsané testovací případy, ale využívá pro tyto účely popsané funkcionality (obecně položky viz Příloha B: Detail karty položky v produktovém backlogu) nacházející se v produktovém backlogu.

Produktový backlog je tvořen v nástroji Gitlab, který mimo jiné podporuje zobrazení jednotlivých položek prostřednictvím tzv. kanban tabule. Z této tabule lze přehledně vyčíst typ a stav jednotlivých položek na základě přiřazených štítků. Položky se seskupují např. podle toho, zda se jedná o nové funkcionality, chyby, zda čekají na vyřešení, nebo zda jsou již připraveny k otestování. Výsledky testování tester zaznamenává k jednotlivým testovaným položkám a rozhoduje, zda jsou připraveny k nasazení či se vrací zpět k programátorům na dořešení.

Společnost si zakládá na agilním přístupu. Neřídí se však striktním dodržováním jedné konkrétní agilní metodiky, ale využívá jejich kombinace a přizpůsobuje si je podle svých potřeb. Flexibilně se staví ke změnám a reaguje na současné trendy. Je otevřena k podnětům na zlepšování svých interních procesů. Vývojový tým se kontinuálně snaží vylepšovat mimo jiné i své procesy testování. Model TMMi umožňuje společnosti určit na jaké úrovni zralosti se tyto procesy nacházejí a identifikovat vyskytující se nedostatky. Na základě těchto informací dává možnost vytvořit návrh na eliminaci zjištěných nedostatků tak, aby bylo dosaženo při nejmenším druhé úrovně tohoto modelu.

5.4. Zvolená metoda pro hodnocení společnosti dle TMMi

Aby bylo uskutečnitelné vypracovat návrh na zlepšení procesů testování pro jednotlivé oblasti dle TMMi s cílem dosáhnout druhé úrovně tohoto modelu, je nutné nejprve v dané společnosti ohodnotit současný stav jednotlivých procesních oblastí. TMMi však veřejně neposkytuje metodu hodnocení. Je tedy nutné zvolit neoficiální metodu hodnocení, která byla např. navržena někým jiným a případně ji upravit potřebám hodnocené společnosti.

Neoficiální metody jsou k nalezení v různých odborných pracích např. *Software Test Process Assessment Methodology*[47], *Software quality control with the usage of IDEAL and TMMi models*[48], *A Framework for Maturity Assessment in Software Testing for Small and Medium-Sized Enterprises*[49], nebo *Practical software testing: a process-oriented approach*[50]. Vhodnou hodnoticí metodu lze také nalézt v diplomové práci Tomáše Doška s názvem *Modely zlepšování procesů testování softwaru a zajištění kvality* [7]. Autor této práce navrhl neformální metodu hodnocení, která nevyžaduje akreditovaného hodnotitele. Principiálně je tato metoda téměř shodná s metodami ve většině uvedených pracích. Důvod pro výběr této metody spočívá především v její jednoduchosti a zevrubnému popisu. Ačkoliv návrh metody vznikl již v roce 2012, požadavky na hodnocení se prakticky nezměnily, z toho důvodu navržená metoda

koresponduje s modelem TMMi i v současnosti a může být využita pro účely této diplomové práce. Místopředseda představenstva TMMi Foundation Eric van Veenendaal byl s navrženou metodou seznámen a poskytl tak autorovi hodnotnou zpětnou vazbu, což velmi přispělo k finální autenticitě navržené metody.

5.4.1. Popis zvolené hodnotící metody

Hodnotící metoda navržená Tomášem Doškem se skládá ze tří fází: Plánování, Sběr dat a porovnání, Vyhodnocení. Popis jednotlivých fází se v návrhu občas překrývá. Z toho důvodu následující text stručně popisuje metodu hodnocení tak, jak bude použita pro účel této práce. Cílem metody je ohodnotit současnou úroveň procesů testování v konkrétní společnosti a navrhnout úpravy podle zjištěných nedostatků tak, aby bylo dosaženo druhé úrovně modelu TMMi. [7] Navržená metoda je upravena pro potřeby této práce. Hlavní úpravu představuje skutečnost, že hodnocení je prováděno pouze proti specifickým cílům, a ne proti konkrétním praktikám, kterými jsou tyto cíle dosahovány. Pro agilní prostředí jsou tyto praktiky příliš podrobné anebo jejich aplikace v agilním prostředí není v souladu s agilními praktikami.

V plánu hodnocení je třeba definovat:

- Cíle/účel hodnocení, který by měl být určen v souladu s cíli podniku
- Určení cílové úrovně, proti které se bude hodnotit
- Požadavky na zdroje – např. spolupráce zaměstnanců
- Jaké budou výstupy hodnocení

Šablony hodnotících dokumentů a postup hodnocení

Pro hodnocení jsou stěžejní dvě šablony (šablony jsou součástí návrhu od Tomáše Doška [7]). První je *šablona hodnocení* (viz Obrázek 6), do které se budou postupně zaznamenávat výsledky hodnocení. Výsledky prezentují míru splnění povinných komponent modelu TMMi. Těmito komponenty jsou:

- Procesní oblasti (PA)
- Specifické cíle (SG)

Šablona vyobrazuje strukturu hodnocení. Obsahuje všechny komponenty, které je potřeba splnit pro dosažení cílené úrovně. Každá procesní oblast obsahuje specifické cíle. Hodnocení celé procesní oblasti udává nejnižší přidělené hodnocení jeho specifických cílů.

Komponenty TMMi				Hodnocení		Identifikované nedostatky
PA1	Test Policy and Strategy					
	SG1	Establish a Test Policy				
		SP1.1	Define test goals			
		SP1.2	Define test policy			
		SP1.3	Distribute the test policy to stakeholders			
	SG2	Establish a Test Strategy				
		SP2.1	Perform a generic product risk assessment			
		SP2.2	Define test strategy			
		SP2.3	Distribute the test strategy to stakeholders			
	SG3	Establish Test Performance Indicators				
		SP3.1	Define test performance indicators			
		SP3.2	Deploy test performance indicators			
	.	.	.			
	.	.	.			
	.	.	.			
	GG2	Institutionalize a Managed Process				
		GP2.1	...			
		GP2.2	...			

Obrázek 6 - Šablona hodnocení [7]

Popis jednotlivých komponent modelu TMMi, kterých je třeba pro hodnocení, je obsažen v dokumentu *Test Maturity Model Integration (TMMi)* [3]. Je však nutné vzít na zřetel, že hodnocená společnost se řídí čistě agilními principy. Část specifických praktik (SP) modelu TMMi nejsou v agilním vývoji aplikovatelné, nebo musejí být pro agilní prostředí přizpůsobeny. TMMi Foundation poskytuje dokument, který se přímo specializuje na využití modelu v agilním prostředí s názvem *TMMi in the Agile world* [5]. Dokument obsahuje výpis aplikovatelných a neaplikovatelných komponent. Dále popisuje komponenty jednotlivých procesních oblastí (PA) pro aplikaci v agilním prostředí s cílem dosáhnout druhé úrovně zralosti procesů testování. Pro hodnocení budou tedy stěžejní tyto dva dokumenty.

Pro vyjádření míry splnění jednotlivých komponent je použita stupnice hodnocení. Dokument TAMAR takovouto stupnici poskytuje a je využita i v návrhu Tomáše Doška (viz Tabulka 4).

Hodnocení	Popis	Podmínky
F	Plně dosaženo (Fully Achieved)	Plně v souladu s doporučeným plnění cíle
P	Částečně dosaženo (Partially Achieved)	Pouze částečně v souladu s doporučeným plnění cíle
N	Nedosaženo (Not Achieved)	Zcela nesplňuje doporučené plnění cíle
NR	Nehodnoceno (Not Rated)	Nebylo možné vyhodnotit
NA	Neaplikovatelné (Not Applicable)	Neaplikovatelné pro danou organizaci

Tabulka 4 - Stupnice hodnocení TMMi [7]

Rozhodnutí o udělení známce konkrétním komponentám modelu bude diskutováno se zaměstnanci společnosti. Po odsouhlasení udělení známky projektovým manažerem bude známka podle příslušného stupně plnění zaznamenána. Znamka NR se používá v případě, že z nějakého důvodu není možné rozhodnout o hodnocení konkrétní komponenty (např. nedostatek zdrojů). Při udělení známky NR je komponenta brána za nesplněnou. Další možnou známkou je NA (neaplikovatelné). Díky této známce se komponenta nezapočítává do výsledného hodnocení. Plné dosažení komponenty je značeno písmenem F, značí plné splnění praxe dle referenčního dokumentu TMMi nebo odpovídající alternativy. [2]

Samotné porovnání současného stavu s referenčním modelem musí být zdokumentováno. Porovnání se provádí na úrovni specifických cílů. K zdokumentování bude sloužit druhá šablona *tabulky hodnocení* (viz Tabulka 5).

Identifikátor cíle	
Název cíle	
Hodnocení cíle	
Aplikace cíle v agilním prostředí:	
<i>Reference:</i>	
Popis aktuálního stavu v hodnocené společnosti:	
Popis nedostatků:	

Tabulka 5 – Šablona tabulky hodnocení [7]

Udělené hodnocení musí být jasné odůvodněné. Z toho důvodu se do *tabulky hodnocení* vyplňuje popis aktuálního stavu procesu v podniku a zdroje informací využité pro porovnání. Pro každou komponentu je za pomoci vhodných zdrojů uvedena její doporučená aplikace v agilním prostředí. Důležitou informací pro návrh změn představuje popis nedostatků, který informuje o nesplněných požadavcích pro dosažení hodnocení F (plně dosaženo) u konkrétní komponenty.

Vyhodnocení dosažené úrovně

Po vyplnění *tabulek hodnocení* (viz Tabulka 5) pro specifické cíle se udělená známka zaznamená do *šablony hodnocení* (viz Obrázek 6). Jak už bylo uvedeno výše, záznam známky se vyplňuje od nejnižší postavených komponent. Nejprve se vyplní známka u specifických cílů. Těm je nadřazena procesní oblast, která přebírá nejnižší dosažené hodnocení cílů, které pod ní patří. Aby bylo kýžené úrovně zralosti dosaženo, musí být všechny její procesní oblasti ohodnoceny známkou F. [7]

6. Hodnocení současného stavu procesů testování

Tato kapitola obsahuje hodnocení procesů testování společnosti Glogster, a.s. s využitím neformální hodnotící metody pro model TMMi. Všechny náležitosti hodnotící metody jsou popsány v předchozí kapitole. Obsah této kapitoly tvoří plán hodnocení, dokumentaci porovnání současného stavu proti druhé úrovni modelu a vyhodnocení dosažené úrovně. Z dokumentace porovnání současného stavu a vyhodnocení dosažené úrovně lze získat zjištěné nedostatky, které jsou předmětem pro vypracování návrhu na zlepšení a zavedení procesů do takové podoby, aby bylo možné dosáhnout cílené úrovně.

6.1. Plán hodnocení

Před samotným hodnocením je třeba vyplnit základní informace do plánu hodnocení. Ten má za úkol nést informace o účelu a základní charakteristice způsobu hodnocení procesů testování dle modelu TMMi. Plán hodnocení je reprezentován následujícími informacemi.

Cíle/účel hodnocení, který by měl být určen v souladu s cíli podniku:

Společnost si klade za cíl optimalizovat a zvýšit efektivitu svých procesů testování s vidinou zvýšení kvality výsledných produktů, snížení nákladů na vývoj a upevnění pozice na konkurenčním trhu. Toho chce dosáhnout využitím modelu na zlepšování procesů testování (TMMi). Hodnocení poskytne společnosti jasnou informaci o současném stavu těchto procesů a umožní vytvořit návrh k odstranění zjištěných nedostatků. Po eliminaci těchto nedostatků bude společnost schopna dosáhnout minimálně druhé úrovně zralosti modelu TMMi.

Určení cílové úrovně, proti které se bude hodnotit:

Hodnocená/cílová úroveň modelu TMMi: 2. úroveň – řízená (managed)

Požadavky na zdroje:

Hlavní zdroj informací pro potřeby hodnocení je spolupráce s vývojovým týmem a autorův přehled o zavedených procesech v hodnocené společnosti. Další zdroje informací představují nástroje a dokumenty, které jsou využívány vývojovým týmem pro podporu procesů testování (GitLab, Jenkins, Interní podpůrné dokumenty).

Výstupy hodnocení:

1. Porovnání současného stavu proti druhé úrovni (viz Kapitola 6.2)
2. Vyhodnocení dosažené úrovně (viz Kapitola 6.3)

6.2. Porovnání současného stavu proti druhé úrovni TMMi

Druhá úroveň obsahuje pět procesních oblastí. Každá z těchto oblastí má své specifické cíle. Proti těmto cílům je provedeno porovnání se současnou situací ve společnosti. Hodnocení je prezentováno hodnotícími tabulkami. Každá tabulka se vždy vztahuje k jednomu konkrétnímu cíli, který spadá do jedné z pěti procesních oblastí. Aby nedošlo ke špatné interpretaci názvů jednotlivých komponent modelu, jsou tyto názvy uvedeny v originálním jazyce. Každá tabulka obsahuje informace o tom, jak by se měl daný cíl aplikovat v agilním prostředí. Tyto informace slouží jako přehledný rámec, podle kterého lze hodnotit současný stav procesů v hodnocené společnosti. Rámec je tvořen agilními praktikami, které jsou získávány syntézou informací z několika vhodných zdrojů. Těmito zdroji jsou *TMMi in the Agile world* [5], *Test Maturity Model integration (TMMi)* [3], *Agile Testing: A Practical Guide for Testers and Agile Teams* [6]. Pro ukotvení do příslušných etap agilního vývoje, bylo také využíváno zdroje *Towards Agile Implementation of Test Maturity Model Integration (TMMi) Level 2 using Scrum Practices* [12].

6.2.1. PA: Test Policy and Strategy

Identifikátor cíle	PA2.1.SG1
Název cíle	Establish a Test Policy
Hodnocení cíle	P
Rozdíly v interpretaci tohoto cíle mezi použitými zdroji:	
• <i>Agile Testing: A Practical Guide for Testers and Agile Teams</i> [6] testovací politiku vůbec nezmiňuje. Dalo by se to vysvětlit tím, že položky testovací politiky mohou být zahrnuty v dokumentu testovací strategie. • <i>TMMi in the Agile world</i> [5] však dokument testovací politiky vyžaduje. Počítá však s velmi stručnou formou a nevyžaduje vedení testovací politiky v samostatném dokumentu. Uvádí, že testovací politika je nezbytná pro dosažení společného pohledu na testování a jeho cíle mezi všemi zúčastněnými stranami v rámci organizace. Na základě doporučených vlastností testovací politiky byl vytvořen následující způsob aplikace cíle v agilním prostředí, který je vhodný pro hodnocenou společnost.	

Aplikace cíle v agilním prostředí:

Testovací politika bývá většinou jednostránkový dokument na organizační úrovni, který má za cíl seznámit všechny zúčastněné strany, v rámci organizace, s testovací filozofií společnosti. V dokumentu jsou definovány cíle testování, které jsou v souladu s podnikovými cíli (cíle testování by měly být měřitelné), je zde nastíněna metodika testování, kdo je za testování zodpovědný apod. V agilních metodikách může testovací politika sloužit jako úvod pro testovací strategii. Dále objasňuje zúčastněným stranám (většinou na organizační úrovni), co je to testování, proč je v organizaci zavedené, čeho se jím chce dosáhnout a jakým způsobem. [5]

Reference:

- *TMMi in the Agile world* [5]

Popis aktuálního stavu v hodnocené společnosti:

Testovací politika ve společnosti Glogster nemá žádnou formálně zavedenou podobu. Je brána za samozřejmou součást oblasti testování a know-how společnosti.

Popis nedostatků:

- Neexistence dokumentu politika testování. Stručně objasněná politika testování může dobře sloužit jako ucelený úvod k testovací strategii. I na vyšších úrovních organizace by mohly mít zúčastněné strany k dispozici dokument s ucelenými informacemi o oblasti testování v kontextu společnosti.

