Univerzita Hradec Králové

Fakulta informatiky a managementu

Katedra informačních technologií

Vývoj softwaru pomocí agilních metodik

Bakalářská práce

Autor: Daniel Hanák
Studijní obor: Informační management

Vedoucí práce: prof. Ing. Vladimír Bureš, Ph.D., MBA

Odborný konzultant: Ing. Roman Bělonohý (Head of Development, Quadient)

Hradec Králové březen 2024

Prohlášení:

Prohlašuji, že jsem bakalářskou práci zpracoval samostatně a s použitím uvedené literatury.

V Hradci Králové dne 21.3.2024 Daniel Hanák

Poděkování:

Děkuji vedoucímu bakalářské práce prof. Ing. Vladimíru Burešovi, Ph.D., MBA za metodické vedení práce, praktické rady a zkušenosti, které byli velice užitečné a motivační. Zároveň děkuji Ing. Romanu Bělonohému za čas a ochotu při odborných konzultacích. Největší poděkování patří mojí ženě, její rodině a mým dvěma synům za trpělivost, podporu a nekonečnou přízeň, která mi byla oporou v každém okamžiku mého studia.

Abstrakt

Tato bakalářská práce se zaměřuje na aplikaci agilních metodik ve vývoji softwaru v korporátním prostředí společnosti Quadient. Cílem práce je analyzovat, jak jsou agilní principy a praktiky přizpůsobeny a integrovány do procesů této společnosti, která se specializuje na vývoj cloudových řešení. Výzkum ukazuje, že Quadient adaptuje tradiční agilní role a procesy k vyhovění specifickým potřebám svého podnikání, přičemž stále zachovává klíčové aspekty agilní flexibility a inovace. Práce poskytuje detailní pohled na různé aspekty agilního vývoje v Quadientu, včetně integrace testerů do vývojových týmů, flexibility v řešení úkolů a adaptace běžných agilních postupů, jako jsou groomingy, standupy a sprintové retrospektivy. Tato adaptace je zkoumána v kontextu zachování efektivity a vyhovění interním procesům a požadavkům. Dále práce zdůrazňuje důležitost pružnosti a schopnosti rychle reagovat na měnící se tržní podmínky a potřeby zákazníků, což je klíčové pro udržení konkurenceschopnosti v dynamickém softwarovém průmyslu. Výsledky práce představují komplexní přehled o adaptaci a implementaci agilních metodik ve společnosti Quadient, a nabízí ucelený pohled na to, jak lze agilní principy efektivně uplatňovat v korporátním prostředí.

Klíčová slova: agilní metodiky, vývoj softwaru, korporátní prostředí, adaptace procesů, cloudová řešení, efektivita a inovace

Abstract

Title: Agile Software Development

This bachelor thesis focuses on the application of agile methodologies in software development within the corporate environment of Quadient. The aim of the study is to analyze how agile principles and practices are adapted and integrated into the processes of this company, which specializes in the development of cloud solutions. The research reveals that Quadient adapts traditional agile roles and processes to meet the specific needs of its business, while still retaining key aspects of agile flexibility and innovation. The work provides a detailed view of various aspects of agile development in Quadient, including the integration of testers into development teams, flexibility in task resolution, and the adaptation of common agile practices such as groomings, stand-ups, and sprint retrospectives. This adaptation is examined in the context of maintaining efficiency and meeting internal processes and requirements. Furthermore, the thesis emphasizes the importance of agility and the ability to quickly respond to changing market conditions and customer needs, which is crucial for maintaining competitiveness in the dynamic software industry. The findings of the work present a comprehensive overview of the adaptation and implementation of agile methodologies in Quadient, offering a holistic view of how agile principles can be effectively applied in a corporate environment.

Key words: agile methodologies, software development, corporate environment, process adaptation, cloud solutions, efficiency and innovation

Obsah

1 Úvod

Bakalářská práce se zabývá agilními metodikami vývoje softwaru, což jsou přístupy, které se staly zásadními v moderním softwarovém inženýrství. Pojem „agilní“ v tomto kontextu odkazuje na schopnost být rychlý, obratný a schopný rychle a efektivně reagovat na změny nebo nové příležitosti. Agilní metodiky, jako je Scrum, Kanban nebo Extreme Programming, mají své kořeny v anglicky mluvícím prostředí, což se odráží v jejich terminologii. Mnoho termínů používaných v agilních metodikách je těsně spjato s jejich anglickými názvy, které často nemají přímý český ekvivalent.

V rámci této práce bylo rozhodnuto zachovat originální anglické termíny tam, kde překlad není jednoznačný nebo by mohl vést k nedorozuměním. Například pojem „scrum master“ je v kontextu agilních metodik specifický pojem, pro který není v češtině přesný ekvivalent. Na druhou stranu, některé termíny, jako „task“, mají běžně používaný český ekvivalent – v tomto případě „úkol“ nebo „úloha“.

V textu se tedy lze setkat s mixem českého i anglického jazyka. U každého termínu, kde je to považováno za vhodné, je poskytnut v závorce český překlad nebo významový ekvivalent, aby byla terminologie co nejpřístupnější i pro čtenáře, kteří nejsou zcela obeznámeni s anglickým jazykem. Běžně používané české protějšky jsou použity bez matoucího nebo komplikovaného překladu. Tento přístup umožňuje udržet odbornou přesnost, zatímco zároveň poskytuje čtenářům kontext a lepší porozumění tématu.

Jak již bylo zmíněno, agilní metodiky se hojně používají v oblasti vývoje softwaru, který prošel během posledních desetiletí dramatickými změnami, přičemž agilní metodiky hrají klíčovou roli v této evoluci. Tradiční přístupy k vývoji softwaru, které byly populární v druhé polovině 20. století, se často ukázaly jako nedostatečně flexibilní a neefektivní v dynamickém a neustále se měnícím technologickém prostředí. V reakci na tyto výzvy byly vytvořeny agilní metodiky, které zahrnují různé přístupy, jako je Scrum, Extreme Programming a Kanban. Tyto metody se zaměřují na iterativní vývoj, týmovou spolupráci a adaptabilitu, což umožňuje týmům rychleji reagovat na změny požadavků zákazníka a tržních podmínek.

V průběhu let se agilní praktiky adaptovaly a rozšířily v mnoha průmyslových odvětvích, nejen v IT. Agilní metody transformují pracovní procesy, kulturu a mindset ve firmách, od startupů po velké korporace. Diskutuje se často o výzvách, které přináší

implementace agilních metod, jako jsou změny v organizační struktuře, potřeba kontinuálního vzdělávání a překonávání odporu proti změnám.

Vývoj agilních metodik byl značně ovlivněn Agilním manifestem (Beck et al. 2001), který definoval klíčové hodnoty a principy agilního vývoje softwaru. Tyto principy zdůrazňují důležitost lidí a interakcí nad procesy a nástroji, fungující software nad vyčerpávající dokumentací, spolupráci se zákazníkem nad smluvním vyjednáváním a přizpůsobení se změnám nad následováním pevného plánu. Tento základní soubor hodnot a principů umožnil vývojářům a organizacím přistupovat k vývoji softwaru zcela novým a efektivnějším způsobem.

Agilní metodiky představují revoluční přístup ve vývoji softwaru, který klade důraz na flexibilitu, týmovou spolupráci a schopnost rychle reagovat na změny, uvádějí Šochová a Kunce (2019). Tyto metody jsou zvláště užitečné v dynamickém a neustále se měnícím prostředí IT projektů, což dokládá i jejich rostoucí popularita v různých odvětvích.