Tabulka 6 - Hodnocení – PA2.1.SG1 - Establish a Test Policy

Identifikátor cíle	PA2.1.SG2
Název cíle	Establish a Test Strategy
Hodnocení cíle	P

Rozdíly v interpretaci tohoto cíle mezi použitými zdroji:

- *Agile Testing: A Practical Guide for Testers and Agile Teams* [6] tvrdí, že v agilním testování tento dokument není nutný, ale nezavrhuje jeho vytvoření. Uvádí, že tento dokument může sloužit především novým zaměstnancům pro porozumění zavedeným procesům testování. Takový dokument by měl obsahovat informace o testovacích praktikách, typech prováděných testů a využívaných nástrojích.
- *TMMi in the Agile world* [5] vytvoření tohoto dokumentu vyžaduje. Uvádí, že dodržování vhodně vypracované testovací strategie vede k menšímu překrývání mezi různými testovacími činnostmi, což zvyšuje efektivitu procesu testování. V

náležitostech, které by měly být v dokumentu uvedeny, se oba uvedené zdroje shodují.

Aplikace cíle v agilním prostředí:

Strategie testování představuje statický dokument, který se nemusí často aktualizovat. Stanovuje standardy testování ve společnosti a celkový přístup k testování. Účelem dokumentu je zajištění toho, že všichni účastníci testování, a hlavně noví zaměstnanci, získají celkový přehled o procesech testování v organizaci. Obsahuje např. informace o způsobu použití testovacích technik (jakým způsobem jsou prováděny jednotlivé typy testů), jakým způsobem mají být zadokumentovány nalezené chyby, jaké jsou při testování využívány nástroje apod. V agilním prostředí může být strategie testování zahrnuta v jednom dokumentu spolu s testovací politikou a testovacím plánem. [5, 6]

Reference:

- *TMMi in the Agile world* [5]
- *Agile Testing: A Practical Guide for Testers and Agile Teams* [6]

Popis aktuálního stavu v hodnocené společnosti:

Ve společnosti není tento dokument zaveden. Seznámení s testovací strategií bývá prováděno ústní formou a každý zaměstnanec se musí se zavedenou strategií testování sžít a nést v paměti.

Popis nedostatků:

- Neexistence dokumentu strategie testování. Řešeno pouze ústní formou. Není v rozporu s agilním přístupem, ale zavedení dokumentu je doporučováno. Ušetří čas a usnadní předání informací ohledně strategie testování především novým zaměstnancům.

Tabulka 7 - Hodnocení – PA2.1.SG2 - Establish a Test Strategy

Identifikátor cíle	PA2.1.SG3		
Název cíle	Establish Indicators	Test	Performance
Hodnocení cíle	F		
Rozdíly v interpretaci tohoto cíle mezi použitými zdroji:			
TMMi in the Agile world [5] uvádí, že by měly být ve společnosti zavedeny některé základní ukazatele, kterými je možné měřit postup procesů testování v průběhu času. Také ale upozorňuje, že v agilním prostředí se tyto ukazatele zaměřují spíše			

na tým a celý vývoj než pouze na procesy testování, což je však s TMMi v souladu. Dále doporučuje některé příklady toho, co by se v agilním vývoji mělo měřit: uniklé defekty, rychlost, hodnocení spokojenosti zákazníků, procento automatizace testů.

- *Agile Testing: A Practical Guide for Testers and Agile Teams* [6] detailně referuje o možnostech měření konkrétních skutečností při agilním testování. Zabývá se např. měřením počtu prošlých testů, pokrytí kódu testy, měření počtu defektů apod. Na základě diskuzí s vývojovým týmem byly zvoleny ukazatele, které jsou pro společnost Glogster vhodné viz následující text.

Aplikace cíle v agilním prostředí:

Jako je tomu u všech řízených procesů, tak i u procesu testování je třeba nějakým způsobem měřit jeho pokrok. V tradičních metodikách je běžné měřit prakticky vše pomocí rozsáhlého systému metrik a měřících nástrojů - měří se např. i pracovní výkon každého zaměstnance zvlášť. Oproti tomu v agilním prostředí se ukazatele zaměřují na systém vývoje a tým jako celek. Sleduje se pouze jen to nejnnutnější. Ukazatele by měly především sloužit k motivování týmu se vyvíjet, upozorňovat na nedostatky a směřovat vývoj k dosažení požadovaných cílů. Sada ukazatelů by měla být definována na základě testovacích cílů, které jsou obsaženy v testovací politice. [5, 6]

V agilním prostředí je doporučováno a zároveň dává smysl měřit např. tyto skutečnosti:

1. V rámci časově ukotvených iterací je vhodné měřit plnění stanovených úkolů pro daný sprint. Toto měření slouží k udržení tempa, efektivity práce a k přesnějším časovým odhadům, kterých je využíváno v plánování dalších sprintů. Nejčastěji se používá tzv. Burndown Chart. [5, 6]
2. Spokojenost zákazníků s produktem (např. dotazník, dotazy a stížnosti získané od podpory zákazníků, recenze apod.) – Zpětná vazba od zákazníků bývá nejlepší motivací. Dají se tímto také měřit chyby, které se dostaly až na produkci – to může sloužit jako podnět k pečlivějšímu testování nebo k rozhodnutí o zahrnutí problematické oblasti do automatizovaných testů. [5]
3. Kolik procent napsaného kódu je pokryto testy (Unit Testy). Slouží jako motivace programátorů k psaní testů. Pomáhá odhalit části kódu vyžadujících refaktorování nebo pokrytí testy. Pro měření existují opensource nástroje. [5, 6]
4. Sledování počtu provedených automatizovaných testů a jejich výsledného stavu. Slouží jako primární ukazatel úspěšnosti buildu. Udržení výsledků testů v zelených hodnotách je v zájmu celého týmu. Automatizované testy by se měly

spouštět několikrát v průběhu sprintu, aby bylo možné na zjištěné problémy okamžitě reagovat. [5, 6]

5. Měření zaznamenaných defektů – V této oblasti je vhodné měřit: Kolik času zabere oprava nalezeného defektu. Kolik chyb se nachází na produkčním prostředí. Lze také zjistit, které chyby se často vrací a je třeba se zamyslet nad jejich příčinou a zda by bylo vhodné automatizovat testy pro lepší kontrolu takovéto oblasti. [5, 6]

Reference:

- *TMMi in the Agile world* [5]
- *Agile Testing: A Practical Guide for Testers and Agile Teams* [6]

Popis aktuálního stavu v hodnocené společnosti:

Společnost Glogster má kvalitně navržené ukazatele pro měření a splňuje všechny výše uvedené metody měření. Níže je uvedeno plnění jednotlivých měření v hodnocené společnosti, které je odvozeno od číslovaného seznamu viz výše. Jsou zde uvedeny využívané nástroje a praktiky, které tato měření umožňují.

1. Pro záznam úkolů, které je třeba vyřešit v rámci jednotlivých sprintů, je ve společnosti využíván nástroj GitLab – nástroj mimo jiné dokáže zobrazovat procentuální postup v daném sprintu (kolik procent vybraných úkolů je již splněno), nebo ho vyobrazit prostřednictvím grafu Burndown Chart.
2. Spokojenost zákazníků je primárně měřena prostřednictvím jednoduchého dotazníku prostřednictvím nástroje Google Form. Odkaz na tento formulář je umístěn na strategických místech aplikace. Výsledky jsou shromažďovány v tabulkovém dokumentu, který se nachází na cloudovém uložišti, ke kterému mají přístup všechny zainteresované strany. Ke každému záznamu je ručně přidělen povahový štítek, který umožňuje vyobrazit získané informace pomocí koláčového grafu. Graf procentuálně rozděluje záznamy podle jejich charakteru – pozitivní, negativní, důsledek chyby, podnět na novou funkci.
3. Toto měření je prováděno pomocí nástrojů PHPUnit, PHP Code Sniffer a PHP Mess Detector. Nástroje umožňují detailně vyobrazit pokrytí kódu testy, jejich kvalitu či problematické části, kterým je třeba věnovat pozornost.
4. Automatizované testy jsou ve společnosti vytvářeny prostřednictvím Codeception frameworku, který umožňuje psát testy pro všechny úrovně. Tyto testy jsou automatizovaně spouštěny při každém releasu na testovací prostředí. O to se stará automatizační nástroj Jenkins. V tomto nástroji lze získat přehledný report z výsledků testování.

5. K zaznamenávání defektů se využívá již zmíněný nástroj GitLab. Nástroj umožňuje defekty seskupovat a filtrovat podle času, priority či povahy a prostředí, na jakém se defekt nachází. Tyto vlastnosti lze přehledně monitorovat a zobrazovat pomocí seznamového výpisu nebo prezentovat prostřednictvím kanban tabulí.

Popis nedostatků:

- Nebyly nalezeny nedostatky, které by bránily dosažení tohoto cíle.

Tabulka 8 - Hodnocení – PA2.1.SG3 - Establish Test Performance Indicators

6.2.2. PA: Test Planning

Identifikátor cíle	PA2.2.SG1
Název cíle	Perform Risk Assessment
Hodnocení cíle	P

Rozdíly v interpretaci tohoto cíle mezi použitými zdroji:

- *TMMi in the Agile world* [5] doporučuje posuzování rizik na úrovni jednotlivých položek, které se vyskytují v produktovém backlogu. Riziko by se mělo posuzovat již v době jejich zadávání do produktového backlogu nebo při plánování iterace či sprintu. Na posuzování rizik by se měl podílet celý vývojový tým. Jako možnou techniku posuzování rizika uvádí Risk Poker (podrobný popis techniky se nachází např. na webu *tmap.net* [51]).
- *Agile Testing: A Practical Guide for Testers and Agile Teams* [6] naprosto souhlasí s předchozím zdrojem a pouze podrobněji doplňuje další možné techniky, které jsou popsány v následujícím textu. Také upozorňuje na to, že posuzování rizika se nezaměřuje pouze na funkcionalitu, ale i na bezpečnost, výkon, použitelnost a další aspekty.

Aplikace cíle v agilním prostředí:

Posouzení rizik představuje důležitou položku při plánování každého releasu nebo konkrétního sprintu. Hlavní důvod posuzování rizik představuje fakt, že není vždy možné vše detailně a dokonale otestovat. Je tedy třeba prioritizovat nejvíce riskantní oblasti, kterými by se měl tester primárně zabývat a věnovat jim daleko větší pozornost. V agilním prostředí se posouzení rizik běžně provádí při plánování releasu nebo sprintu, kdy jsou vytvářeny a zadávány jednotlivé položky např. user stories do produktového backlogu. Každé položce se uděluje priorita, ta většinou určuje pořadí, v jakém budou jednotlivé případy řešeny a zařazovány do jednotlivých iterací. Následně se tým musí dohodnout, jaký stupeň rizika danému případu přidělí. Stupeň rizika může být reprezentován číselnou stupnicí, kde se číselná hodnota získá součinem hodnot dopadu

(jaký dopad může mít výskyt chyby nebo úplná nefunkčnost u tohoto případu) a pravděpodobnosti výskytu (jaká je pravděpodobnost, že v tomto případě nastane problém). Nebo se určí stupně rizika pomocí jednoznačných štítků např. malé riziko, střední riziko, vysoké riziko. Na základě takto posouzených rizik tester získává cennou informaci, které musí přizpůsobit přístup k testování. Posouzení rizik může sloužit k určování testů, které je třeba automatizovat, nebo zařadit mezi regresní manuální testy. [5, 6]

Reference:

- *TMMi in the Agile world* [5]
- *Agile Testing: A Practical Guide for Testers and Agile Teams* [6]

Popis aktuálního stavu v hodnocené společnosti:

Posuzování rizik v hodnocené společnosti není jasně definováno a ukotveno. K jednotlivým položkám, které jsou obsaženy v product backlogu, se přiřazuje pouze priorita, která představuje buď akutnost vyřešení anebo obchodní hodnotu. Stanovení míry rizika je až na samotném testerovi nebo vývojáři – míra rizika se nikam nezaznamenává. Takto nastavený systém může při nesprávně odhadnuté situaci zapříčinit nedostatečné otestování rizikového případu a vydání možné chyby na produkční prostředí.

Popis nedostatků:

- Neexistence systematického posuzování rizika.
- Nutné zaznamenávat riziko ke každé položce v produktovém backlogu.

Tabulka 9 - Hodnocení – PA2.2.SG1 - Perform Risk Assessment

Identifikátor cíle	PA2.2.SG2
Název cíle	Establish a Test Approach
Hodnocení cíle	P

Rozdíly v interpretaci tohoto cíle mezi použitými zdroji:

- *TMMi in the Agile world* [5] uvádí, že testovací přístup by měl být plánován během plánování iterací a měl by se zakládat na stupních přidělených rizik. Nalezení defektu v průběhu testování může vést k opětovnému posouzení rizika a tím pádem i ke změně testovacího přístupu.
- *Agile Testing: A Practical Guide for Testers and Agile Teams* [6] souhlasí s TMMi. Uvádí např., že případy s vysokým rizikem by měly být pokryty automatizovanými testy. Oproti tomu případy s nízkým stupněm rizika se

v krajních případech nemusí testovat vůbec. Dále jen dodává, že testovací přístup na základě posouzených rizik se také odvíjí od povahy vyvíjeného softwarového produktu. Pokud se například jedná o software, jehož chybovost může ohrozit životy apod., tak nezáleží na přiděleném riziku a musí být pokryta adekvátními testy všechna rizika bez ohledu na jejich stupně.

Aplikace cíle v agilním prostředí:

Tento specifický cíl spočívá v zavedení testovacího přístupu, který se zakládá na posouzených rizicích. Zjednodušeně je možné říci, že nastavuje pravidla o způsobu a přístupu k testování jednotlivých případů a oblastí s ohledem na udělený stupeň rizika se snahou tato rizika zmírnit a testovat co nejefektivněji. Určuje se zde např. jaké použít techniky testování a na jakých úrovních daný případ testovat úměrně ke stupni rizika. Typicky se zde řeší možnosti automatizace a zařazení konkrétního případu do skupiny regresních testů. Testovací přístup se stanovuje při plánování releasu, nebo sprintu.

Testovací přístup by se měl zakládat také na kritériích pro ukončení testů (DoD – Definition of Done), které určují, kdy je daný případ bezpečně otestovaný a je připraven na release. Tyto kritéria mohou také obsahovat požadovaný testovací přístup. Typické případy, které nesplňují kritéria pro ukončení, jsou vloženy do nevyřízených případů a mohou být řešeny v rámci další iterace. [5, 6]

Reference:

- *TMMi in the Agile world* [5]
- *Agile Testing: A Practical Guide for Testers and Agile Teams* [6]

Popis aktuálního stavu v hodnocené společnosti:

Stejně jako u předchozího cíle (PA2.2.SG1) je i testovací přístup závislý čistě na rozhodnutích testera či vývojáře. Testovací přístup ke konkrétním případům je běžně řešen při plánování sprintu. V této fázi se tým rozhoduje o vhodnosti použití konkrétních technik a úrovní testů. Kritéria pro ukončení testů vyplývají z informací obsažených v kartě jednotlivých položek (např. user stories) v produktovém backlogu. Dále se spoléhá na svědomitost testera, nebo jsou vedeny diskuze se členy týmu v průběhu iterace.

Popis nedostatků:

- Hlavní nedostatek vyplývá z nedostatku předchozího cíle. Jelikož ve společnosti není zavedeno systematické posuzování rizika, tak není možné jasně definovat testovací přístup na základě konkrétního stupně těchto rizik.

Identifikátor cíle	PA2.2.SG3
Název cíle	Establish Test Estimates
Hodnocení cíle	P

Rozdíly v interpretaci tohoto cíle mezi použitými zdroji:

- *TMMi in the Agile world* [5] doporučuje zohledňovat testovací odhady na úrovni jednotlivých položek, které se nacházejí v produktovém backlogu. V případě, že testovací případy jsou vedeny samostatně, odhady se provádějí jen u testovacích případů. V agilním prostředí se však často testuje na základě vytvořených stories. V tomto případě by se odhady na testování měly jasně vymezit u každé story.
- *Agile Testing: A Practical Guide for Testers and Agile Teams* [6] problematiku odhadů prakticky neřeší. Ohledně této oblasti se jen zmiňuje, že odhady na testování jsou v agilním prostředí většinou součástí odhadů na samotný vývoj.
- *Agile Testing Estimation – TestLodge* [52] - V případě časových odhadů na testování byl využit i tento zdroj, kde je téma blíže popsáno. Autor článku doporučuje provádět odhady na testování společně s odhady na vývoj. Hlavním důvodem je to, že testování je v agilních metodikách přirozenou součástí vývoje. Navíc odhady na testování držené odděleně, mohou vést k ignoraci těchto odhadů a oddělovat testování od vývoje.

Aplikace cíle v agilním prostředí:

V agilním prostředí bývají odhady testování prováděny v rámci plánování jednotlivých sprintů/iterací a podílí se na nich celý tým. Odhaduje se převážně potřeba vynaloženého úsilí, které je zastoupeno jednotkou času potřebného na vyřešení konkrétní položky produktového backlogu. Odhady v podobě nákladů se většinou neprovádějí. Přístupů a metod k odhadování časové náročnosti je mnoho. Probíhá to například tak, že se ke každé položce, která byla vybrána z produktového backlogu a je zařazena do daného sprintu, odhaduje přibližná doba na vývoj a následně na otestování. Tyto dva časové odhady se sečtou a vyjde přibližná doba na kompletní splnění dané položky. Je pak na samotném týmu, zda dobu na testování a vývoj budou přisuzovat k dané položce zvlášť nebo jako součet. To se může odvíjet od skutečnosti, zda karty/záznamy jednotlivých položek, které jsou vedeny v produktovém backlogu, slouží zároveň jako testovací případy. Na základě odhadnuté časové náročnosti jednotlivých případů lze jejich součtem následně odhadovat i délku konkrétního sprintu či releaseu. [5, 52]

Reference:

- *TMMi in the Agile world* [5]
- *Agile Testing Estimation – TestLodge* [52]

Popis aktuálního stavu v hodnocené společnosti:

Odhady časové náročnosti jsou diskutovány buď již při zapisování položky do produktového backlogu anebo až při plánování konkrétního sprintu. Ne vždy se tento odhad zapíše k dané položce. Většinou se nad odhady vede diskuze, jejíž výsledkem bývá přibližný odhad délky daného sprintu. Jelikož se časová náročnost mnohdy řeší jen okrajově a není jasně zdokumentována, tak může docházet ke značně nepřesným odhadům.

Popis nedostatků:

- Nezavedený jednotný systém pro časové odhady jednotlivých úkolů.
- Časové odhady nejsou mnohdy zaznamenávány.

Tabulka 11 - Hodnocení – PA2.2.SG3 - Establish Test Estimates

Identifikátor cíle	PA2.2.SG4
Název cíle	Develop a Test Plan
Hodnocení cíle	P

Rozdíly v interpretaci tohoto cíle mezi použitými zdroji:

- *TMMi in the Agile world* [5] uvádí, že v agilním vývoji se většinou nevytváří žádný explicitní testovací plán. Předpokládá se, že jednotlivé stories, resp. úkoly, které mají být prováděny, již obsahují (po splnění předchozích cílů) vše potřebné pro účely testování. Plánování je zde zastoupeno diskuzemi týmu během plánování iterace. Během iteračního plánování (v případě potřeby) může být projednána identifikace zdrojů/pracovníků. Ta má za úkol zjistit, zda je zajištěn dostatečný počet pracovníků se znalostmi potřebnými k provedení konkrétních testů (např. nefunkcionální testování).
- *Agile Testing: A Practical Guide for Testers and Agile Teams* [6] do plánování testů zahrnuje souhrn aktivit, které jsou v modelu TMMi zastoupeny v jiných cílech nebo procesních oblastech. Mezi ně patří posouzení a prioritizace rizik (viz PA2.2.SG1), stanovení testovacího přístupu (viz PA2.2.SG2), časové odhady (viz PA2.2.SG3). Dále musí být ujasněno na jakém prostředí se bude testovat a zda jsou připravena testovací data. Plánování je vedeno formou diskuzí, kde se tester může doptávat na detaily.

Aplikace cíle v agilním prostředí:

Plán testů je v agilním prostředí tradičně součástí plánování sprintu či releasu a většinou nebývá dokumentován. V případě, že jsou splněny předchozí cíle, tzn.: jsou posouzena rizika (viz PA2.2.SG1), ujasněn testovací přístup (viz PA2.2.SG2), a časové odhady náročnosti (viz PA2.2.SG3), je plánování testování téměř hotové. V tuto chvíli je prostor pro dodatečné otázky a diskuzi, aby bylo jasné, že je vše pochopeno správně (počítá se zde se součinností vlastníka produktu). Také se řeší, kdo bude konkrétní případy testovat. To se rozhoduje na základě znalostí, dovedností a zdrojů, které má konkrétní člen týmu k dispozici. Důležité je také určení, na jakém testovacím prostředí se bude testovat (pokud jich je více), a jaká testovací data jsou nutná k umožnění testování (např. připravená data v databázi). [5, 6]

Reference:

- *TMMi in the Agile world* [5]
- *Agile Testing: A Practical Guide for Testers and Agile Teams* [6]

Popis aktuálního stavu v hodnocené společnosti:

V případě, že nejsou brány v potaz nedostatky předchozích cílů, na které tento cíl navazuje, je jinak tento cíl v souladu s aplikací cíle v agilním prostředí viz výše. Při každém plánování iterace se provádí otevřená diskuze, kde se domlouvají detaily řešení plánovaných položek z produktového backlogu. Řeší se např., kdo a jak bude dané položky testovat (pokud to není zřejmé). Jedná-li se např. o nefunkční testy v podobě korekce textu v cizím jazyce, je toto testování předáno pracovníkovi z oddělení péče o zákazníky, který je rodilý Američan. O tvorbu automatizovaných testů se dělí tester i hlavní vývojář (rozhoduje se na základě úrovně automatizovaného testu). Určení, na jakém testovacím prostředí se bude daná položka testovat, je prováděno jak při plánování iterace, tak především v průběhu iterace. Ve chvíli, kdy vývojář předává položku k otestování, zároveň uvádí, na jakém testovacím prostředí se daná položka nachází.

Popis nedostatků:

- Nedostatky, které brání splnění tohoto cíle, pramení z nedostatků předchozích cílů. Těmito cíli jsou: PA2.2.SG1 (viz Tabulka 9), PA2.2.SG2 (viz Tabulka 10) a PA2.2.SG3 (viz Tabulka 11). Po splnění těchto cílů bude splněn i tento cíl.

Tabulka 12 - Hodnocení – PA2.2.SG4 - Develop a Test Plan

Identifikátor cíle	PA2.2.SG5
Název cíle	Obtain commitment to the Test Plan
Hodnocení cíle	F

Rozdíly v interpretaci tohoto cíle mezi použitými zdroji:

- *Agile Testing: A Practical Guide for Testers and Agile Teams* [6] tento cíl nijak nevymezuje. Zřejmě je to dáno tím, že zavázání se k testovacímu plánu je přirozeným výsledkem plánování v agilním vývoji.
- *TMMi in the Agile world* [5] – aplikace cíle tak, jak ho definuje tento zdroj je obsažen v textu níže.

Aplikace cíle v agilním prostředí:

Za kvalitu produktu je v agilním prostředí zodpovědný tým jako celek, proto je vhodné, aby celý tým souhlasil s nastavenými pravidly a procesy. Zavázání se k plánu testů v agilním prostředí představuje zjednodušeně potvrzení celého týmu, že souhlasí s nastaveným testovacím plánem a všemi jeho náležitostmi viz PA2.2.SG4 (Tabulka 12). Tento závazek představuje výsledek dobře nastaveného procesu. Tohoto závazku je docíleno tím, že veškerého plánování se účastní celý tým. Plánování je ukončeno v okamžiku vzájemné shody. [5]

Reference:

- *TMMi in the Agile world* [5]

Popis aktuálního stavu v hodnocené společnosti:

V hodnocené společnosti je veškeré plánování, např. sprintů a releasů (zahrnující i plánování testů), podmíněno shodou mezi všemi zúčastněnými stranami. Každý má prostor vyjádřit svůj názor a přednést své argumenty. Zavázání se k plánu testů je tak přirozeným výsledkem plánování.

Popis nedostatků:

- Nebyly nalezeny nedostatky, které by bránily dosažení tohoto cíle modelu TMMi.

Tabulka 13 - Hodnocení – PA2.2.SG5 - Obtain commitment to the Test Plan

6.2.3. PA: Test Monitoring and Control

Identifikátor cíle	PA2.3.SG1
Název cíle	Monitor Test Progress Against Plan
Hodnocení cíle	F

Rozdíly v interpretaci tohoto cíle mezi použitými zdroji:

- *Agile Testing: A Practical Guide for Testers and Agile Teams* [6] je v souladu s informacemi obsaženými v následujícím zdroji. Platí zde stejné informace, které byly uvedeny v tabulce PA2.1.SG3 (viz Tabulka 8).
- *TMMi in the Agile world* [5] uvádí, že v agilním vývoji je monitorování postupu testování možné mnoha způsoby. Mezi ně patří např. Burndown Chart, tabule úkolů (např. Kanban tabule), či report automatizovaných testů. Především se však pokrok testů řeší při denních diskuzích, kde se průběžně sděluje, co již bylo provedeno, co je ještě třeba a případně kde nastal problém. Tyto praktiky TMMi povoluje a využití některých z uvedených praktik je v souladu se splněním tohoto cíle.

Aplikace cíle v agilním prostředí:

Cílem monitorování pokroku testů oproti plánu je sledování a zaznamenávání průběhu testování. Sleduje se např., co je již otestováno a připraveno na release, nebo co na otestování teprve čeká. Také se pozoruje, zda je pokrok testování v souladu s testovacím plánem. Nebo se monitoruje testovací prostředí, aby testeré měli přehled o tom, zda je připravené na testování konkrétních oblastí a funkcí. V agilním prostředí je pro tyto účely využíváno různorodé množství metod a nástrojů. Řada z nich již byla popsána v cíli PA2.1.SG3 (viz Tabulka 8). Zde jsou uvedeny pouze ty, které se přímo vážou na sledování průběhu testů a zároveň jsou vhodné pro použití v hodnocené společnosti:

- Burndown Chart – Využívá se k měření pokroku vývoje v jednotlivých iteracích oproti plánu. V grafu se promítá postupné zavírání naplánovaných úkolů v závislosti na čase. Úkoly bývají zavírány až po kompletním otestování a odsouhlasení splněného úkolu. V grafu se tedy promítá i čas vynaložený na testování. Pokud je testování dokumentováno odděleně v samostatných testovacích případech, lze ho využít i se zaměřením pouze na testování. [5, 6]
- Tabule úkolů – Tabule úkolů, většinou označována jako kanban tabule, může být tvořena prostřednictvím softwarového nástroje, nebo využitím skutečné fyzické tabule. Tabule je zpravidla rozdělena na několik sloupců (např. To do, In progress, Ready for test, Done), tyto sloupce představují stav daného úkolu, ve které se nachází. V případě testování má tester a zúčastněné strany okamžitý přehled o tom, co se má otestovat. V případě, že se některé úkoly přestanou mezi sloupci pohybovat v přípustných intervalech, může to značit problém a kolizi s plánem, kterou je třeba řešit. [5, 6]

- Ústně – V agilním prostředí je přirozené, když spolu členové týmu pravidelně komunikují. I pokrok testování může být řešen ústně např. na denních schůzích nebo během diskuzí. [5, 6]

Reference:

- *TMMi in the Agile world* [5]
- *Agile Testing: A Practical Guide for Testers and Agile Teams* [6]

Popis aktuálního stavu v hodnocené společnosti:

Ve společnosti jsou testovací a uživatelské případy evidovány dohromady. Případy představují jednotlivé položky v produktovém backlogu, které jsou vytvářeny v nástroji GitLab. Každá z těchto položek během svého životního cyklu cestuje mezi konkrétními stavy (To do, In progress, Ready for test, Ready for deploy, Closed). Položky se mohou podle jednotlivých stavů filtrovat. Při zobrazení položek ve stavu „Ready for test“ (k otestování) je možné přehledně vidět jednotlivé položky k otestování. Ze vzniklého výpisu lze zjistit, jak dlouho v tomto stavu jsou a v komentářích se dozvědět dodatečné informace (např.: nelze otestovat – testovací prostředí je mimo provoz). GitLab umožňuje zobrazit položky i prostřednictvím kanban tabule (viz Příloha A: Kanban tabule využívaná společností Glogster), kde je možné pokrok testování taktéž sledovat (sledováním přesouvaných úkolů napříč sloupci tabule). Tato metoda se zároveň doplňuje ústní formou na denních schůzích nebo běžných diskuzích, kde je řešen stav testování a případné problémy blokující další postup.

Popis nedostatků:

- Nebyly nalezeny nedostatky, které by bránily dosažení tohoto cíle modelu TMMi.

Tabulka 14 - Hodnocení – PA2.3.SG1 - Monitor Test Progress Against Plan

Identifikátor cíle	PA2.3.SG2
Název cíle	Monitor Product Quality Against Plan and Expectations
Hodnocení cíle	F
Rozdíly v interpretaci tohoto cíle mezi použitými zdroji:	
• <i>Agile Testing: A Practical Guide for Testers and Agile Teams</i> [6] – v knize není sledování kvality produktu přímo definováno. Z textu však vyplývá, že kvalitu produktu posuzuje především zákazník. Ten kvalitu posuzuje zejména mírou splnění akceptačních kritérií, které jsou ověřovány akceptačními testy.	

- *TMMi in the Agile world* [5] se odkazuje na předchozí cíl PA2.3.SG1 (viz Tabulka 14) a dodává, že pro monitorování kvality produktu se využívají stejné mechanismy jako v předchozím cíli. Dále popisuje, jaké další techniky se využívají přímo pro sledování kvality.

Aplikace cíle v agilním prostředí:

K průběžnému monitorování kvality produktu jsou v agilním prostředí běžně využívány tyto metody.

1. V agilním týmu je zákazník jeho právoplatným členem. Na jeho straně jsou prováděny akceptační testy, které ověřují stanovená akceptační kritéria. Výsledky těchto testů slouží pro průběžné sledování kvality produktu. [6]
2. Monitorují se také skutečnosti týkající se nalezených defektů/chyb, jako např. četnost nalezených defektů, rychlost vyřešení defektů, dále počet defektů opravených v aktuálním sprintu a defektů, které je třeba zařadit do řešení v následujících iteracích. [5, 6]
3. Monitorování spokojenosti zákazníků prostřednictvím dotazníků, uživatelských recenzí. Měří se fluktuace uživatelů (počet nových uživatelů oproti ztraceným uživatelům). [5]
4. Posuzování kvality je také předmětem diskuzí při denních schůzích. Zúčastněné strany na konci iterace posuzují, zda jsou dosažena DoD kritéria. [5]

Reference:

- *TMMi in the Agile world* [5]
- *Agile Testing: A Practical Guide for Testers and Agile Teams* [6]

Popis aktuálního stavu v hodnocené společnosti:

Společnost splňuje všechny body, jež jsou uvedeny výše. V následujícím textu je uvedeno plnění jednotlivých měření v hodnocené společnosti, které je odvozeno od číslovaného seznamu aplikace cíle v agilním prostředí.

1. Hodnocená společnost vyvíjí vlastní službu, kterou sama provozuje a poskytuje až cílovým uživatelům. Společnost (zastoupena především vývojovým týmem) tedy sama sobě určuje akceptační kritéria a řídí celý vývoj. Jsou zde také prováděny akceptační testy (manuální i automatizované), ke kterým se přistupuje z uživatelského pohledu. Pokud jsou testy zakončeny pozitivním výsledkem (byla splněna akceptační kritéria), předpokládá se, že produkt je natolik kvalitní, aby mohl být vypuštěn mezi cílové uživatele.