Scrum, jeden z nejpopulárnějších agilních rámů, se vyznačuje svou strukturovaností a pružností, uvádí Myslín (2016). Skrze krátké iterativní cykly práce, známé jako sprinty, Scrum umožňuje týmům efektivně reagovat na změny a kontinuálně se zlepšovat. Jeho klíčové prvky jako role, artefakty a schůzky jsou základem pro efektivní a transparentní práci týmu.

Agilní přístupy se dle Doležala (2022) aplikují ve škále kontextů od malých startupů po velké korporace. Jejich adaptabilita a zaměření na rychlou dodávku hodnoty zákazníkům činí agilní metodiky ideálními pro projekty, kde je potřeba rychle reagovat na tržní a technologické změny.

V kontextu agilního vývoje je klíčové pochopit, že agilita není pouze o procesech a pravidlech – je to spíše o přepracování myšlenkových vzorců a adaptaci na nové organizační prostředí. Agilní přístup znamená zavrhnutí nefunkčních přístupů a transformaci nejen v oblasti hierarchie či firemní kultury, ale i v sociálních interakcích a sebepojetí. Tyto změny se dotýkají nejen pracovního, ale i osobního života, ovlivňují náš přístup k agilnímu myšlení, jeho přijetí a zapracování do naší identity. Porozumění agilní filozofii a její adaptace na jedinečné charakteristiky jednotlivců a organizací je esenciální, neboť co funguje v jedné společnosti, nemusí nutně platit pro jinou.

2 Cíl a metodika práce

Cílem této bakalářské práce je poskytnout čtenářům komplexní přehled o agilních metodikách, jejich klíčových principech a terminologii, s důrazem na propojení teorie s praxí. Pro hlubší porozumění aplikaci agilních principů ve skutečném podnikovém prostředí byla práce pravidelně konzultována s manažerem vývojového oddělení Quadientu, Romanem Bělonohým, prostřednictvím systematických sezení jednou za dva týdny, celkem šestkrát. Tyto konzultace zahrnovaly diskuse o konceptu, implementaci a revizi materiálů, včetně obrázků, které v některých případech obsahují zakryté části k ochraně citlivých informací Quadientu a jeho partnerů. Tato metodika umožňuje čtenářům nejen porozumět základním konceptům, ale také vidět, jak jsou tyto principy aplikovány ve skutečných podnikových prostředích, přičemž zachovává etické standardy ochrany informací.

První část práce představuje teoretický rámec agilního přístupu, včetně jeho historického vývoje, základních principů a rozdílů oproti tradičním metodám vývoje softwaru. Tato část rozebírá klíčové agilní hodnoty, jako jsou adaptabilita, pružnost, týmová spolupráce a neustálé zlepšování a také popisuje fundamentální terminologii. V rámci této terminologické části jsou popsány také rozličné agilní metodiky, jako je Scrum, Kanban a Lean. Dále je poskytnuto srovnání mezi agilními a tradičními přístupy, jako je vodopádový model, přičemž je zdůrazněna schopnost agilních metod efektivně reagovat na potřeby rychlé adaptace a iterativního vývoje v dynamickém prostředí současného softwarového inženýrství. Je zdůrazněno, že zatímco tradiční modely často vyžadují podrobné plánování a mají rigidní strukturu, agilní metodiky nabízejí větší flexibilitu a zaměřují se na iterativní a inkrementální vývoj, který umožňuje efektivnější řešení nejistot a rychlých změn v požadavcích.

Navazující části práce je pozornost zaměřena na propojení teorie s praktickými příklady prostřednictvím případových studií. Tyto studie poskytují čtenářům detailní pohled na aplikaci agilních metod v reálných projektech a ukazují, jak jsou agilní principy implementovány různými organizacemi a jaké výsledky jsou díky nim dosahovány. Tato sekce slouží jako most mezi teoretickým porozuměním a praktickým uplatněním agilních principů. V rámci této části je kladen důraz na rozmanitost přístupů a strategií, které organizace používají při zavádění agilních metod. Analyzovány jsou specifické výzvy a příležitosti, s nimiž se organizace setkávají

a způsoby, jakými tyto výzvy řeší pomocí agilních principů. Dále je v této části práce ukázáno, jak agilní metody přispívají k lepší komunikaci a spolupráci v týmech, zlepšení odpovědnosti a transparentnosti a jak efektivně podporují neustálé zlepšování produktů a procesů. Zvláštní pozornost je věnována i tomu, jak se díky agilním metodám mění role a odpovědnosti jednotlivých členů týmu a jak tyto změny ovlivňují celkovou dynamiku a výkonnost týmu. Tato mezivrstva tedy poskytuje ucelený pohled na praktické aplikace agilních metod, zdůrazňuje jejich flexibilitu a přizpůsobivost a demonstruje, jak agilní přístupy přinášejí hmatatelné výhody v různých projektech a organizacích.

V závěrečné, praktické části práce je zkoumána aplikace agilních metodik v konkrétní korporaci. Analyzováno je, jak mohou být teoretické znalosti agilního přístupu aplikovány v reálném podnikovém prostředí, počínaje definicí hodnoty pro zákazníka, přes vývoj funkcionality s reálným přínosem, tvorbu testovací strategie, doručení požadovaného produktu, až po konečnou údržbu konkrétní části softwaru. Cílem této části je identifikovat klíčové faktory úspěchu a potenciální výzvy, s důrazem na sledování životního cyklu požadavku a na způsoby, jakými se v agilním prostředí formují sociální vztahy a pracovní kultura. Zvláštní pozornost je věnována tomu, jak agilní praktiky ovlivňují zapojení zaměstnanců, motivaci a spokojenost v práci a jak přispívají k vytváření proaktivního a adaptabilního pracovního prostředí. Dále je zkoumána integrace agilních metodik s dalšími oblastmi podnikového řízení, jako je řízení kvality, řízení rizik a inovační management. Ukazuje se, jak se agilní přístupy prolínají s těmito oblastmi a jak mohou být efektivně využívány pro dosažení vyšší celkové výkonnosti a konkurenceschopnosti organizace.

Nakonec se práce věnuje analýze běžných výzev a překážek, které se objevují při implementaci agilních metodik a navrhuje možná řešení těchto problémů. Tato část poskytuje návrhy na zlepšení a strategie pro překonání překážek, čímž přispívá k hlubšímu porozumění a lepšímu uplatnění agilních metod v praxi.

3 Teoretická část

3.1 Historie

Kořeny agilních metodik lze sledovat hluboko do historie, ačkoliv první jasně artikulované použití vědecké metody, které můžeme považovat za předchůdce agilního přístupu, představil Francis Bacon v roce 1620. Vývoj agilních metodik však nabral na významu ve 30. letech 20. století, kdy Walter Shewhart z Bell Labs aplikoval na produkty a procesy cykly neustálého zlepšování (Montgomery a Borror 2017; Saier 2017), obsahující fáze specifikace, výroby a inspekce. Tento přístup, který Rigby a kol. (2020) popisují, může být považován za realističtější začátek agilního myšlení. Rozvoj agilních metod pokračoval různými směry, přičemž každý z nich přispěl k dnešní rozmanité a bohaté krajině agilních praktik.

Historie agilních metod začíná v 80. letech s různými přístupy k inkrementálnímu vývoji softwaru, jako je Boehmův spirálový model (Turner 2004; Alzamil 2019). Tento model byl založen na řízení rizik a rozděloval vývojový proces do několika cyklů. V polovině 80. let se objevil Rapid Iterative Production Prototyping „RIPP“ od Du Pont Information Engineering Association, popisuje Buragga a Jhanjhi (2013), který vedl k metodě Rapid Application Development „RAD“ Jamese Martina (Weber 1999; Larman 2004; Berger a Beynon-Davies 2009). RAD položil podle Abbase a kol. (2008) základy pro Dynamic Systems Development Method „DSDM“ a inspiroval další vývoj v oblasti agilních metod. Vznik Agilního manifestu (Beck et al. 2001) přinesl různé hybridní přístupy, kombinující tradiční a agilní metody, jak uvádí Hohl a kol. (2018), například Water-Scrum-Fall, který kombinuje Scrum a vodopádový model.

Dokument od Jianga a Eberleina (2009) poskytuje hluboký pohled do společných kořenů agilního vývoje a tradičních přístupů v softwarovém inženýrství a odhaluje jeho vývoj z Lean praktik. Zde jsou tři klíčové oblasti, které poskytují přehled o základních charakteristikách tradičního a agilního softwarového inženýrství.

> **Historie a vývoj agilního přístupu** – původ termínů „agilní“ a „obratnost“ sahá do výrobního průmyslu roku 1991, kdy se vyvinul koncept „lean development“. Tento přístup zdůrazňoval eliminaci plýtvání, zvyšování edukace, rychlé dodávání a posilování týmů, což představuje základní stavební kameny spojující klasické a agilní metody vývoje softwaru.

Podobnosti mezi praktikami v tradičním a agilním vývoji – analýza odhaluje významné podobnosti a překryvy v používání praktik mezi tradičním softwarovým inženýrstvím a agilním vývojem. Tyto nálezy naznačují, že tyto dva přístupy by měly být vnímány jako vzájemně se doplňující, nikoli jako odlišné školy myšlení.

Principy klasického a agilního vývoje – základní principy používané jak v klasickém softwarovém inženýrství, tak v agilním vývoji se ukazují jako trvalé a aplikovatelné v různých praktikách. Tento pohled podtrhuje, že přístup k vývoji softwaru se musí přizpůsobit specifickým potřebám každého projektu, což zahrnuje možnost mírného upravení procesů i v rámci téže organizace.

Historie a vznik agilních metod můžeme vysledovat až k vývoji různých metod v oblasti softwarového inženýrství, které se začaly rozvíjet v reakci na potřebu větší adaptability v rychle se měnícím prostředí. Klíčovým momentem pro agilní filozofii bylo setkání v roce 2001 ve Snowbirdu v Utahu, kde se sešlo sedmnáct vývojářů, kteří sami sebe označili jako „organizační anarchisté“. Uvádí se (Rigby et al. 2020), že tato skupina diskutovala o adaptivnějších způsobech vývoje softwaru a vypracovala **Manifesto for Agile Software Development**.

Z Manifestu vyplynulo dvanáct principů, jak popisují Beck a kol. (2001), které původně vzešly z oblasti vývoje softwaru. Od té doby se tyto principy rozšířily do mnoha dalších průmyslových odvětví, což potvrzuje společná zpráva (Project Management Institute a Agile Alliance 2017), zdůrazňující jejich širokou aplikovatelnost a vliv na moderní přístupy k řízení projektů a vývoji produktů. Tento manifest zdůrazňoval hodnoty, jako jsou **jednotlivci a interakce nad procesy a nástroji**, **fungující software nad obsáhlou dokumentací**, **spolupráce se zákazníky nad vyjednáváním smluv** a **reagování na změny nad sledováním plánu**. Skupina zahrnovala zastánce různých přístupů, jako jsou Scrum, Extreme Programming, Crystal, Adaptive Software Development „ASD“, Feature-Driven Development „FDD“ a Dynamic Systems Development Method „DSDM“ (Fritzsche 2008; Sharma a Hasteer 2016; Bollati et al. 2017; Rigby et al. 2020). Tyto přístupy byly kolektivně označovány jako lehké rámce, což odráží jejich důraz na menší a jednodušší sady pravidel, které umožňují rychlejší adaptaci na měnící se požadavky.

Model inspirovaný Ahmedem Sidkym vyjadřuje agilní myšlení jako mindset určený hodnotami Agilního manifestu, řízený jeho principy a umožněný různými praktikami. Tento pohled na agilní myšlení popisují Mordi a Schoop (2021), stejně tak Klunder a kol. (2022). Je důležité poznamenat, že i když termín „agilní“ se stal populárním po vydání Manifestu, přístupy a techniky, které se dnes používají v projektových týmech, existovaly již mnoho let, v některých případech desetiletí, před jeho publikací. Agilní přístupy a metody jsou obecnými termíny zahrnujícími různé rámce a metody (Project Management Institute a Agile Alliance 2017), které splňují hodnoty a principy Agilního manifestu. Agilní a Kanban metoda jsou považovány za podmnožiny lean myšlení, které sdílí koncepty jako „soustředění se na hodnotu“, „malé výrobní dávky“ a „eliminace plýtvání“.

Agilní přístupy nabízejí řešení problémů, s nimiž se týmy softwarového vývoje potýkaly po generace. Hlavními sliby agilního přístupu jsou včasné dodání projektů, vyšší kvalita softwaru, dobře konstruovaný a udržovatelný kód, spokojenost uživatelů a lepší pracovní prostředí pro vývojáře. Scrum a Extreme Programming, Lean a Kanban. Tyto čtyři agilní školy myšlení se zaměřují na různé oblasti softwarového vývoje, ale mají jednu společnou věc – soustředí se na změnu myšlení týmu (Thomas a Baker 2008; Mahanta et al. 2016). Tato změna myšlení pomáhá týmu přejít od povrchního přidávání několika agilních praktik ke skutečnému zlepšení způsobu, jak software tvořit a dodávat. Agilní myšlení je souborem metod a metodologií, které pomáhají týmu efektivněji pracovat a lépe se rozhodovat. Tyto metody a metodologie zahrnují všechny oblasti tradičního softwarového inženýrství, jak tvrdí Stellman a Greene (2015), včetně projektového řízení, softwarového designu a architektury a zlepšování procesů. Každá z těchto metod a metodologií se skládá z praktik, které jsou zjednodušené a optimalizované tak, aby byly co nejjednodušší k přijetí.

3.2 Agilní metodiky

Agilní metodiky, zastoupené přístupy jako Scrum, Kanban, Extreme Programming a Lean, představují dynamický a flexibilní rámec pro vývoj softwaru. Tyto metodiky nejsou pouze souborem procesů, ale spíše představují způsob myšlení, který klade důraz na rychlou adaptaci na změny, interaktivitu a iterativní postupy. V agilním prostředí je hlavním cílem dosáhnout skutečných výsledků a flexibilních řešení, která jsou přizpůsobena potřebám zákazníků. Oba, Martin (2003) a Williams (2010), se

shodují, že agilní filozofie je reflektována v odlišné firemní kultuře a atmosféře, kde je důraz kladen na hravost a zábavný prvek práce. Tento přístup podporuje nejen efektivitu, ale i týmový duch a kreativitu.