2. Velká četnost nalezených defektů v konkrétní oblasti softwarového produktu může často vést ke změnám požadavků na danou funkcionalitu. Pokud jde naopak o defekty, které se v dané oblasti často vracejí, tak se tato náchylná oblast pokrývá automatizovanými testy. Tímto je pak zajištěna kvalita sw produktu, protože automatizované testy odhalí chybu včas a nestane se tak, že by se chyba mohla ocitnout až na produkčním prostředí.
3. Spokojenost zákazníků je primárně měřena prostřednictvím jednoduchého dotazníku, který je vytvořen v nástroji Google Form. Odkaz na tento formulář je umístěn na strategických místech aplikace. Výsledky jsou shromažďovány v tabulkovém dokumentu, který se nachází na cloudovém uložišti a mají k němu přístup všechny zainteresované strany. Ke každému záznamu je ručně přidělen povahový štítek, který umožňuje vyobrazit získané informace pomocí koláčového grafu. Graf procentuálně rozděluje záznamy podle jejich charakteru – pozitivní, negativní, důsledek chyby, podnět na novou funkci.
4. Kvalita je řešena během průběžných diskuzí, kdy tým hodnotí, zda produkt splňuje všechny očekávané vlastnosti (použitelnost, bezpečnost, funkčnost, spolehlivost apod.). Diskuze na toto téma probíhají přibližně dvakrát za měsíc, nebo když je diskuze vyvolána na popud např. negativní zpětné vazby od uživatelů.

Popis nedostatků:

- Nebyly nalezeny nedostatky, které by bránily dosažení tohoto cíle modelu TMMi.

Tabulka 15 - Hodnocení – PA2.3.SG2 - Monitor Product Quality Against Plan and Expectations

Identifikátor cíle	PA2.3.SG3
Název cíle	Manage Corrective Actions to Closure
Hodnocení cíle	F

Rozdíly v interpretaci tohoto cíle mezi použitými zdroji:

- *Agile Testing: A Practical Guide for Testers and Agile Teams* [6] klade důraz na rychlou komunikaci. V případě, že dojde k odhalení problému, který brání dalšímu postupu projektu, měl by se okamžitě diskutovat a nalézt vhodné opravné opatření.
- *TMMi in the Agile world* [5] předchozí názor podporuje a poskytuje podrobnější informace k naplnění cíle.

Aplikace cíle v agilním prostředí:

Flexibilita agilních týmů umožňuje reagovat na případné problémy velmi rychle. Dané problémy jsou zjišťovány např. formou odchylky od očekávaného plánu (viditelné např. v Burndown grafu). Také může docházet k nedostatečnému postupu úkolů napříč tabulí (např. Kanban tabulí), či problémům blokujících průběh testování. Zjištěné problémy a nedostatky jsou většinou předmětem okamžitých diskuzí nebo denních schůzek. Díky tomu je celý tým o vzniklých skutečnostech dokonale informován a může na jejich nápravě bezodkladně pracovat. O dané bezodkladnosti jsou samozřejmě také vedeny diskuze. Vzniklý problém se musí důkladně zanalyzovat a teprve na základě jejich povahy se tým rozhodne, zda je nutné ho řešit okamžitě, nebo se může přeložit k řešení v další iteraci. [5, 6]

Příčinou tzv. „zamrznutí“ úkolu během iterace může být i jeho neproveditelnost – nedostatečné technologie, DoD jsou nastaveny moc přísně – velká časová náročnost. Tým v této situaci musí řešit nápravné opatření společně s vlastníkem produktu. Výsledkem řešení může být: nalezení schůdnějšího řešení, vlastník ustoupí od některých přísně nastavených DoD, úkol se přesune do dalších sprintů, nebo může dokonce dojít k úplnému smazání daného úkolu. Tyto změny musí být detailně dokumentovány (např. v kartě konkrétního úkolu nebo v záznamu z denní schůzky) a tým o nich musí mít dokonalý přehled. [5]

Reference:

- *TMMi in the Agile world* [5]
- *Agile Testing: A Practical Guide for Testers and Agile Teams* [6]

Popis aktuálního stavu v hodnocené společnosti:

Společnost je plně v souladu s uvedenou metodou pro správu nápravných opatření. Vzniklé problémy jsou okamžitě diskutovány se zúčastněnými stranami a nápravná opatření zaznamenávána do karet problémových úkolů formou komentářů. Problémové úkoly jsou identifikovány např. při jejich dlouhém setrvávání ve stavu „in progress“ nebo jeho častým přeskokováním mezi stavy „in progress“ (probíhající) a „ready for test“ (připraveno k otestování).

Popis nedostatků:

- Nebyly nalezeny nedostatky, které by bránily dosažení tohoto cíle modelu TMMi.

Tabulka 16 - Hodnocení – PA2.3.SG3 - Manage Corrective Actions to Closure

6.2.4. PA: Test Design and Execution

Identifikátor cíle	PA2.4.SG1
Název cíle	Perform Test Analysis and Design using Test Design Techniques
Hodnocení cíle	P

Rozdíly v interpretaci tohoto cíle mezi použitými zdroji:

- Mezi zdroji *Agile Testing: A Practical Guide for Testers and Agile Teams* [6] a *TMMi in the Agile world* [5] nebyly nalezeny žádné zásadní rozdíly.

Aplikace cíle v agilním prostředí:

V agilním prostředí jsou testéři součástí vývojového týmu. O analýzu a návrh testů se starají již ve fázi, kdy jsou vytvářeny uživatelské případy a požadavky na softwarový produkt, na jejichž tvorbě se testéři často podílejí. U agilního testování není zcela běžné vytváření podrobně popsaných testovacích případů. Vhodně popsané use cases nebo user stories mohou sloužit zároveň jako testovací případy. [5] Hojně je využíváno tzv. průzkumného testování (exploratory testing), kdy tester není svázán zadanými testovacími případy, ale sám postupně přichází na vhodné techniky a způsoby testování dané oblasti/požadavku, ve kterých se zároveň zdokonaluje. Průzkumné testování slouží také pro testování softwaru v širším kontextu, kde se tester nemusí soustředit pouze na konkrétní funkci/požadavek, ale zaměří se rovnou na celou oblast. Doménu agilního testování představuje automatizace. Automatizovanými testy by měly být pokryty především rizikové oblasti a regresní testy. Při návrhu testů v oblasti automatizace lze také využít přístup TDD (Vývoj řízený testy – před psaním kódu se první vytvoří testy, následné procházení testů značí pokrok vývoje) a pro testy na vyšších úrovních BDD (Vývoj řízený chováním – obdoba TDD určena pro automatizované testy na vyšších úrovních). [5, 6]

Pořadí provádění testů je určováno prostřednictvím udělených priorit u každé položky produktového backlogu. Priorita určuje obchodní hodnotu. Musí se však brát ohled i na přidělený stupeň rizika, který značí míru možného negativního dopadu. Záleží na týmem domluvených pravidlech, které určují způsob přístupu k uděleným prioritám a mírám rizika. V agilním prostředí se priority jednotlivých úkolů i jejich rizika, tím pádem i pořadí testů, mohou měnit v průběhu iterace. [5, 6]

Analýza a návrh testů zahrnuje i testovací data, která jsou potřeba pro provádění testů. V agilním vývoji jsou testovací data nutná především pro správný běh automatizovaných

testů. V případě manuálních testů bývají testovací data v mnoha případech vytvářena během samotného testování. [5, 6]

Mezi testovacími požadavky a testy musí být zaručena trasovatelnost. To znamená, že mezi požadavky a testovacími případy (pokud jsou vedeny zvlášť), které testují daný požadavek, musí být dohledatelné spojení určující míru pokrytí požadavků testy. Toto spojení bývá zajištěno pomocí navazujících odkazů v příslušném nástroji nebo propojenými identifikátory. [5]

Reference:

- *TMMi in the Agile world* [5]
- *Agile Testing: A Practical Guide for Testers and Agile Teams* [6]

Popis aktuálního stavu v hodnocené společnosti:

V hodnocené společnosti se tester ve velké míře podílí na vytváření požadavků. Ty se zadávají ve formě user stories, nebo use cases do produktového zásobníku a obsahují i informace potřebné k jejich otestování (na jakém testovacím prostředí testovat, na co se při testování zaměřit, oblast testování, jaký je očekávaný výsledek testované položky – DoD atd.). Do detailu konkrétní položky produktového backlogu se často vkládá i grafický návrh s detailně popsány vlastnostmi a kroky, které určují, jak má daná funkcionality detailně vypadat a fungovat. Proto zde nejsou odděleně vytvářeny podrobné testovací případy a scénáře. Detailně popsané položky produktového backlogu zde slouží zároveň jako testovací případy. Úroveň detailu popisu jednotlivé položky záleží na její složitosti. V mnoha případech je požadavek/položka popsána jen jako krátká user story a tester na její otestování využívá technik průzkumného testování. Automatizované testy jsou přímo vytvářeny na základě diskuzí, při kterých je rozhodováno o případech vhodných k automatizaci.

Pořadí testování určují přiřazené stupně priority a výsledky diskuze při plánování sprintu/releasu, nebo samotný průběh vývoje v rámci iterace. Během plánování se mimo jiné diskutuje posouzení rizik. Řeší se mj. otázky, zda přednostně otestovat rizikovou oblast nebo oblast s větší prioritou. Rizika se však nijak formálně nezaznamenávají.

Testovací data se vytvářejí na základě aktuální potřeby. Většinou jsou vytvářeny při samotném testování. Data pro automatizované testy jsou buď automaticky generována při průběhu testů, nebo ručně vytvářena během jejich vytváření.

Trasovatelnost mezi požadavky a testy nemusí být řešena, protože požadavky jsou zaznamenávány ve formě user stories nebo use cases a slouží zároveň jako testovací případy.

Popis nedostatků:

- Jednotlivým položkám by měla být přiřazena míra rizika, jako u každého zakládaného úkolu/požadavku (vyplývá z nalezeného nedostatku PA2.2.SG1).

Tabulka 17 - Hodnocení – PA2.4.SG1 - Perform Test Analysis and Design using Test Design Techniques

Identifikátor cíle	PA2.4.SG2
Název cíle	Perform Test Implementation
Hodnocení cíle	F
Rozdíly v interpretaci tohoto cíle mezi použitými zdroji: • Mezi zdroji <i>Agile Testing: A Practical Guide for Testers and Agile Teams</i> [6] a <i>TMMi in the Agile world</i> [5] nebyly nalezeny žádné zásadní rozdíly. Aplikace cíle v agilním prostředí: Implementace testů představuje poslední fázi před samotným prováděním testů. Kontroluje se, zda je vše pro tuto aktivitu připraveno. Dochází zde také k implementaci automatizovaných testů, jejíž výsledkem jsou testy připravené k prvnímu spuštění. V agilním prostředí je tato fáze výsledkem předchozí fáze, která se týká analýzy a návrhu testů. [5, 6] Tato fáze byla popsána v předchozím cíli (PA2.4.SG1). Předpokládá se tedy, že při splnění předchozího cíle nestojí nic v cestě k současnému splnění i tohoto cíle. [5] <i>Reference:</i> • <i>TMMi in the Agile world</i> [5] • <i>Agile Testing: A Practical Guide for Testers and Agile Teams</i> [6] Popis aktuálního stavu v hodnocené společnosti: Po analýze a návrhu jsou ve společnosti splněny všechny předpoklady k provedení testů. Některé skutečnosti, jako např. návrh a vytváření automatizovaných testů, často probíhají až v průběhu iterace. Tato skutečnost je však v souladu s flexibilitou agilního přístupu a nebrání ve splnění tohoto cíle. Popis nedostatků: • Nebyly nalezeny nedostatky, které by bránily dosažení tohoto cíle modelu TMMi.	

Tabulka 18 - Hodnocení – PA2.4.SG2 - Perform Test Implementation

Identifikátor cíle	PA2.4.SG3
Název cíle	Perform Test Execution
Hodnocení cíle	P

Rozdíly v interpretaci tohoto cíle mezi použitými zdroji:

- Mezi zdroji *Agile Testing: A Practical Guide for Testers and Agile Teams* [6] a *TMMi in the Agile world* [5] nebyly nalezeny žádné zásadní rozdíly. Kniha *Agile Testing: A Practical Guide for Testers and Agile Teams* [6] pouze obsahuje více detailních informací.

Aplikace cíle v agilním prostředí:

V agilním prostředí obecně platí, že testování není prováděno odděleně od vývoje, ale představuje součást vývojového procesu. Testovací aktivity jsou prováděny průběžně během vývoje při vzájemné spolupráci celého týmu (hlavně mezi testery a vývojáři). V malých týmech se většinou netestuje podle podrobně vytvořených testovacích případů – např. vhodně popsané user stories, nebo use cases mohou být dostačující. Testy se provádí v souladu s prioritami, jež byly definovány během iteračního plánování. V iterativním vývoji často vzniká potřeba regresního testování. To je možné provádět manuálně, ale větší důraz se klade na jejich automatizaci. Automatizované testy je možné průběžně spouštět, díky čemuž vývojáři získávají okamžitou zpětnou vazbu. [5, 6]

V případě nalezení defektu během testování se nalezený problém nezaznamenává okamžitě. V agilních projektech je zvykem, že se nejprve daný problém diskutuje v rámci týmu. Tým se může rozhodnout, zda daný problém řešit okamžitě bez nutnosti jeho zaznamenání, nebo je třeba daný problém zaznamenat. Důvodem pro zaznamenání defektu může být jeho složitá povaha nebo odložení jeho řešení na později. Pro zaznamenávání defektů většinou slouží vhodné nástroje (Jira, Redmine, HP ALM, Bugzilla, GitLab, tabule úkolů atd.) ty většinou umožňují problém dokonale popsat, přiřadit jim adekvátní podrobnosti, případně mapovat defekt na konkrétní požadavek a mnoho dalšího. Je důležité k záznamu také přiřadit, na jakém prostředí se defekt vyskytuje – je možné, že se jedná o defekt, který unikl na produkční prostředí, nebo se ve společnosti vyskytuje více typů testovacího prostředí. Postup testování se průběžně zaznamenává z důvodu sledování jeho pokroku. [5, 6]

Reference:

- *TMMi in the Agile world* [5]
- *Agile Testing: A Practical Guide for Testers and Agile Teams* [6]

Popis aktuálního stavu v hodnocené společnosti:

Společnost Glogster je, mimo jeden nedostatek, v souladu s praktikami uvedenými výše. Testy jsou prováděny na základě jednotlivých položek z produktového backlogu, které vývojář označí jako „Ready For Test“ (připraveno k otestování). To je znamení pro testera, které značí, že daná položka byla nahrána na testovací prostředí (okamžitě po nahrání na testovací prostředí jsou spuštěny automatizované testy) a je připravena k manuálnímu otestování. V tuto chvíli začne tester, na základě povahy položky, detailu popisu a dalších vlastností dané položky, s jejím testováním.

V případě nalezení defektu během testování se nalezený problém nezaznamenává okamžitě. Nejprve se daný problém diskutuje v rámci týmu. Tým se může rozhodnout, zda daný problém řešit okamžitě bez nutnosti jeho zaznamenání, nebo je vytvoření záznamu nutné. Pro zaznamenávání defektů slouží nástroj Gitlab. Defekty se v Gitlabu nacházejí na stejné úrovni jako ostatní úkoly, které jsou řešeny v jednotlivých iteracích. Pro jejich odlišení je karta označována štítky s názvem „Bug“, podle nich se defekty v nástroji dají oddělit od ostatních položek. Ke kartám se dále přiřazují tyto podrobnosti: prostředí, ve kterém se defekt vyskytuje, priorita (určuje závažnost defektu), operační systém a typ prohlížeče (pokud se defekt vyskytuje pouze u konkrétního systému). Pokrok testování je zaznamenáván přesunem položek ze stavu „Ready For Test“ do stavu „Ready For Deploy“ (připraveno k vydání). V případě, kdy daná položka nesplňuje akceptační kritéria, zůstává ve stavu „Ready For Test“ a vyčkává na to, až si ji vývojář opět převezme k přepracování, nebo se nad dalšími kroky vede diskuze se členy týmu, viz následující cíl PA2.4.SG4.