Scrum je jednou z nejrozšířenějších agilních metodik, která se opírá o principy empirismu a štíhlého myšlení. Tato metodika zdůrazňuje důležitost týmové spolupráce a iterativního přístupu k práci, kde týmy pracují v krátkých cyklech zvaných „sprinty“. Tyto sprinty umožňují týmu průběžně se adaptovat a zlepšovat, což je klíčové pro dynamický vývoj softwaru (Dyba a Dingsoyr 2008; Doležal 2022). Důležitým aspektem Scrumu je také zapojení klienta a průběžná zpětná vazba, která zvyšuje pružnost a efektivitu celého procesu. Scrum (Batra et al. 2010; Šochová a Kunce 2019) se stal synonymem pro efektivní týmovou práci a adaptabilní plánování v softwarovém inženýrství.

Dle Schonbergera (2007) a stejně tak dle Brechnera (2015), Kanban, původem z Japonska, přistupuje k řízení pracovních procesů z hlediska vizualizace a optimalizace toku práce. Tato metodika se zaměřuje na vyvážení pracovních požadavků s dostupnými zdroji a je zvláště užitečná pro identifikaci a řešení úzkých míst v procesech (Lei et al. 2017; Ahmad et al. 2018; Doležal 2022). Kanban se tak stal důležitým nástrojem pro efektivní řízení softwarových projektů, kde je důležité průběžné zlepšování a adaptace.

Extreme Programming představuje sofistikovaný přístup, který klade důraz na sociální aspekty a týmovou spolupráci. Tento přístup se snaží opustit tradiční obranné mechanismy a podporuje otevřenost, komunikaci, jednoduchost a odvahu (Pikkarainen et al. 2008; Beck a Andres 2012). Extreme Programming je charakterizován krátkými vývojovými cykly, které poskytují rychlou zpětnou vazbu a flexibilitu v plánování. Důležitou součástí Extreme Programmingu je automatizované testování a verbální komunikace, což přispívá k efektivnímu a evolučnímu designu produktu.

Lean, jako přístup k vývoji softwaru, klade důraz na minimalizaci plýtvání a maximalizaci efektivity, poukazuje Martin (2003) a stejně tak Myslín (2016). Tento přístup je zaměřen na eliminaci neefektivních procesů a podporuje neustálé zlepšování a štíhlost operací. Lean je charakterizován jednoduchostí, efektivitou a rychlou odezvou na měnící se požadavky trhu a zákazníků. V IT kontextu, lean development

uplatňuje nástroje a pravidla podobná ostatním agilním metodám, ale zároveň přináší specifické principy a nástroje pro jejich realizaci.

Tyto metodiky společně představují široké spektrum přístupů k řízení a vývoji softwaru. Každá z nich nabízí jedinečné nástroje a techniky, ale všechny sdílejí základní agilní principy jako adaptabilitu, rychlou reakci na změny a týmovou spolupráci. Tyto přístupy reflektují potřebu flexibilního a dynamického přístupu ve světě, kde se požadavky a podmínky neustále mění a podporují tvorbu kvalitního softwaru prostřednictvím iterativního, adaptivního a spolupracujícího procesu.

3.3 Jednotlivci a týmy

V rámci agilního vývoje softwaru jsou klíčové role a dynamika v týmech rozhodující pro úspěch projektů. Tyto role a týmy jsou pečlivě navrženy tak, aby podporovaly flexibilitu, adaptabilitu a efektivitu, které jsou pro agilní přístup charakteristické.

V centru tohoto systému je role product ownera, která je nezbytná pro určení směru a priorit projektu. Osoba product ownera je zodpovědná za definování cílů produktu a určování priorit jednotlivých úkolů (McGreal a Jocham 2018; Šochová a Kunce 2019). Důležitým aspektem její role je také zajištění, aby každý sprint přinesl hodnotný produktový inkrement, který může být předložen zákazníkovi pro zpětnou vazbu. Tato role je klíčová pro zajištění, že vývoj produktu je v souladu s obchodními cíli a potřebami uživatelů.

Šochová (2017) spolu s Justicem (2021) ukazují další klíčovou roli – scrum master, který má za úkol vytvářet a udržovat vysoko-výkonné týmy. Jako facilitátor a kouč pomáhá týmu odstraňovat překážky a efektivně fungovat a to tím, že se zaměřuje na podporu samostatně organizovaného týmu a na agilní myšlení. Jeho role je zásadní pro úspěch celého agilního týmu.

Samoorganizované týmy jsou dle Todara (2019) základem agilního přístupu. Tyto týmy se vyznačují schopností adaptovat své postupy a přístupy a mají možnost samy se rozhodovat o svém postupu práce. Důležitá je důvěra a společný cíl mezi členy týmu, což je podporováno konceptem, že pokud selže jeden člen, selhává celý tým. To vytváří silný pocit spolupráce a vzájemné zodpovědnosti.

Multifunkční týmy představují další důležitý prvek v agilních metodách. Tyto týmy, jak tvrdí Gomes Silva a kol. (2022), kombinují rozličné dovednosti a odbornosti, což jim

umožňuje pracovat na komplexních úkolech a dodávat ucelené funkcionality. Schopnost multifunkčních týmů pracovat na úkolech napříč různými systémy a technologiemi je klíčová pro rychlou a efektivní práci.

Vývojové týmy v rámci Scrumu jsou samoorganizované a multifunkční, s hlavním úkolem dodávat hodnotu zákazníkovi na konci každého sprintu. Jak uvádí Keith (2020), v těchto týmech neexistují pevně dané role, což umožňuje členům pracovat společně na dosažení cílů projektu a vytvářet funkční produktové inkrementy.

Tyto role a týmy společně tvoří základní stavební kameny agilního vývoje softwaru, kde každý prvek hraje klíčovou roli v úspěchu celého projektu. Dynamika a vzájemná podpora mezi různými rolemi a týmy je zásadní pro dosažení agilních cílů, jako je adaptabilita, rychlá odezva na změny a efektivní týmová spolupráce.

3.4 Sprint a jeho struktura

V rámci agilního vývoje softwaru má struktura sprintu a jeho různé fáze zásadní význam pro úspěch projektu. Každá fáze sprintu má specifické cíle a praktiky, které podporují efektivitu a adaptabilitu týmů.

Sprint je základním stavebním kamenem agilních metodik, jako je Scrum. Podle Coopera a Sommera (2016) je definován jako fixní časový úsek, během něhož má tým dosáhnout stanovených cílů, sprint umožňuje týmu pracovat na předem určených úkolech s cílem dosáhnout konkrétních výsledků. Délka sprintu je obvykle nastavena tak, aby umožnila týmu dokončit dostatečné množství práce a následně prezentovat hotovou funkcionalitu zákazníkům.

Cíl sprintu je klíčový pro směřování práce týmu. Definuje, co tým plánuje dosáhnout během sprintu a má být orientován na hodnotu pro zákazníka, nikoli pouze na funkcionalitu, jak uvádí Cooper a Sommer (2018). Cíl sprintu je stanoven ve spolupráci mezi product ownerem a vývojovým týmem a je jedním z nejdůležitějších aspektů plánování sprintu.

Standup meeting (ranní porada) je denní setkání týmu, které slouží k rychlému sdílení informací a koordinaci činností, tvrdí Martin (2003). Tato krátká setkání zvyšují transparentnost a podporují týmovou spolupráci, což je klíčové pro rychlé identifikování a řešení problémů.

Hodnocení sprintu, často označované jako „demo“, je příležitostí pro tým prezentovat dokončenou práci na konci sprintu. Tento meeting je dle Šochové a Kunce (2019) zaměřen na akceptaci a prezentaci výsledků práce týmu a poskytuje zásadní platformu pro zpětnou vazbu od product ownera a ostatních zúčastněných stakeholderů.