Popis nedostatků:

- Vzniklému defektu by měla být přiřazena míra rizika, jako u každého zakládaného úkolu (vyplývá z nalezeného nedostatku PA2.2.SG1).

Tabulka 19 - Hodnocení – PA2.4.SG3 - Perform Test Execution

Identifikátor cíle	PA2.4.SG4
Název cíle	Manage Test Incidents to Closure
Hodnocení cíle	F

Rozdíly v interpretaci tohoto cíle mezi použitými zdroji:

- Mezi zdroji *Agile Testing: A Practical Guide for Testers and Agile Teams* [6] a *TMMi in the Agile world* [5] nebyly nalezeny žádné zásadní rozdíly.

Aplikace cíle v agilním prostředí:

Ukončení/uzavření nalezených defektů a úkolů je v agilním prostředí řešeno naprosto totožně. Je třeba identifikovat, zda nalezený problém blokuje vydání a uzavření závislého úkolu. Tato skutečnost je diskutována během denních schůzí. [5, 6]

V případě, že defekt blokuje dokončení konkrétního úkolu (položky v backlogu), mohou padnout dva typy rozhodnutí. Rozhodnutí o vyřešení problému v současné iteraci – defekt se vyřeší, znovu otestuje. Pokud vše proběhne v pořádku, může být úkol vydán a společně s defektem uzavřen. Druhým rozhodnutím může být jeho odložení i s blokováným úkolem k řešení v příští iteraci. [5, 6]

Dalším případem je defekt, který nic neblokuje (např. nově nalezený defekt v produkčním prostředí). Takový defekt se zařadí mezi ostatní položky v backlogu. Podle udělené priority se následně rozhoduje, zda je nutná jeho okamžitá oprava, nebo může vyčkat na pozdější řešení. [5]

Velmi užitečnou funkci představuje vyhledávání mezi uzavřenými defekty. Je možné, že se z nějakého důvodu již uzavřený defekt znovu objeví. Při jeho nalezení je velice pravděpodobné, že v kartě defektu byla řešena jeho příčina a způsob vyřešení. Uzavřené defekty tedy mohou sloužit jako databáze znalostí, která napomáhá k urychlení nápravných činností. Dalším případem může být uzavření defektu z důvodu jeho bezvýznamné priority. Pokud např. nový tester nalezne takovýto defekt, nemusí se zabývat jeho opětovným zaznamenáváním. [6]

Reference:

- *TMMi in the Agile world* [5]
- *Agile Testing: A Practical Guide for Testers and Agile Teams* [6]

Popis aktuálního stavu v hodnocené společnosti:

Společnost plně pokrývá všechny praktiky a činnosti, které jsou uvedeny výše. Popis aktuálního stavu je tedy roven aplikaci cíle viz výše.

Popis nedostatků:

- Nebyly nalezeny nedostatky, které by bránily dosažení tohoto cíle modelu TMMi.

Tabulka 20 - Hodnocení – PA2.4.SG4 - Manage Test Incidents to Closure

6.2.5. PA: Test Environment

Identifikátor cíle	PA2.5.SG1
Název cíle	Develop Test Environment Requirement
Hodnocení cíle	F

Rozdíly v interpretaci tohoto cíle mezi použitými zdroji:

- Mezi zdroji *Agile Testing: A Practical Guide for Testers and Agile Teams* [6] a *TMMi in the Agile world* [5] nebyly nalezeny žádné zásadní rozdíly.

Aplikace cíle v agilním prostředí:

Smyslem testovacího prostředí je poskytnutí dostatečně stabilního prostoru pro vykonávání testů. Toto prostředí by mělo být co nejvěrnějším odrazem provozního prostředí. Je důležité především v době, kdy už je produkt reálně využíván, nebo představován potenciálním uživatelům. V tomto případě se nové funkce a přírůstky produktu nejdříve testují v testovacím prostředí. Výskyt defektů tedy neznamena omezení funkčnosti reálně využívaného produktu. [5, 6]

Při agilním vývoji je stabilita testovacího prostředí velmi důležitá. Jelikož k testování dochází průběžně po celou dobu iterace, tak pády testovacího prostředí mohou tento proces zastavovat a zpomalovat tím celou iteraci. Tomu se nejčastěji předchází tím, že vývojáři nejprve používají své osobní vývojové prostředí např. ve formě sandboxů, na kterém provádějí vlastní předběžné testování. Teprve potom, co jim bezchybně projdou všechny testy, mohou být změny nahrány na testovací prostředí, kde jsou prováděny detailní testy na všech úrovních. [6]

Požadavky na testovací prostředí se většinou stanovují na počátku projektu. Často je pro to vytyčena tzv. nultá iterace, kde se řeší vše potřebné k zahájení plnohodnotného vývoje. Případně při plánování releasu nebo sprintu jsou požadavky revidovány a zkoumány. Ověřuje se tím, zda prostředí splňuje všechna kritéria pro testování konkrétních položek. [5]

Reference:

- *TMMi in the Agile world* [5]
- *Agile Testing: A Practical Guide for Testers and Agile Teams* [6]

Popis aktuálního stavu v hodnocené společnosti:

Cíl je plně v souladu s praktikami hodnocené společnosti. Požadavky na testovací prostředí byly stanoveny v tzv. nulté iteraci na samém počátku projektu a byly úspěšně naimplementovány, viz plnění cíle PA2.5.SG2.

Popis nedostatků:

- Nebyly nalezeny nedostatky, které by bránily dosažení tohoto cíle modelu TMMi.

Tabulka 21 - Hodnocení – PA2.5.SG1 - Develop Test Environment Requirement

Identifikátor cíle	PA2.5.SG2
Název cíle	Perform Test Environment Implementation
Hodnocení cíle	P

Rozdíly v interpretaci tohoto cíle mezi použitými zdroji:

- *Agile Testing: A Practical Guide for Testers and Agile Teams* [6] se samotnou implementací testovacího prostředí nezabývá. Implementace je zde přirozeným výsledkem předchozího cíle PA2.5.SG1.
- *TMMi in the Agile world* [5] doporučuje implementaci testovacího prostředí v co nejranější fázi vývoje. Především však záleží na tom, jak se k této problematice postaví samotný tým.

Aplikace cíle v agilním prostředí:

Nasazení vývojového prostředí podle specifikovaných požadavků, bývá často jedním z úkolů počáteční iterace. Samozřejmě dochází i k průběžnému zdokonalování a upravování během vývoje. Měla by být však zaručena jeho připravenost k provádění testů v okamžiku, kdy je tomu potřeba. [5]

Reference:

- *TMMi in the Agile world* [5]

Popis aktuálního stavu v hodnocené společnosti:

Ve společnosti se vyskytují dvě testovací prostředí. První se nazývá „STAGE“. Toto prostředí pracuje s vlastní oddělenou testovací databází. Data jsou tedy nezávislá na tzv. živých datech z produkčního prostředí. Poté, co vývojář otestuje nový kód na svém sandboxovém prostředí, nahraje novou verzi na „STAGE“. V tomto okamžiku se spustí automatizované regresní testy. Pokud testy neproběhnou s kladným výsledkem, vytvoří se report o vzniklých problémech. V tuto chvíli je testovací prostředí, v mnoha případech,

nepoužitelné. V lepším případě jsou vzniklé chyby okamžitě opraveny a dochází k novému nahrání verze na testovací prostředí. V případě rozsáhlejšího problému se nahrává poslední funkční verze. Když proběhnou všechny regresní testy, je testovací prostředí s novou verzí připraveno k manuálnímu testování. Po dotestování a odsouhlasení, že nalezené defekty nebrání vydání, se nová verze přesouvá na druhé testovací prostředí s již produkční databází. Jelikož se jedná o tzv. předprodukční prostředí, tak je ve společnosti známo pod názvem „PRELIVE“. Toto prostředí naprosto odpovídá tomu produkčnímu, jen k němu zatím nemají přístup koncoví uživatelé. Přistupuje se k němu přes nastavení proxy. V tomto prostředí je provedeno druhé kolo testování, které již není tak podrobné. Ověřují se pouze, zda předprodukční data a prostředí nemají vliv na správné fungování nové verze. Pokud je vše v pořádku, může se verze nahrát na produkci.

Popis nedostatků:

- Často se stává, že položky, které jsou označeny „K otestování“, se ve stejné chvíli nenacházejí na testovacím prostředí. Díky tomu občas dochází k nepřehlednosti pro testera a zúčastněné strany, kteří mohou začít testovat a vzápětí se dozvědět, že netestují novou verzi produktu, která je označena k otestování.
- Při selhání automatizovaných testů, se mnohdy prostředí stává nepoužitelné a přerušuje proces testování.

Tabulka 22 - Hodnocení – PA2.5.SG2 - Perform Test Environment Implementation

Identifikátor cíle	PA2.5.SG3
Název cíle	Manage and Control Test Environments
Hodnocení cíle	F

Rozdíly v interpretaci tohoto cíle mezi použitými zdroji:

- *Agile Testing: A Practical Guide for Testers and Agile Teams* [6] – řízení a kontrola testovacího prostředí se v tomto zdroji nenachází. Je možné si to vyložit tak, že zajištění jeho použitelnosti a funkčnosti je naprosto samozřejmé.
- *TMMi in the Agile world* [5] tento cíl blíže specifikuje, viz text níže.

Aplikace cíle v agilním prostředí:

Prostředí by mělo být pod neustálou kontrolou, aby byla zaručena jeho stabilita a připravenost. Při zjištění problému je důležité vědět, kdo se správou prostředí zabývá. V agilních společnostech to může být samostatná jednotka, která se nachází mimo vývojový tým (např. administrátor infrastruktury). V takovém případě se problém

nahlásí, ale neeviduje se jako součást sprintu. V opačném případě může být správcem regulérní člen agilního týmu. Zde se daný problém zapíše mezi defekty a je s ním nakládáno stejně jako s ostatními položkami v produktovém backlogu. [5]

Reference:

- *TMMi in the Agile world* [5]

Popis aktuálního stavu v hodnocené společnosti:

Kontrola stavu testovacího prostředí je ve společnosti zaručena tím, že jsou na něm, v častých intervalech, spouštěny automatizované testy. Pokud začnou tyto testy hromadně padat, ve valné většině případů to značí problém na daném prostředí. Při vzniku problému dochází k jeho okamžité identifikaci a podle zjištěné povahy se plánují další kroky. V případě, že je vývojový tým schopen daný problém vyřešit svými silami, dochází k okamžité práci na opravě, nebo je založen defekt pro pozdější řešení. Pokud však tým na opravu nestačí, je kontaktován administrátor, který problém řeší vzdáleně.

Popis nedostatků:

- Nebyly nalezeny nedostatky, které by bránily dosažení tohoto cíle modelu TMMi.

Tabulka 23 - Hodnocení – PA2.5.SG3 - Manage and Control Test Environment

6.3. Vyhodnocení dosažené úrovně

V předchozí kapitole bylo provedeno porovnání současného stavu procesů testování proti specifickým cílům druhé úrovně modelu TMMi. Každému cíli byla udělena hodnotící známka, která představuje míru jeho plnění. Každý specifický cíl mohl být hodnocen známkou F, P, N, NR nebo NA. Znamka F značí zcela naplněný cíl. To znamená, že současná situace ve společnosti je plně v souladu s uvedenou aplikací cíle v agilním prostředí. Pokud je cíl ve společnosti částečně řešen, ale nedochází k jeho plnému splnění, uděluje se mu známka P (částečně splněno). Znamka N značí zcela nesplněný cíl. Pokud se během hodnocení vyskytuje překážka, kvůli které není možné daný cíl ohodnotit, uděluje se známka NR – nehodnoceno. Posledním případ představuje neaplikovatelnost daného cíle v hodnocené společnosti – zde se uděluje známka NA (neaplikovatelné). Následující tabulka obsahuje všechny hodnocené cíle s udělenými známkami (viz Tabulka 24 - Výsledek hodnocení - 2. úroveň TMMi). Splnění či nesplnění každé procesní oblasti určuje nejnižší dosažená známka specifického cíle, který pod danou oblast patří. Aby se mohla procesní oblast považovat za splněnou, musí být všechny její specifické cíle ohodnoceny známkou F. Druhé úrovně modelu TMMi je dosaženo v případě, že jsou všechny procesní oblasti zcela naplněny a ohodnoceny známkou F.

Komponenty TMMi			Hodnocení	
PA1	Test Policy and Strategy		P	
	SG1	Establish a Test Policy		P
	SG2	Establish a Test Strategy		P
	SG3	Establish Test Performance Indicators		F
PA2	Test Planning		P	
	SG1	Perform Risk Assessment		P
	SG2	Establish a Test Approach		P
	SG3	Establish Test Estimates		P
	SG4	Develop a Test Plan		P
	SG5	Obtain commitment to the Test Plan		F
PA3	Test Monitoring and Control		F	
	SG1	Monitor Test Progress Against Plan		F
	SG2	Monitor Product Quality Against Plan and Expectations		F
	SG3	Manage Corrective Actions to Closure		F
PA4	Test Design and Execution		P	
	SG1	Perform Test Analysis and Design using Test Design Techniques		P
	SG2	Perform Test Implementation		F
	SG3	Perform Test Execution		P
	SG4	Manage Test Incidents to Closure		F
PA5	Test Environment		P	
	SG1	Develop Test Environment Requirement		F
	SG2	Perform Test Environment Implementation		P
	SG3	Manage and Control Test Environments		F

Tabulka 24 - Výsledek hodnocení - 2. úroveň TMMi

Z výsledné tabulky hodnocení je patrné, že jsou ve společnosti plně, či částečně pokryty všechny specifické cíle. Dvě procesní oblasti, konkrétně PA3 Test Monitoring and Control a PA4 Test Design and Execution, jsou ohodnoceny jako zcela splněné. Procesní oblasti PA1 Test Policy and Strategy, PA2 Test Planning, PA4 Test Design and Execution a PA5 Test Environment, jsou splněny pouze zčásti. Na základě zjištěných nedostatků těchto oblastí, je v následující kapitole vypracován návrh na zlepšení. Po převedení tohoto návrhu do reality, by měla společnost splňovat druhou úroveň modelu TMMi.

7. Návrh nového řešení

Tato kapitola obsahuje návrh na zvýšení zralosti procesů testování. Ten je koncipován tak, aby vhodně zapadal do kontextu společnosti Glogster. Návrh je rozdělen na jednotlivé procesní oblasti modelu TMMi. V těchto oblastech se zabývá nedostatky, které byly zjištěny při hodnocení. Smyslem vzniklého návrhu je dodržení specifických cílů druhé úrovně modelu TMMi s ohledem na uchování agilních praktik. Návrh se zakládá na informacích, které byly získány během hodnocení společnosti a na autorových zkušenostech z praxe.