Sprintová retrospektiva, zkráceně „retro“, je událost, která se zaměřuje na zlepšování procesů a způsobu práce týmu. Doležal (2022) tvrdí, že na rozdíl od hodnocení sprintu, které se zaměřuje na produkt, se retrospektiva soustředí na tým a jeho pracovní procesy. Cílem je identifikovat oblasti pro zlepšení a vytvořit plán na jejich realizaci v dalších sprintech.

Plánování sprintu je důležitou schůzkou, která významně ovlivňuje úspěšnost nadcházejícího sprintu, ukazuje Myslín (2016). Tato schůzka zahrnuje definici cíle sprintu, vytvoření sprint backlogu (seznam úkolů na daný sprint) a definici potřeb pro další jednání. Kvalitní příprava sprintu je klíčová pro jeho úspěch a je základem pro efektivní práci týmu.

Každá z těchto fází sprintu představuje nezbytný prvek v cyklu agilního vývoje, poskytující strukturu a směr, které jsou nezbytné pro úspěšné dokončení projektů. Tyto fáze společně podporují transparentnost, adaptabilitu a efektivitu týmů, což jsou klíčové charakteristiky agilního přístupu.

3.5 Práce se zadáními a požadavky

V agilních metodikách, jako je Scrum, je práce se zadáními a požadavky pečlivě strukturovaná a řízená prostřednictvím různých nástrojů a procesů, aby bylo zajištěno efektivní a cílené plnění cílů projektu.

Product backlog (seznam úkolů daného produktu) je zásadním prvkem v agilním plánování. Tento prioritizovaný seznam obsahuje položky, které mají vést k naplnění cíle vývoje produktu. Product backlog je dynamický a neustále se vyvíjí (Šochová a Kunce 2019; Doležal 2022), což umožňuje týmům efektivně řídit a prioritizovat pracovní úkoly. Je to živý dokument, který je pod neustálým dohledem a úpravou ze strany product ownera, který určuje priority jednotlivých položek.

Myslín (2016) uvádí sprint backlog jako specifický výběr úkolů z product backlogu, na kterých se tým zaměří v průběhu aktuálního sprintu. Tyto vybrané úkoly určují, co a jak bude v daném sprintu realizováno. Sprint backlog je klíčový pro plánování a úspěšnou

realizaci sprintu a vývojový tým na jeho základě určuje, jaké cíle a úlohy mají být splněny.

User story je základním formátem, jakým jsou zadání úkolů formulována v agilních metodikách. Typická user story vysvětluje, co uživatel potřebuje a jakou hodnotu to pro něj přináší. Tento způsob formulace umožňuje týmu jasně pochopit požadavky z pohledu uživatele a efektivně na nich pracovat. Podle Šochové a Kunce je základní formát user story následující *„Jako Uživatel chci Funkcionalitu, abych dostal Business Value“* (Šochová a Kunce 2019) Konkrétní příklad pak uvádějí *„Jako administrátor chci kontaktní formulář, abych se včas dozvěděl o chybách systému.“*.

Epic (epický nebo velký úkol) je používán pro velké, komplexní úkoly, které jsou příliš rozsáhlé na to, aby byly realizovány v rámci jednoho sprintu. Tyto úkoly se pak dělí na menší, konkrétnější části, obvykle ve formě user story. Epic je užitečný pro organizaci práce na větších funkcionalitách nebo projektech (Moreira 2017; Šochová a Kunce 2019). Umožňuje udržet globální kontext a koherenci mezi jednotlivými úkoly.

V rámci agilního týmu je každý úkol definován jako konkrétní kus práce nebo aktivita, která je potřebná pro dosažení cílů definovaných v product nebo sprint backlogu. Tyto úkoly jsou často vizualizovány na tabuli v metodě Kanban a jsou přiřazovány konkrétním členům týmu, jak zdůrazňuje Brechner (2015) a doplňuje Doležal (2022). Tato vizualizace a přiřazení pomáhají v organizaci a sledování pokroku práce.

Backlog grooming (úprava backlogu – seznamu úkolů) je procesem, kde tým pracuje na porozumění a upřesňování položek v product backlogu. Tento proces je klíčový pro přípravu na nadcházející sprinty (Drury et al. 2012; Šochová a Kunce 2019) a zahrnuje pravidelné meetingy pro detailní procházení a rozdělování user stories a úkolů, aby byly připraveny pro začlenění do sprintů.

Tyto různé aspekty práce se zadáními a požadavky v agilním vývoji softwaru tvoří systém, který umožňuje týmům efektivně plánovat, prioritizovat a realizovat úkoly, což vede k vytváření hodnoty a úspěchu projektů.

3.6 Definice a průběh práce

V agilním vývoji softwaru jsou definice a průběh práce nezbytné pro účinnost a průhlednost vývojového procesu. Tyto aspekty jsou klíčové pro zajištění kvality a efektivity ve vývoji produktů.

Definition of done (definice pro dokončení úkolu) známá i jako „DoD“, je zásadním pojmem v agilních metodikách, jak uvádí Šochová a Kunce (2019) a stejně tak Rigby a kol. (2020). Tato dohoda mezi vývojovým týmem a product ownerem určuje, kdy může být položka z product backlogu považována za dokončenou. DoD stanovuje specifická kritéria pro každou dokončenou funkčnost, zahrnující jak kvalitu, tak připravenost k nasazení. Díky tomu je možné zajistit, že všechny výstupy splňují stanovené standardy a jsou připraveny na hodnocení sprintu. Jak se tým vyvíjí, může se také rozšiřovat DoD, aby byla zajištěna vyšší kvalita a nasaditelnost výsledků.

Continuous integration (průběžná integrace) neboli „CI“ je proces, který umožňuje týmům efektivně spolupracovat v reálném čase. Klíčovým prvkem CI je automatické spojení „integrace“ nového kódu s existujícím kódem v projektu po každé změně (Šochová a Kunce 2019; van Merode 2023). Tento proces podporuje okamžité identifikování a opravy problémů, což vede k větší stabilitě a kvalitě celkového kódu. CI vyžaduje použití centrálního úložiště zdrojového kódu, což umožňuje týmům efektivně spravovat a testovat kód.

Continuous delivery (nepřetržité dodávání), známé jako „CD“, je další klíčovou součástí agilních procesů. CD podporuje nepřetržitý tok práce, což umožňuje inkrementální vydávání produktu, obvykle na týdenní bázi (Doležal 2022; van Merode 2023). V rámci tohoto procesu jsou všechny změny v kódu, ať už jde o nové funkce, opravy chyb, nebo experimenty, automaticky převáděny do produkčního prostředí. Tento přístup umožňuje rychlejší a častější vydávání nových verzí produktu, což zvyšuje flexibilitu a umožňuje rychle reagovat na potřeby trhu nebo zákazníka.

DoD, CI a CD jsou základními prvky, které společně tvoří rámec pro efektivní a transparentní vývoj softwaru v agilním prostředí. Tyto procesy a dohody umožňují týmům udržovat vysokou kvalitu práce a efektivně reagovat na změny a výzvy v průběhu vývojového cyklu.

3.7 Principy

Jak již bylo několikrát řečeno, dnešní rychle se měnící a vysoce konkurenční prostředí nutí organizace být schopny rychle reagovat na tržní změny a potřeby zákazníků. Agilní metodologie, která původně vznikla v kontextu softwarového vývoje, se stala klíčovým přístupem umožňujícím takovou flexibilitu a efektivitu. Základy tohoto přístupu jsou definovány v Agilním manifestu (Beck et al. 2001). Tyto principy nejenže představují

základní hodnoty a přesvědčení agilních metod, ale také poskytují rámec pro efektivnější a adaptabilnější proces vývoje.

Agilní metodologie popisuje několik principů, ale zaměřuje se na čtyři hlavní hodnoty, které podporují adaptabilitu, týmovou spolupráci, klientskou orientaci a schopnost rychle reagovat na změny. Tyto hodnoty nejsou jen abstraktními ideály, ale jsou pevně zakotveny v každodenní praxi týmů a organizací, které agilní přístupy implementují. V následujících bodech je v souladu s Beckem a kol. (2001) a zároveň se Šochovou a Kuncem (2019) podrobněji popsána každá z těchto čtyř základních hodnot.

Jednotlivci a interakce mají přednost před procesy a nástroji. Tento princip zdůrazňuje význam lidského faktoru a komunikace v týmu nad striktním dodržování procesů a používání nástrojů.

Fungující software je důležitější než vyčerpávající dokumentace. Cílem je vytvořit software, který funguje a splňuje potřeby uživatelů, nikoli vyvíjet rozsáhlou dokumentaci.

Spolupráce se zákazníkem má přednost před vyjednáváním o smlouvě. Agilní přístup klade důraz na blízkou a průběžnou spolupráci se zákazníkem, aby bylo možné lépe reagovat na jeho potřeby a požadavky.

Reagování na změny je důležitější než dodržování plánu. Agilní metodologie přijímá změny a adaptace jako součást procesu vývoje, místo striktního dodržování původně stanovených plánů.

3.8 Rozdíly s tradičním přístupem

V oblasti systémového inženýrství existuje široká škála přístupů k vývoji softwaru a systémů, z nichž každý má své specifické výhody a omezení v závislosti na kontextu projektu. Mezi tyto přístupy patří například Waterfall (vodopádový model), Spiral (spirálový model), Lean (štíhlá výroba), Critical Path Method (metoda kritické cesty) často označovaná jako „CPM“ nebo nadstavba CPM – Critical Chain Project Management, známá pod zkratkou „CCPM“ (Leach 2000; Subair 2014; Stellman a Greene 2015; Project Management Institute 2021). Každý z těchto modelů přináší unikátní perspektivu na proces tvorby softwaru a systémů, přičemž některé jsou vhodnější pro určité typy projektů než jiné.

Vodopádový model, který je jedním z nejstarších a nejznámějších přístupů, je často kritizován za svou rigiditu a lineární strukturu. Tento model předpokládá, že vývoj probíhá v pevně stanovených fázích – požadavky, návrh, implementace, ověření a údržba. Hlavní omezení tohoto přístupu spočívá v jeho předpokladu, že všechny požadavky na systém lze plně pochopit a definovat na začátku projektu, což je v praxi často nereálné. V důsledku toho může být vodopádový model neefektivní v prostředích, kde jsou časté změny požadavků nebo kde je potřeba rychle reagovat na měnící se tržní podmínky.

Agilní metodologie vznikla jako reakce na tyto a další omezení tradičních přístupů, jako je vodopádový model. Agilní přístupy, jako je Scrum nebo Kanban, se zaměřují na iterativní a inkrementální vývoj, kde je důraz kladen na pružnost, rychlou adaptaci na změny, týmovou spolupráci a kontinuální zpětnou vazbu od zákazníků. Tímto způsobem agilní metodologie poskytuje flexibilní a dynamický rámec pro řízení softwarových projektů, který je schopen se přizpůsobit a efektivně reagovat na neustálé změny v požadavcích a tržním prostředí.

Vodopádový model má své místo v projektech, kde jsou požadavky dobře definované a stabilní a kde je minimální riziko změn během vývoje. Naproti tomu, v prostředích, kde je vyšší míra nejistoty a kde je potřeba rychlé adaptace, nabízí agilní metodologie efektivnější a flexibilnější alternativu.

Tradiční vodopádový model vývoje softwaru, i když má své místo a výhody, často naráží podle Stellmana a Greena (2015), stejně tak podle Thesinga a kol. (2021), na několik významných překážek, které mohou projekt zkomplikovat nebo dokonce ohrozit jeho úspěch.

Neschopnost reagovat na změny – vodopádový model vyžaduje, aby byly všechny aspekty projektu podrobně definovány a schváleny na samém začátku. Tento rigidní přístup vede k obtížnostem při reagování na změny požadavků nebo tržních podmínek během vývoje projektu. Týmy se pak mohou ocitnout v situaci, kdy jsou nuceny dodržet zastaralé specifikace, které již neodpovídají aktuálním potřebám.

Problémy s podporou a škálovatelností – kvůli pevně stanoveným plánům a nedostatku flexibility při reakci na změny, týmy často přistupují k neoptimálním řešením a záplatám, aby splnily požadavky. Toto může vést

k problémům se škálovatelností a udržitelností softwaru v dlouhodobém horizontu, což ztěžuje jeho další rozvoj a údržbu.

Problémy se změnami požadavků – vodopádový proces je nepružný, když dojde ke změnám požadavků ze strany zákazníků nebo trhu. Týmy jsou pak často nuceny se vrátit k předešlým fázím vývoje, což je časově náročné a neefektivní. V důsledku toho může být výsledný produkt nekompatibilní s aktuálními potřebami a očekáváními.

Zpoždění a neefektivita – přísná struktura vodopádového modelu může způsobit značná zpoždění, zejména pokud analýza a návrh zabírají příliš mnoho času. Kromě toho, nepružné procesy schvalování změn mohou vést k dalším zdržením, což zvyšuje tlak na týmy, aby splnily termíny, často za cenu kompromisů v kvalitě a funkčnosti.

Tato omezení vodopádového modelu vyvolala potřebu alternativních přístupů, jako je agilní metodologie, která se zaměřuje na flexibilitu, adaptabilitu a spolupráci, aby se překonaly tyto výzvy.

Agilní metodika, ač je široce uznávána pro svou flexibilitu a efektivitu ve vývoji softwaru, samozřejmě není všespásná a v určitých situacích může narazit také na své limity. Přestože nabízí řadu výhod, je důležité si být vědom možných komplikací a omezení, která mohou při jejím používání nastat, uvádí Gligor a kol. (2015) a doplňuje Stellman a Greene (2015).

Nedostatečné pochopení agilních hodnot a principů – projekty, které používají některé agilní techniky a praktiky, ale nezachycují skutečné agilní hodnoty a principy, často narazí na stejné druhy problémů, které postihují projekty vodopádového modelu. Tento nedostatek hlubokého pochopení může vést k neúspěchu projektů, které se snaží být „agilní“ pouze na povrchu.

Pouhé přejmenování stávajících praktik bez skutečné změny – některé týmy mohou přijmout agilní metodologie pouze formálně, přičemž ve skutečnosti se jejich způsob práce nezmění. Místo toho si ponechají stávající pracovní procesy a pouze je přejmenují do agilní terminologie, což vede k neefektivnosti a neschopnosti dosáhnout skutečných výhod agilního přístupu.

Zvýšená pracovní zátěž a časový tlak – v některých případech se může zdát, že přechod na agilní metodologie vede k větší pracovní zátěži a více práce mimo

běžnou pracovní dobu, zejména když se snaží splnit všechny požadavky v časově omezených iteracích. To může vést k pocitu, že agilní přístup není zlepšením a může způsobit vyčerpání týmu.