7.1. Návrh na zlepšení PA: Test Policy and Strategy

Model TMMi vyžaduje alespoň stručně (může být na jedné stránce) vypracovanou testovací politiku a strategii. V agilním vývoji je povoleno zahrnout jak testovací politiku, tak i strategii, která na testovací strategii volně navazuje, do jednoho dokumentu. Mohlo by se zdát, že zakládání této dokumentace je v rozporu s agilními principy, kde se klade důraz na ústní formu komunikace a oproštění se od důkladné dokumentace. Agilní princip „*Fungující software před vyčerpávající dokumentací*“ [30] však neznamená, že by se neměla vytvářet žádná dokumentace, ale pouze taková, která nese žádoucí přidanou hodnotu. Testovací politika a strategie jsou dva dokumenty na organizační úrovni, které zastřešují celou oblast testování. Jsou vhodné především k tomu, aby poskytovaly zúčastněným stranám jasný vhled do této oblasti. [5] Konkrétně slouží jako rámec, od kterého se testování ve společnosti odvíjí. Z praktického hlediska lze těchto dokumentů také využít v případě, kdy společnost přijímá nové pracovníky na pozice, které se týkají oblasti testování. Zde mohou sloužit jako možnost poskytnutí materiálů, kde si pracovníci mohou nastudovat vše potřebné pro vstup do problematiky testování v rámci společnosti. Může se tak ušetřit čas na školení. [6]

V dokumentu „politika testování“, jsou definovány cíle a principy testování softwaru, kterými se společnost řídí. Dokument by měl případně obsahovat strukturu testovacího oddělení, informace o požadavcích na testery (stupeň vzdělání, znalosti, případně poskytnuté školení), dodržované normy a metody pro zajištění průběžného zlepšování procesů. Např. norma ISO/IEC/IEEE 29119 poskytuje podrobně uvedené požadavky na vytváření tohoto dokumentu a uvádí také příklad politiky testování pro agilní korporaci (viz Obrázek 7 - Příklad dokumentu Politika testování [53])

D.1 Příklad 1 – Agilní korporace

Agilní korporace je velká vydavatelská společnost produkující časopisy a knihy. Viz více podrobností v úvodu přílohy C.

Politika testování pro Agilní korporaci, V1.2 (13. 2. 2009)

Vydal: Ursula Mayers, Vedoucí vývoje

Schválil: Stephan Blacksmith, Vedoucí testování

Rozsah: Tato Politika testování popisuje korporátní pohled na testování v Agilní korporaci a poskytuje rámec, uvnitř kterého bude probíhat veškeré testování prováděné v interních projektech uvnitř organizace.

Úvod: Agilní korporace uznává potřebu testování svých interních produktů. Náklady na vývoj softwarových systémů vysoké kvality lze rozdělit do čtyř kategorií: náklady na prevenci, náklady na testování, náklady související s interními selháními a náklady spojené s externími selháními. Je obecně levnější zabránit defektům, než je odhalovat a opravovat (náklady na testování plus náklady na interní selhání), a nejvyšší náklady představují externí selhání, která jsou odhalena uživateli. Aby se tomu předešlo, Agilní korporace praktikuje Vývoj řízený testováním (TDD) a Vývoj řízený akceptačním testováním (ATDD), což jsou techniky softwarového vývoje. Ve své implementaci TDD používá Agilní korporace techniky bílé skříňky, která je definována v části 4 normy ISO/IEC/IEEE 29119.

Cíle testování: Cílem testování je poskytnout dostatečné informace k určení aktuální kvality testovaného systému. Jako takové jsou všechny aktivity zaměřené na dosažení tohoto cíle považovány za aktivity testování softwaru (například integrační, systémové, akceptační a regresní testování).

Testovací proces: Testování softwaru bude založeno na testovacích procesech definovaných v ISO/IEC/IEEE 29119-2 a upraveno v souladu s přístupem vývoje.

Struktura testovacího oddělení: Testování bude zajištěno v rámci Agilní korporace z centrální skupiny testerů, kteří jsou na projekt přiděleni. Kromě toho centrální „expert“ na testování softwaru pod Vedoucím testování poskytne projektům podle potřeby konzultační služby v oblasti testování. Struktura testovacího oddělení uvnitř projektu se bude řídit směrnicemi projektu.

Školení testerů: Od všech členů testovacího týmu se očekává odpovídající vysokoškolské vzdělání nebo alespoň minimální úroveň certifikace v oboru testování softwaru. Navíc se od testerů očekává znalost agilních konceptů nebo jejich získání do tří měsíců od připojení se k týmu testerů.

Normy: Dokumentace testování bude založen na normě ISO/IEC/IEEE 29119-3 „Dokumentace testování“, uzpůsobené použití v agilních projektech.

Další relevantní politiky: Politika softwarového vývoje pro Agilní korporaci, V4.3 (12. 12. 2008)

Zlepšování testovacího procesu a stanovení hodnot: Retrospektivy na konci iterací zachytí ponaučení, měření a koncepty zlepšování, které budou poskytnuty hlavnímu testovacímu oddělení.

Obrázek 7 - Příklad dokumentu Politika testování [53]

Strategie testování vstupuje, na rozdíl od testovacího plánu, více do hloubky. Je zde formulováno, jakým způsobem je samotné testování prováděno a jak dosáhnout cílů, jenž jsou definovány v politice testování. Testovací strategie se skládá z několika bodů, které je nutné ve společnosti formulovat. Společnost Glogster by měla, do své „testovací strategie“, zahrnout a popsat minimálně tyto body [3, 6]:

- Řízení a posuzování rizik
- Výběr a nastavení priorit testování
- Úrovně testů
- Regresní testování
- Automatizace testování a použité nástroje
- Správa incidentů
- Vstupní a výstupní kritéria

- Testovací prostředí
- Nastavené ukazatele sledující proces testování

Po vypracování testovací politiky a strategie podle výše zmíněného návrhu, nebude nic bránit plnému splnění související procesní oblasti modelu TMMi.

7.2. Návrh na zlepšení PA: Test Planning

V oblasti plánování bylo nalezeno několik nedostatků. Zejména se jedná o nesystematické posuzování rizik a z toho plynoucí nemožnost definovat testovací přístup na základě konkrétního stupně rizika. Dalším nedostatkem jsou nedostatečné časové odhady, které jsou v mnoha případech řešeny jen ústně a nelze je tak vhodně kontrolovat. Eliminace uvedených nedostatků je cílem této podkapitoly.

Posuzování rizik představuje velmi důležitou aktivitu, která umožňuje značně eliminovat nežádoucí dopad na výsledný produkt. Slouží k tomu, aby se tým na vysoce rizikové oblasti zaměřil s obzvlášť velkou pečlivostí. Především testeři by zde měli razantně zvýšit svou pozornost a snažit se danou položku otestovat co nejdůkladněji. Posouzená rizika se následně dají použít jako podnět pro vznik regresních testů. Mezi jednotlivými položkami se nacházejí i nalezené defekty. U těchto defektů je třeba také posoudit míru rizika, protože se následně stávají předmětem testování stejně, jako ostatní položky v produktovém backlogu (posouzením míry rizika u vzniklých defektů dojde ke splnění procesní oblasti PA4 Test Design and Execution).

Posuzování by mělo probíhat již ve fázi zapisování jednotlivých úkolů do produktového backlogu, případně při plánování releasu či sprintu. Společnost má možnost vybrat si ze dvou navržených způsobů evidence rizik. Nejjednodušším způsobem je stanovení slovně reprezentované stupnice závažnosti rizika (např. nízké, střední, vysoké). V takovéto formě se mohou rizika udělovat jednotlivým položkám v produktovém backlogu. Druhý více podrobný způsob je založen na součinu dvou hodnot. První hodnota představuje pravděpodobnost výskytu problému a druhá závažnost dopadu. Pravděpodobnosti i dopadu se uděluje hodnota ve škále 1-5 (čím větší hodnota tím hůře). Následným součinem těchto hodnot je určen výsledný stupeň rizika (viz Obrázek 8). Ten se může zapisovat buď v číselné formě, nebo je možné vytvořit škály, které budou svou závažnost vyjadřovat slovně, viz první způsob posuzování rizik. Např. 0-9=nízké riziko, 10-17=střední riziko, 18-25=vysoké riziko. Zaznamenávání rizika je ve společnosti možné provádět formou štítků, které se přidávají k jednotlivým položkám ve využívaném nástroji GitLab. Díky tomu bude možné úkoly řadit a filtrovat posouzené míry rizika.

#	Item	Impact	Probability	Risk
1	Incorrect cost displayed	4	2	8
2	User can't choose different shipping option	5	1	5
3	Item isn't eligible for selected shipping option, but selection allowed	3	2	6
4	Estimated cost doesn't match actual cost at checkout	3	4	12
5	Invalid postal code entered and not caught by validation	4	1	4
6	User can't understand shipping option rules	2	3	6
7	User can't change shipping address	5	2	10
8	User changes shipping address, but cost doesn't change accordingly	5	4	20

Obrázek 8 - Příklad posouzení rizik [6]

Aby zaznamenaná rizika plnila svůj účel, je třeba určit, jak se bude k jednotlivým stupňům přistupovat. Za předpokladu, že je stupeň rizika zaznamenáván v podobě, jak je navrženo výše, by aplikovatelný přístup mohl vypadat takto:

- V případě, že tester má za úkol otestovat více položek – pořadí testování určuje jejich stupeň rizika (od nejvyššího po nejnižší). V případě nedostatečné časové rezervy je zaručeno, že nejrizikovější položky jsou otestovány.
- Riziko: Vysoké – tyto položky by měly být předmětem diskuze o jejich zařazení mezi regresní testy. V nejlepším případě by měly být zařazeny mezi automatizované testy.

Další nedostatek, bránící splnění této procesní oblasti, byl shledán v nedostatečném určování časových odhadů na vývoj a testování. Časové odhady by se měly zaznamenávat ke každému případu – při nejlepším v etapě plánování sprintu, kdy jsou vedeny diskuze nad výběrem konkrétních případů z produktového backlogu. Díky tomu bude podpořeno časové ukotvení daného sprintu. Navíc jasné ukotvení této činnosti v metodice napomůže k postupnému zdokonalování přesnosti odhadů.

7.3. Návrh na zlepšení PA: Test Environment

V této oblasti byly nalezeny dva nedostatky týkající se testovacího prostředí. Pro pochopení podstaty prvního problému, je nejdříve nutné objasnit jeho příčinu. Projekt Glogster se větví na dílčí oblasti (podprojekty), které jsou řešeny paralelně. V jednotlivých sprintech jsou však tyto podoblasti plánovány zároveň. Mnohdy se navzájem doplňují, nebo jsou na sobě dokonce závislé. Leckdy se však stává, že úkoly jsou označeny jako „K otestování“ a přitom se nenacházejí na testovacím prostředí. Je to způsobeno tím, že vydávání nových verzí má na starost hlavní vývojář, který většinou vydává novou verzi se svými změnami, i se

změnami ostatních vývojářů. Dochází tedy k tomu, že jiný programátor má daný úkol splněn a otestován na vlastním vývojářském prostředí. V takovéto chvíli úkol označí „K otestování“ a dál se již o nic nestará. Často tak dochází ke zmatení testera, který začne danou položku testovat a vzápětí se dozvídá, že se položka nenachází na daném prostředí. Tato skutečnost se v současnosti řeší ústní a elektronickou komunikací, což ale není zcela komfortní řešení.

Vhodnější řešení by mohla představovat automatizace vydávání těchto čekajících změn, nebo umožnit vydávání i ostatním vývojářům. Tím by se zaručilo, že vše, co je v úkolech označeno k otestování, se zároveň nachází v testovacím prostředí.

Druhý problém spočívá v odstavkách testovacího prostředí při chybovém nahrání nové verze. V případě, že se při nahrávání vyskytne nějaká chyba, dojde k zastavení vydávací pipeline⁷, jejíž součástí je také část s automatizovanými testy. Při selhání automatizovaných testů se mnohdy prostředí stává nepoužitelné a přerušuje proces testování.

Tento problém by mohl být řešen prostřednictvím automatického rollbacku. To znamená, že při výskytu problému v procesu vydávání dojde k automatickému vratnému procesu, kdy se na testovací prostředí nahraje poslední bezchybná a funkční verze. Znatelně se tak zkrátí prodleva v podobě přerušení testování, které právě probíhá na předchozí verzi.

7.4. Posouzení návrhu

Účelem této podkapitoly je evaluace výsledků diplomové práce. Zakládá se na dotazování se klíčových členů vývojového týmu. Těm jsou předkládány otevřené otázky referující o využitelnosti modelu TMMi pro zvyšování zralosti procesů testování v agilním prostředí. Nejdůležitější část však představuje evaluace hodnocení společnosti podle modelu TMMi a vzniklý návrh. Aby byl zastoupen jak manažerský, tak i technický pohled, bude dotazován projektový manažer Martin Santorel, klíčoví vývojáři Filip Šubr a Martin Kluska. Aby byli dotazování uvedeni do kontextu, byla jim k prostudování poskytnuta tato práce s doporučením zaměřit se především na kapitoly: 4 Test Maturity Model Integration, 6 Hodnocení současného stavu procesů testování a 7 Návrh nového řešení.

Následně jim byly předloženy následující otázky:

- Jak se stavíte k průběžnému zlepšování procesů?

⁷ Vydávací pipeline – jedná se o sled aktivit při vydávání nové verze softwaru. Ve společnosti Glogster se vyskytuje v nástroji Jenkins, který má na starost automatizaci celého vydávacího procesu.

- Myslíte si, že model TMMi může sloužit jako vhodný rámec pro zvyšování zralosti procesů testování v agilním prostředí?
- Jak na vás působí výsledky hodnocení v této práci?
- Chtěl byste k hodnocení něco dodat?
- Je vzniklý návrh pro danou společnost vhodný? Pokud ne, tak proč?
- Je z vašeho pohledu vzniklý návrh proveditelný? Pokud ne, tak proč?

Odpovědi jednotlivých členů vývojového týmu jsou obsahem následující podkapitoly.

7.4.1. Odpovědi členů vývojového týmu společnosti Glogster

Odpovídá projektový manažer Martin Santorel

1. Jak se stavíte k průběžnému zlepšování procesů?

Z manažerského pohledu vnímám kontinuální vylepšování, jak procesů v oblasti testování, tak i postupů řízení agilního vývoje, jako nezbytný a velice důležitý proces pro efektivní fungování společnosti, který je zapotřebí pro vznik kvalitního softwarového produktu. Neustálou práci na vylepšování procesů a zároveň zapracovávání zpětné vazby z trhu vnímám jako zcela klíčovou pro vznik fungujícího produktu s přidanou hodnotou. Zároveň tyto techniky vnímám jako prostředek k efektivnímu vývoji a získání spokojených zákazníků. Dobře navržené a implementované procesy pomáhají společnosti efektivně eliminovat rizika neefektivního vývoje a dodání nekvalitního či nepoužitelného produktu. Pro vyspělé technologické společnosti je nepřetržité zlepšování velmi důležité, aby udrželi tempo s konkurencí.

2. Myslíte si, že model TMMi může sloužit jako vhodný rámec pro zvyšování zralosti testovacích procesů v agilním prostředí?

Po shlédnutí výstupů této práce, kde byl využit model TMMi, si myslím, že daný model může reálně pomoci agilně orientovaným ICT⁸ společnostem se dobře zorientovat v dané problematice a skutečně může pomoci s nastavením nových, nebo zlepšením současných procesů. Jsem přesvědčen, že model TMMi skutečně může společnosti dopomoci k dosažení vyšší úrovně vyspělosti procesů testování.

⁸ ICT – Informační a komunikační technologie

Navíc můžeme zpětně vyhodnocovat její účinnost a přínosy při týmové retrospektivě.

3. Jak na vás působí výsledky hodnocení v této práci?

Dívám se na toto spíše z organizačního pohledu. Provedené hodnocení poukazuje na možnosti zlepšení, které by mohlo vést k vyšší efektivitě týmu při vývoji softwarového produktu. Konkrétně mi tento model/hodnocení pomohlo se dobře zorientovat v dané problematice. Poukázalo na nedostatky a jasně identifikovalo oblasti, na které se můžeme při zlepšování procesů zaměřit.

4. Chtěl byste k hodnocení něco dodat?

Velice přehledně zpracované hodnocení s dobře popsány nedostatky a jasně vytyčenými možnostmi a návrhy na co se zaměřit. Velmi dobře identifikované příležitosti, v jaké oblasti bychom se mohli posunout na vyšší úroveň s konkrétními návrhy. Především skvělý podklad pro rozvinutí debaty s týmem o tom, jak se ke konkrétním oblastem postavit a jak integrovat vylepšení postupů a procesů k zajištění lepší efektivity vývoje a úspory času.