Kulturní a filozofické rozdíly – neúspěchy agilních projektů jsou často důsledkem kulturních a filozofických rozdílů mezi tradičními (vodopádovými) a agilními metodologiemi. Mezi hlavní příčiny neúspěchu patří nedostatek zkušeností s agilními metodami, firemní filozofie v rozporu s agilními hodnotami a vnější tlak na dodržování tradičních praktik.

Tyto body poukazují na to, že úspěch agilního přístupu závisí nejen na adopci určitých praktik, ale také na hlubokém pochopení a přijetí agilních hodnot a principů, stejně jako na schopnosti kulturně a filozoficky se přizpůsobit novému způsobu práce.

4 Případové studie

4.1 Agilní vývoj

Z hlediska agilního vývoje je zajímavá studie popsaná Shorem a kol. (2022), která detailně zkoumá chování a výsledky Extreme Programming v reálných vývojových situacích. Tato studie, vedena Nancy van Schooenderwoert a dalšími odborníky, poskytuje hluboký vhled do toho, jak Extreme Programming může efektivně přispět ke zlepšení softwarového vývoje.

Studie se soustředí na několik klíčových aspektů Extreme Programming, jako jsou iterativní vývoj, intenzivní testování a refaktoring kódu a ukazuje, jak tyto praktiky mohou vést k výraznému snížení chyb a zlepšení efektivity a kvality softwaru. Zahrnuje také srovnání výsledků týmů využívajících Extreme Programming s těmi, které používaly jiné agilní metodiky, jako je Scrum, poskytujíc tak cenný pohled na výhody a omezení různých přístupů v agilním vývoji.

Nancy van Schooenderwoert a její kolegové ukázali, že správné uplatnění Extreme Programming může vést k dramatickým vylepšením ve vývojovém procesu, což je důležité pro organizace, které hledají efektivnější a adaptabilnější metody softwarového inženýrství. Studie poskytuje důkazy o úspěchu Extreme Programming v různých prostředích a zdůrazňuje důležitost agilních principů v moderním vývoji softwaru.

Snížení chyb a zvýšení efektivity – jedna z případových studií uvádí, že týmy využívající Extreme Programming dosáhly výrazného snížení počtu chyb. Z průměrných 2270 chyb na 381, což představuje 83% pokles. Kromě toho tyto týmy dokázaly dodávat výsledky o 24 % rychleji a s 39 % menším počtem zaměstnanců.

Srovnání s jinými agilními metodami – další případové studie porovnávaly výsledky týmů používajících Extreme Programming s těmi, které používaly jiné agilní metodiky, jako je Scrum. Například pomocí QSM (quantitative software management – metodologie zaměřená na kvantitativní řízení a analýzu softwarových projektů) bylo zjištěno, že tým využívající Scrum dosáhl 11% snížení chyb a 58% zkrácení harmonogramu, zatímco tým s Extreme

Programming metodikou dosáhl 75% snížení chyb a 53% zkrácení harmonogramu.

Multi-týmová analýza – v další části studie se zkoumala efektivita Extreme Programming v rámci rozsáhlejší analýzy zahrnující tisíce vývojářů v různých týmech. Zde bylo zjištěno, že týmy používající Extreme Programming dosáhly 75% snížení chyb a 30% zkrácení plánovaných časů.

4.2 Agilní nasazování

V oblasti nasazování softwaru nabízí agilní přístupy výrazné výhody, jak ukazuje studie Díaze a kol. (2018). Jednou z hlavních výhod agilních metod je jejich flexibilita a schopnost rychle reagovat na měnící se potřeby trhu a zákazníků. Tato adaptabilita je zásadní pro efektivní a pružné nasazování softwaru, což umožňuje organizacím udržovat krok s rychle se vyvíjejícím technologickým prostředím.

Agilní metody rovněž podporují těsnější spolupráci mezi vývojovými týmy a zákazníky. Tato spolupráce vede k lepšímu porozumění požadavkům a očekáváním zákazníků, což umožňuje rychlejší a přesnější vývoj softwaru. V kontextu DevOps (metodologický přístup v oblasti softwarového inženýrství, který spojuje software **dev**elopment – vývoj a IT **op**eration**s** – provoz) je tato spolupráce rozšířena také na týmy zodpovědné za provoz a nasazování softwaru, což umožňuje rychlejší a efektivnější průběh celého vývojového cyklu.

Dále studie zdůrazňuje význam měkkých dovedností, jako jsou komunikace, týmová práce a flexibilita, které jsou nezbytné pro úspěch v agilním prostředí. Týmy, které tyto dovednosti rozvíjejí, jsou schopny efektivněji spolupracovat a řešit problémy, což vede k lepší integraci a plynulosti procesů nasazování.

Výsledkem je, že přechod na agilní metody může výrazně přispět k zefektivnění nasazování softwaru, zkrácení cyklů vývoje a zlepšení celkové kvality a hodnoty softwaru pro uživatele. Tento přístup tedy nabízí podstatné výhody pro organizace, které usilují o rychlou adaptaci na měnící se tržní podmínky a potřeby zákazníků.

4.3 Agilní testování

V rámci moderních softwarových vývojových metodik hraje scénářové testování klíčovou roli, zejména v kontextu agilního vývoje. Podle Crispina a Gregoryho (2009), scénářové testování umožňuje simulaci reálných podmínek užívání softwaru

prostřednictvím specifických případů a situací. Tento přístup nejenže podporuje uživatelsky centrovaný vývoj, ale také reflektuje pružnost a adaptabilitu, které jsou zásadní pro agilní metodiku.

Agilní metodika se soustředí na flexibilní a iterativní přístup k vývoji softwaru, kde je důraz kladen na rychlou odezvu na změny a nepřetržitou zpětnou vazbu od uživatelů. Scénářové testování, jakožto integrální součást agilního procesu, zajišťuje, že software je testován v kontextu, který odráží jeho skutečné použití. Takto se odhalují chyby a nedostatky, které by mohly být při tradičních testovacích metodách přehlédnuty. Kromě toho, scénářové testování podporuje spolupráci a komunikaci mezi členy agilního týmu, což je zásadní pro úspěšný agilní vývoj.

V kontextu agilního vývoje je tedy scénářové testování neocenitelným nástrojem pro zajištění, že vývoj softwaru je nejen flexibilní a uživatelsky orientovaný, ale také zaměřený na kvalitu a spolehlivost. Tímto přístupem agilní týmy mohou efektivně reagovat na měnící se požadavky a zajišťovat, že konečný produkt splňuje očekávání uživatelů a zároveň zachovává vysokou úroveň kvality.

4.4 Agilní údržba

Výsledky studie Ibrahima a kol. (2019) ukazují, že přijetí agilní metodiky v procesu údržby softwaru má své výzvy a výhody. Mezi výzvy patří přerušované sprint plánování, nestabilní rychlost a nekonzistence burndown grafů (vizuální nástroj používaný v agilním řízení projektů, který ukazuje, jak rychle tým pracuje a dokončuje úkoly v průběhu času). Přestože agilní přístup přináší zlepšení kvality softwaru, je nutné pečlivě řešit tyto výzvy, aby bylo možné účinně aplikovat agilní principy v údržbě softwaru. Studie zdůrazňuje význam správného zapojení zákazníka, efektivní komunikace a adaptace dokumentace v kontextu agilní údržby softwaru.