5. Je vzniklý návrh pro danou společnost vhodný? Pokud ne, tak proč?

Ano. Vzniklý návrh a podrobná analýza jednotlivých oblastí je vhodná a přínosná pro společnost Glogster a určitě z ní můžeme čerpat. S týmem se skutečně budeme bavit o tom, jak vzniklý návrh uchopit, skutečně realizovat a integrovat do provozu.

6. Je z vašeho pohledu vzniklý návrh proveditelný? Pokud ne, tak proč?

Ano, vzniklý návrh je proveditelný ve všech bodech. Některé body se mohou drobně lišit nebo upravovat po interakci se členy týmu ve společnosti tak, abychom dosáhli maximální efektivity a nasazení procesů tak, aby byly přínosné a reálně pomáhaly týmu v efektivní práci. Samotný proces nasazení testovacích procesů bude také podléhat testování a retrospektivě týmu, aby se vyhodnotily přínosy a zpětně se zapracovaly vylepšení jednotlivých postupů. Ve své podstatě je agilní vývoj neustálým procesem vylepšování, především procesů vývoje (jejichž součástí jsou i procesy testování) a komunikace mezi týmy a zadavatelem.

1. Jak se stavíte k průběžnému zlepšování procesů?

Rozhodně podporuji a prosazuji. Jsem zastávce větší automatizace pro odlehčení “manuální pravidelné práce”.

2. Myslíte si, že model TMMi může sloužit jako vhodný rámec pro zvyšování zralosti testovacích procesů v agilním prostředí?

Ano, avšak závisí na velikosti společnosti / projektu. Ne vždy je dostatek prostoru vše testovat či se řídit dle “směrnic”.

3. Jak na vás působí výsledky hodnocení v této práci?

Libí se mi, že je v každé hodnotící tabulce uvedený příklad plnění daného cíle a následná reálná ukázka aktuálního stavu.

4. Chtěl byste k hodnocení něco dodat?

Pro lepší splnění části PA2.5.SG2 by bylo nejlepší mít volitelně pro různé user stories možnost si spustit celé prostředí odděleně (tj., mít N-počet prostředí), bohužel technicky je to příliš náročné.

5. Je vzniklý návrh pro danou společnost vhodný? Pokud ne, tak proč?

Ano. Je vidět, že daný návrh je vytvořen společností Glogster na míru, a proto neobsahuje přehnané nároky, které by pro danou společnost byly nevhodné.

6. Je z vašeho pohledu vzniklý návrh proveditelný? Pokud ne, tak proč?

Ano, dle hodnocení se jedná spíše o připravení procesů, jejichž implementace by nemusela být tak náročná. Pouze PA2.5.SG2 je, jak jsem již výše zmínil složitější. Jediné řešení je, že všechny prostředí bude možné přesunout do cloudu (např. docker). Na této části se již částečně pracuje.

1. Jak se stavíte k průběžnému zlepšování procesů?

Kladně. Agilní způsob vývoje je založen na neustálém vylepšování procesů na základě zpětné vazby.

2. Myslíte si, že model TMMi může sloužit jako vhodný rámec pro zvyšování zralosti testovacích procesů v agilním prostředí?

Může a určitě je jeho implementace vhodným začátkem, pokud společnost žádný takový rámec ještě nepoužívá. Je ovšem potřeba pamatovat na to, že TMMi určitě není jediná možnost.

3. Jak na vás působí výsledky hodnocení v této práci?

Výsledky se do značné míry shodují s tím, co jsem už věděl nebo tušil. To mě příjemně překvapilo a z mého pohledu dodalo TMMi určitou váhu. Je velmi pravděpodobné, že se v budoucnu budeme snažit o lepší splnění cílů.

4. Chtěl byste k hodnocení něco dodat?

Líbí se mi přehledné rozdělení do tabulek, kdy není potřeba hodnocení a popisy nedostatků v jednotlivých bodech dohledávat ve větších celcích textu.

5. Je vzniklý návrh pro danou společnost vhodný? Pokud ne, tak proč?

Ano, hodnocení i návrh řešení odpovídá agilní povaze Glogsteru a jednotlivé body lze snadno seřadit podle priorit a řešit víceméně nezávisle na ostatních.

6. Je z vašeho pohledu vzniklý návrh proveditelný? Pokud ne, tak proč?

Návrh je proveditelný i když je možné, že v době implementace budou některé body zbytečné nebo nerelevantní.

7.5. Vyhodnocení odpovědí

Odpovědi členů vývojového týmu se ve valné většině svou povahou shodují. Všichni dotazovaní uvádí, že neustálé zlepšování ať už procesů vývoje a testování nebo praktik a dovedností členů týmu, představuje nedílnou součást agilního vývoje. Z odpovědí také plyne, že model TMMi může být vhodnou variantou pro využití v agilním vývoji, pokud se tým rozhodne svůj proces zlepšování procesů testování pojmout formálně s využitím nějakého modelu/nástroje, který je k tomu určený. Model TMMi je vhodný pro identifikaci problematických procesních oblastí a nabízí sadu doporučených praktik a procesů, kterými lze nalezené nedostatky eliminovat. Při jeho aplikaci je však třeba brát v potaz povahu agilního týmu a využívané praktiky. To znamená, že agilní tým by se neměl primárně soustředit na slepé plnění konkrétního modelu nebo směrnic (pokud to povaha softwarového produktu nevyžaduje), ale zároveň jich může být využito pro identifikaci

konkrétních nedostatků, jejichž eliminace může mít pro vývojový tým vhodnou přidanou hodnotu.

Samotné hodnocení společnosti získalo velice kladnou odezvu. Členové týmu zmiňují přehlednost každé hodnotící tabulky, kde jsou nejprve uvedeny možnosti plnění každého cíle a následně současná situace ve společnosti. Hodnocení tak odhaluje jednotlivé nedostatky, které brání ve splnění konkrétního cíle. Projektový manažer dále dodává, že jednotlivé tabulky mohou sloužit jako vhodný podnět k diskusi celého týmu o tom, jak se ke konkrétním oblastem postavit a jak integrovat vylepšení postupů a procesů k zajištění lepší efektivity vývoje a úspory času.

Vzniklý návrh byl vyhodnocen jako proveditelný a vhodný pro společnost Glogster. Je však třeba vzniklý návrh detailně diskutovat se členy týmu, přičemž se musí daným částem návrhu přiřadit prioritizace s ohledem na míru reálného přínosu pro společnost. Je možné, že budou realizovány jen ty části návrhu, které budou vyhodnoceny jako nejvíce přínosné a vhodné pro investování času na jejich realizaci.

Odstavec výše zároveň poukazuje na nevýhodu využití modelu TMMi, která spočívá v jeho stupňovité povaze. Aby daná společnost mohla dosáhnout druhé úrovně zralosti tohoto modelu, musí zcela splňovat procesní oblasti a cíle dané úrovně. Přičemž nedbá na reálné priority a hodnoty dané společností. Daleko vhodnější by bylo, kdyby model TMMi umožnil způsob plynulé reprezentace. V takovém případě by si společnost mohla vybrat jen takové procesní oblasti, které jsou pro ni důležité. Teprve až na úrovních vybraných oblastí by se hodnotila a zvyšovala jejich úroveň zralosti. Nicméně i přes tento nedostatek je možné model TMMi vhodně využít pro účely zvyšování zralosti procesů testování v agilním vývoji. Model může sloužit především k identifikaci problematických procesních oblastí s poskytnutím doporučených praktik a postupů, jak zjištěné problémy eliminovat. Zároveň však nenakazuje striktně daná doporučení plnit, ale dává možnost, aby si každá společnost plnění jednotlivých cílů nastavila podle svých potřeb a priorit.

Závěr

V úvodní kapitole diplomové práce byl formulován hlavní cíl, který spočívá v ohodnocení současných procesů testování společnosti Glogster a na jeho základě vytvoření návrhu na zlepšení procesů testování tak, aby bylo možné dosáhnout druhé úrovně (definovaná) modelu TMMi. Pro dosažení hlavního cíle bylo potřeba provést rešerši dostupných kvalifikačních prací, odborných publikací a internetových zdrojů. Čerpáním především z výstupů provedené rešerše bylo nutné dosáhnout dalších definovaných dílčích cílů diplomové práce (DC1 – DC7).

Prvním dílčím cílem DC1 bylo ucelení teoretických poznatků o řízení kvality, testování softwaru a zlepšování kvality softwaru. Tento dílčí cíl byl naplněn kapitolou č. 1. Kapitola nejprve seznamuje čtenáře s oblastí kvality softwaru a jejím vlivem na tržní konkurenceschopnost. Dále je v kapitole uvedena oblast řízení kvality softwaru, která má na kvalitu výsledného produktu velký vliv. Jelikož nejvýznamnějším nástrojem řízení kvality jsou testovací činnosti, byla zde popsána i tato oblast. V rámci podkapitoly testování softwaru byla tato oblast popsána i s uvedením nejvyužívanějších technik testování a jeho členěním. Následně bylo objasněno testování v agilním kontextu a na závěr byly stručně popsány procesy testování, na které se model TMMi zaměřuje. Druhá a třetí kapitola obsahuje uvedení do problematiky zlepšování procesů testování prostřednictvím modelů a rámců k tomu určených. Před samotným představením modelu TMMi bylo důležité, v druhé kapitole, čtenáře seznámit s termínem zlepšování kvality softwaru, protože daný model představuje jeden z možných nástrojů spadající do této oblasti. Byl zde také uveden model CMMI, ze kterého model TMMi vychází. Třetí kapitola se zaměřuje na zvyšování zralosti procesů testování. Zde je zmíněný termín osvětlen a obsahuje stručný popis dvou dalších modelů, které jsou pro tuto oblast (mimo TMMi) využívány. Tato kapitola slouží jako úvod ke kapitole o samotném TMMi.

Dílčí cíl DC2 – Představit model TMMi pro zvyšování zralosti procesů testování s možnostmi jeho uplatnění v agilním prostředí, je obsahem čtvrté kapitoly diplomové práce. V této kapitole je model detailně popsán. Jsou zde vymezeny všechny jeho úrovně, komponenty a informace o možném hodnocení procesů podle modelu TMMi. Důležitou část také představuje podkapitola týkající se použitelnosti modelu v agilním prostředí.

V páté kapitole byla představena společnost, která byla podrobena neformálnímu hodnocení procesů testování podle modelu TMMi a jeho druhé úrovni zralosti. K tomu, aby hodnocení bylo možné, musela být určena metoda hodnocení. Vymezení této metody bylo

provedeno také v rámci této kapitoly, díky čemuž byl splněn dílčí cíl DC3 – Zvolit, případně upravit, hodnotící metodu TMMi pro následné provedení neformálního hodnocení společnosti.

Šestá kapitola se zaměřuje na dílčí cíl DC4 – Provést neformální hodnocení konkrétní společnosti proti druhé úrovni zralosti modelu TMMi. Tato kapitola představuje jednu z klíčových částí práce, kde bylo provedeno hodnocení současných procesů testování společnosti. Bylo zde zjištěno několik nedostatků, které brání k dosažení druhé úrovně modelu TMMi. Ty se staly stěžejním zdrojem pro splnění následujícího dílčího cíle.

Dílčí cíl DC5 – Na základě výsledků hodnocení vytvoření návrhu, který umožní dosáhnout druhé úrovně modelu TMMi, splňuje poslední sedmá kapitola obsahuje návrh nového řešení, který má za úkol eliminaci zjištěných nedostatků, aby bylo v budoucnu možné dosáhnout druhé úrovně („řízená“) modelu TMMi. V této kapitole je zároveň zahrnut poslední dílčí cíl DC6 – Evaluace vypracovaného návrhu prostřednictvím rozhovorů s vývojovým týmem společnosti.

Úvodem definované dílčí cíle (DC1 – DC6) jsou v kapitolách diplomové práce dosaženy, čímž je naplněn i hlavní cíl této práce. Diplomová práce po dosažení hlavního a dílčích cílů je přínosem především pro společnost, která byla podrobena hodnocení a pro jejíž účely byl vypracován návrh na zlepšení. Společnost Glogster, i přes velkou vytíženost, plánuje daný návrh realizovat. Práce může být také přínosem pro čtenáře, kteří se zajímají o zlepšování kvality softwaru, a především o využitelnost modelu TMMi v agilním vývoji pro malé a střední podniky. Zejména hodnotící tabulky s uvedenými způsoby plnění konkrétních cílů mohou jiným vývojovým týmům posloužit pro jejich vlastní hodnocení procesů testování dle modelu TMMi a jeho druhé úrovně. Hodnocení a návrh, s cílem dosáhnout třetí a vyšších úrovní modelu TMMi s návazností na výsledky této práce, může sloužit jako vhodný námět pro budoucí závěrečné práce.

Myšlenkou této práce bylo také ověření, zda i tak robustní rámec, jakým je model TMMi, u kterého se spíše očekává uplatnění ve velkých společnostech s početným vývojářským týmem, je možné využít v kontextu malého vývojového týmu pracujícího na středně velkých projektech a startupech. A jak už z práce vyplývá, tak i podle autorova názoru je možné tvrdit, že model vhodný je. Model TMMi velmi dobře napomáhá odhalit problémové oblasti a opomíjené nedostatky. Díky svým vlastnostem a přizpůsobivosti k agilním praktikám může model velmi dobře sloužit ke zlepšování a zvyšování zralosti procesů testování.

Seznam literatury

- [1] ŠNAJDR, Petr. *Průvodce procesními metodikami – Úvod do CMMI díl I. | Podnikatelské příběhy* [online]. 2016 [vid. 2018-10-14]. Dostupné z: <https://www.podnikatelskepribehy.cz/pruvodce-procesnimi-metodikami-uvod-do-cmmi-dil-i/>
- [2] TOMÁŠ HRŮZA. *Jak poznat vyspělé procesy?* [online]. [vid. 2019-01-24]. Dostupné z: <https://www.systemonline.cz/sprava-it/cmmi-model-hodnoceni-vyspelosti-procesu-1.htm>
- [3] ERIK VAN VEENENDAAL. *Test Maturity Model integration (TMMi)*. B.m.: TMMi Foundation. 2012
- [4] PEFFERS, Ken, Tuure TUUNANEN, Marcus A. ROTHENBERGER a Samir CHATTERJEE. *A Design Science Research Methodology for Information Systems Research* [online]. prosinec 2007. Dostupné z: <https://www.tandfonline.com/doi/full/10.2753/MIS0742-1222240302>
- [5] VAN VEENENDAAL, Erik. *TMMi in the Agile world* [online]. B.m.: TMMi Foundation. 2018. Dostupné z: <https://tmmi.org/tm6/wp-content/uploads/2019/01/TMMi-in-the-Agile-world-V1.2.pdf>
- [6] LISA CRISPIN a JANET GREGORY. *Agile Testing: A Practical Guide for Testers and Agile Teams*. B.m.: Pearson, 2009. ISBN 978-0-321-53446-0.
- [7] DOŠEK, Tomáš. *Modely zlepšování procesů testování softwaru a zajištění kvality*. B.m.: Diplomová práce. Vysoká škola ekonomická v Praze, Fakulta informatiky a statistiky. Vedoucí práce Mgr. Anna Borovcová, 2012.
- [8] FRANTIŠ, Michal. *Řízení kvality softwaru ve společnosti XY*. B.m.: Diplomová práce. Univerzita Tomáše Bati ve Zlíně, Fakulta managementu a ekonomiky, Vedoucí práce doc. Ing. Petr Briš, CSc., 2015.
- [9] BARDIOVSKÁ, Petra. *Procesní přístup k testování software - Diplomová práce*, Vysoká škola ekonomická v Praze, Fakulta informatiky a statistiky. 2017, 92.
- [10] MARTIN HOMOLA. *Zvyšovanie vyspelosti procesov testovania v agilných projektoch – kvalitatívny empirický výskum*. B.m.: Diplomová práce. Vysoká škola ekonomická v Praze, Fakulta informatiky a statistiky. Vedoucí práce Ing. et Ing. Michal Doležel, 2017.
- [11] MARTIN KUČERA. *Zvyšování zralosti testovacích procesů v IT společnosti*. B.m.: Diplomová práce. Vysoká škola ekonomická v Praze, Fakulta informatiky a statistiky. Vedoucí práce Ing. Alena Buchalceová, Ph.D., 2009.
- [12] FARID, Ahmed Bahaa, Enas M. FATHY a Mahmoud Abd. ELLATIF. *Towards Agile Implementation of Test Maturity Model Integration (TMMI) Level 2 using Scrum Practices*. *International Journal of Advanced Computer Science and Applications* [online]. 2015, 6(9) [vid. 2018-10-11]. ISSN 21565570, 2158107X. Dostupné z: [doi:10.14569/IJACSA.2015.060931](https://doi.org/10.14569/IJACSA.2015.060931)
- [13] VAHID GAROUSI, MICHAEL FELDERER a TUNA HACALOĞLU. *Software test maturity assessment and test process improvement: A multivocal literature review*.

- Information and Software Technology* [online]. 2017, 85, 16–42. ISSN 09505849. Dostupné z: doi: 10.1016/j.infsof.2017.01.001
- [14] GALIN, Daniel. *Software quality assurance: From Theory to Implementation*. Harlow, England ; New York: Pearson Education Limited, 2004. ISBN 978-0-201-70945-2.
 - [15] ALENA BUCHALCEVOVÁ. *Zlepšování procesů při budování informačních systémů*. B.m.: Oeconomica, 2018. ISBN 978-80-245-2235-7.
 - [16] BEDIR TEKINERDOGAN, JOHN GRUNDY, NOUR ALI a RICHARD M. *Software Quality Assurance*. B.m.: Morgan Kaufmann, 2015. ISBN 978-0-12-802541-3.
 - [17] J. D. MUSA a W. W. EVERETT. *Software-reliability engineering: technology for the 1990s*. B.m.: IEEE Software. 1990
 - [18] TIAN, Jeff. *Software Quality Engineering: Testing, Quality Assurance, and Quantifiable Improvement* [online]. Hoboken, NJ, USA: John Wiley & Sons, Inc., 2005 [vid. 2018-08-31]. ISBN 978-0-471-72232-8. Dostupné z: doi:10.1002/0471722324
 - [19] ČSN ISO/IEC 9126-1:2001 - *Softwarové inženýrství - Hodnocení produktu - Část 1: Model jakosti*. 2002
 - [20] ČSN ISO/IEC 12119 (369022) *Informační technologie. Softwarové balíky. Požadavky na jakost a zkoušení*. 1996
 - [21] *ISO/IEC 25010:2011 Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) -- System and software quality models*. 2011
 - [22] ROUDENSKÝ, Petr a ANNA HAVLÍČKOVÁ. *Řízení kvality softwaru*. 2013. ISBN 978-80-251-3816-8.
 - [23] PETR ZÁVODSKÝ. *Rozdíly mezi testováním, řízením/kontrolou kvality a zajištěním/prokazováním kvality | Blog zaměstnanců CZ.NIC* [online]. 2015 [vid. 2019-04-17]. Dostupné z: <https://blog.nic.cz/2015/07/07/rozdily-mezi-testovanim-rizenimkontrolou-kvality-a-zajistenimprokazovanim-kvality/>
 - [24] *Řízení kvality (Quality Management) - ManagementMania.com* [online]. [vid. 2018-09-13]. Dostupné z: <https://managementmania.com/cs/rizeni-kvality>
 - [25] HLAVA, Tomáš. *Statické a dynamické testy | Testování softwaru* [online]. 2011 [vid. 2017-01-09]. Dostupné z: <http://testovanisoftwaru.cz/metodika-testovani/druhy-typy-a-kategorie-testu/staticke-a-dynamicke-testy/>
 - [26] HLAVA, Tomáš. *černá Skříňka | Testování softwaru* [online]. 2011 [vid. 2017-01-10]. Dostupné z: <http://testovanisoftwaru.cz/tag/cerna-skrinka/>
 - [27] *Software Testing – Types of Testing - software_testing_types.pdf* [online]. [vid. 2017-01-23]. Dostupné z: https://www.tutorialspoint.com/software_testing/pdf/software_testing_types.pdf
 - [28] COEPD. *What is FURPS+? Business Analyst Training in Hyderabad – COEPD* [online]. 5. srpen 2014 [vid. 2018-09-07]. Dostupné

z: <https://businessanalysttraininghyderabad.wordpress.com/2014/08/05/what-is-furps/>

- [29] *What is Agile Testing | QAComplete* [online]. [vid. 2019-04-12]. Dostupné z: <https://qacomplete.com/resources/articles/what-is-agile-testing/?q=agile>
- [30] KENT BECK, MIKE BEEDLE, ARIE VAN BENNEKUM, ALISTAIR COCKBURN a A SPOL. *Manifest Agilního vývoje software* [online]. 2001 [vid. 2019-01-20]. Dostupné z: <http://agilemanifesto.org/iso/cs/manifesto.html>
- [31] KRÁTKÝ, Tomáš. Software process (improvement). 2011, 27.
- [32] DOLEŽEL, Michal. Řízení kvality softwaru Proces testování, Přednášky kurzu 4IT446. 2017, 40.
- [33] DOLEŽEL, Michal. Řízení kvality softwaru Úvod. 2016, 57.
- [34] FONTANA, Rafaela Mantovani, Isabela Mantovani FONTANA, Paula Andrea DA ROSA GARBUIO, Sheila REINEHR a Andreia MALUCELLI. Processes versus people: How should agile software development maturity be defined? *Journal of Systems and Software* [online]. 2014, 97, 140–155. ISSN 01641212. Dostupné z: doi: 10.1016/j.jss.2014.07.030
- [35] GARDLÍK, Peter a Miroslav NOVÁK. Mezinárodní norma ISO/IEC 15504. 2015, 32.
- [36] KOOMEN T. a POL M. *IMPROVEMENT OF THE TEST PROCESS using TPI* [online]. B.m.: Sogeti. 1998. Dostupné z: https://itq.ch/pdf/tpi/tpi_uk.PDF
- [37] *TestSPICE - intacs.info* [online]. [vid. 2018-12-02]. Dostupné z: <https://www.intacs.info/index.php/testspice>
- [38] BLASCHKE, Monique, Michael PHILIPP a Tomas SCHWEIGERT. Test process assessments follow in the footsteps of software process assessments. 2009, 9.
- [39] TMMI FOUNDATION. *About TMMi Assessment Method – TMMi* [online]. [vid. 2018-10-28]. Dostupné z: <https://www.tmmi.org/about-tmmi-assessment-method-tam/>
- [40] TMMI FOUNDATION. *Service Providers – TMMi* [online]. 2018 [vid. 2018-11-16]. Dostupné z: <https://www.tmmi.org/service-providers/>
- [41] TMMI FOUNDATION. *TMMi Foundation Fees 2017* [online]. 2017. Dostupné z: <http://www.tmmi.org/wp-content/uploads/2016/11/TMMi-Foundation-Fees-2017-v1.1.pdf>
- [42] ANDREW GOSLIN a BRIAN WELLS. *TMMi Assessment Method Application Requirements*. B.m.: TMMi Foundation. 2014
- [43] ERIK VAN VEENENDAAL. *TMMi® in the Agile Era* [online]. B.m.: EuroSTAR, 2018. Dostupné z: http://www.erikvanveenendaal.nl/site/wp-content/uploads/2018_eBook_TMMi_In_The_Agile_Era.pdf
- [44] ERIK VAN VEENENDAAL. *Test Process Improvement and Agile: Friends or Foes?* [online]. B.m.: Testing Experience. 2014. Dostupné z: http://www.erikvanveenendaal.nl/NL/files/testingexperience27_09_14_van_Veenendaal.pdf

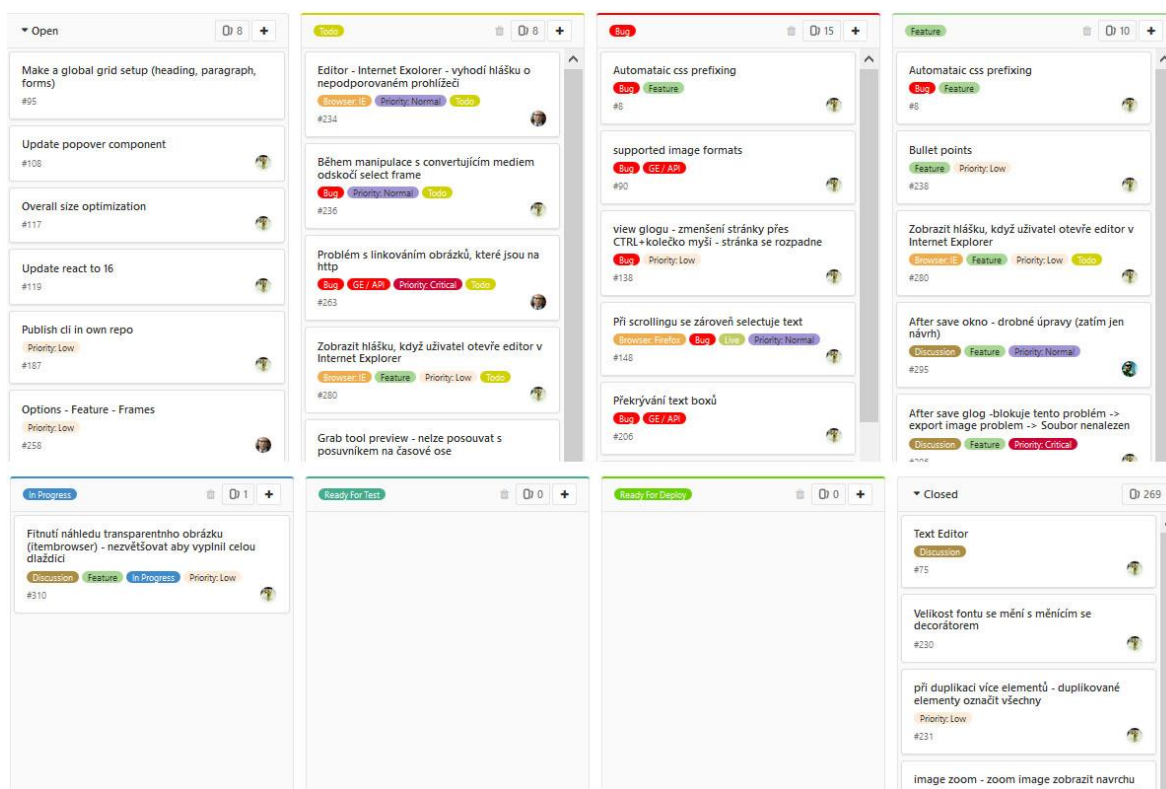
- [45] TMMI FOUNDATION. *Certified Organizations – TMMI* [online]. 2018 [vid. 2018-11-16]. Dostupné z: <https://www.tmmi.org/accredited-certifications/>
- [46] ROMAN BOŘÁNEK. *Flash definitivně skončí v roce 2020, dohodlo se Adobe s prohlížeči - Root.cz* [online]. 2017 [vid. 2019-01-27]. Dostupné z: <https://www.root.cz/clanky/flash-definitivne-skonci-v-roce-2020-dohodlo-se-adobe-s-prohlizeci/>
- [47] SUDARSANAM S.K. *Software Test Process Assessment Methodology* [online]. 2013 [vid. 2018-12-29]. Dostupné z: https://www.researchgate.net/publication/265473397_Software_Test_Process_Assessment_Methodology
- [48] BRIS, Petr, Michal FRANTIS a Monika KOLKOVA. SOFTWARE QUALITY CONTROL WITH THE USAGE OF IDEAL AND TMMI MODELS. *MM Science Journal* [online]. 2015, **2015**(04), 799–807. ISSN 18031269, 18050476. Dostupné z: doi:10.17973/MMSJ.2015_12_201565
- [49] ARAÚJO, Adailton F, Cássio L RODRIGUES, Auri M R VINCENZI, Celso G CAMILO a Almir F SILVA. A Framework for Maturity Assessment in Software Testing for Small and Medium-Sized Enterprises. 2013, 6.
- [50] BURNSTEIN, Ilene. *Practical software testing: a process-oriented approach*. New York: Springer, 2003. Springer professional computing. ISBN 978-0-387-95131-7.
- [51] TMAP.NET. *Risk Poker | TMap* [online]. [vid. 2019-04-13]. Dostupné z: <http://www.tmap.net/wiki/risk-poker>
- [52] JAMES HARVEY. *Agile Testing Estimation – TestLodge Blog* [online]. 2018 [vid. 2019-03-17]. Dostupné z: <https://blog.testlodge.com/agile-testing-estimation/>
- [53] ČSN ISO/IEC/IEEE 29119-3 36 9002, *Softwarové a systémové inženýrství – Testování softwaru – Část 3: Dokumentace testování*. 2015

Přílohy

Příloha A: Kanban tabule využívaná společností Glogster

Zde se nachází náhled Kanban tabule, která je vytvořena v nástroji Gitlab. Tato tabule je využívána ve společnosti Glogster pro přehledné vyobrazení položek produktového backlogu. Jednotlivé sloupce představují stav, nebo povahu jednotlivých položek:

- Open – Otevřené položky, které zatím nebyly naplánovány k vyřešení
- Todo – Položky připravené a naplánované k vyřešení
- Bug – Nalezený defekt, který je třeba opravit
- Feature – Jedná se o novou funkci
- In progress – Na položce se právě pracuje
- Ready For Test – Položka je připravena k otestování
- Ready For Deploy – Položka je připravena na vydání
- Closed – Zavřené/Dokončené položky



Tabulka 25 - Kanban společnosti Glogster

Příloha B: Detail karty položky v produktovém backlogu

Obrázek znázorňuje vzorový příklad položky produktového backlogu, která je vedena v nástroji GitLab ve společnosti Glogster. Ve společnosti jsou jednotlivé položky vytvářeny v podobě User Stories nebo Use Cases. Jejich podoba není ve firmě nijak formalizována. Jediným cílem je, aby nesla všechny potřebné informace, které jsou nutné ke splnění dané položky. Dodatečné informace se dotvářejí v průběhu vývoje a připisují se formou komentářů k dané položce.

The screenshot shows a GitLab issue card for the item "Nová funkce EMBED do sekce ADD". The card is titled "Nová funkce EMBED do sekce ADD" and is marked as "Closed". It was opened 10 months ago by Dominik Veselý. The card contains a description of the feature, a user story, and a design mockup. The design mockup shows a toolbar with options like "Link", "Upload", "Grab", "Embed", "Web Picker", "YouTube", and "flickr Images". Below the toolbar is a section for "EMBED ANY CONTENT FROM WEB" with a text input field and a "Recommended sources" section. The right sidebar contains various metadata fields such as "Assignee", "Epic", "Milestone", "Time tracking", "Due date", "Labels", "Weight", "Confidentiality", "Lock issue", "4 participants", "Notifications", and "Reference".

HTML editor > react > Issues > #254

Closed Opened 10 months ago by Dominik Veselý Reopen issue New issue

Nová funkce EMBED do sekce ADD

Další možnost jak lidé mohou přidávat content do glogů.

Uživatel vloží embed kod a content půjde vložit na canvas

Návrh:

↑ ADD T TEXT ★ GRAPHICS IMAGE WALL AUDIO VIDEO DATA

Link Upload Grab Embed Web Picker YouTube flickr Images

EMBED ANY CONTENT FROM WEB

INSERT your embed code and press ENTER

Recommended sources

PhET INTERACTIVE SIMULATIONS HostMath The Equation Editor Google docs OneDrive

? = nápověda, jak postupovat a získat z dané služby embed kod

@martin.santorcl @patrik.prepsl @filip.subr @pionl

Todo Přidat úkol >

Assignee Dominik Veselý @Cowok Edit

Epic This feature is locked. Upgrade plan

Milestone None Edit

Time tracking No estimate or time spent

Due date No due date Edit

Labels Discussion Prelive Ready For Deploy Edit

Weight This feature is locked. Upgrade plan

Confidentiality Not confidential Edit

Lock issue Unlocked Edit

4 participants

Notifications

Reference: html-editor/react#254