Agilní techniky, známé svou flexibilitou a schopností rychle reagovat na změny, hrají klíčovou roli v údržbě softwaru. Přístup založený na agilní metodologii umožňuje týmům efektivněji spravovat a aktualizovat software, což je zásadní v dnešním rychle se měnícím technologickém prostředí. Studie Ibrahima a kol. (2019) poskytuje konkrétní příklad toho, jak lze agilní techniky uplatnit pro údržbu softwaru s velkým počtem uživatelů, což je situace, která vyžaduje nejen technickou zručnost, ale i schopnost rychle se přizpůsobit požadavkům a potřebám uživatelů.

5 Praktická část

5.1 Představení firmy

5.1.1 Popis, historie a hlavní oblasti působení

Quadient, původně známý jako Neopost, je prominentním poskytovatelem řešení pro zákaznickou komunikaci a automatizaci obchodních procesů. Klíčem k jejich úspěchu je zaměření na vytváření významných zákaznických zážitků prostřednictvím svých produktů a služeb. Tyto služby zahrnují inteligentní automatizaci komunikace, řešení pro inteligentní balíkové skříně a smart mailing a shipping řešení (Quadient 2024a).

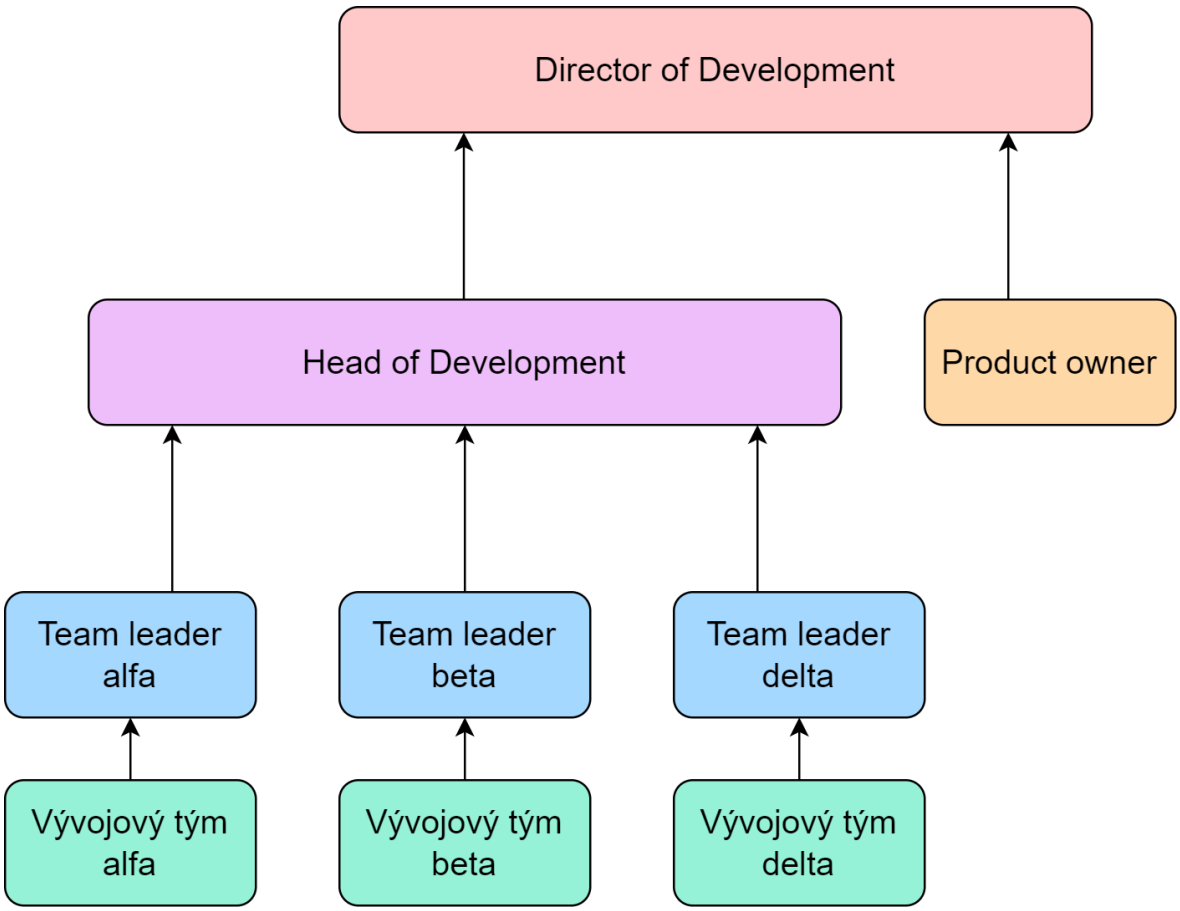
Quadient má bohatou historii, která sahá až do roku 1924. V průběhu let prošel společností řadou evolučních změn, včetně přejmenování z Neopost na Quadient, což odrazilo jejich přechod k digitálním a fyzickým komunikačním řešením. Firma má dlouhou tradici v oblasti inovací a rozvoje, což ji umožnilo udržet si silnou pozici na trhu (Craft.co 2023).

Quadient se zaměřuje na několik klíčových oblastí, které zahrnují – správu zákaznických prostředí, automatizaci obchodních procesů, řešení pro poštovní služby a řešení pro automatizované balíkové boxy. Tyto produkty a služby jsou navrženy tak, aby podporovaly firmy v různých průmyslových odvětvích, včetně finančních služeb, zdravotnictví, pojišťovnictví, telekomunikací a veřejného sektoru. Cílem Quadientu je poskytovat řešení, která zjednodušují a zefektivňují komunikaci mezi podniky a jejich zákazníky (Quadient 2024b).

Quadient se vyznačuje svým důrazem na inovace a adaptabilitu, což mu umožňuje úspěšně reagovat na neustálé změny v požadavcích trhu a potřebách zákazníků.

5.1.2 Organizační struktura

Quadient implementuje v rámci svého oddělení výzkumu a vývoje, zkráceně „R&D“, unikátní hybridní organizační strukturu, která kombinuje tradiční hierarchické prvky s agilními metodami. Jak ukazuje obrázek 1, tato struktura je rozčleněna do několika specializovaných rolí, z nichž každá se zaměřuje na konkrétní aspekty vývoje a podpory produktů.



Obrázek 1 Diagram organizační struktury. Zdroj: vlastní zpracování

Na nejspodnější vrstvě struktury jsou týmy tvořené vývojáři a testery, kteří jsou zodpovědní za bezprostřední vývoj a testování aplikací. Vedle nich stojí tým podpory, který se věnuje technické asistenci a řešení problémů spojených s provozem aplikací u zákazníků. Pro zajištění hladkého běhu aplikací v cloudu je zde tým DevOps, zatímco grafici a UX (user experience neboli uživatelsky přívětivé rozhraní aplikace) pracovníci se soustředí na vizuální a uživatelské aspekty produktů.

Nad těmito týmy stojí dokumentaristé, kteří se věnují tvorbě dokumentace a pracovníci tréninkového centra, kteří připravují vzdělávací materiály a podporují uživatelské zaškolení. Každý z těchto týmů má svého nadřízeného „team leader“, což je odlišné od typického agilního přístupu, který preferuje samoorganizované týmy bez tradičních vedoucích rolí.

Ve střední vrstvě struktury jsou manažeři, známí jako „Head of“, například „Head of Development“ nebo „Head of UX“. Tito manažeři zastřešují jednotlivé týmy a reportují vyšším úrovním managementu. Na vyšší úrovni jsou manažeři první vrstvy, interně označováni „Directors“, jako je „Director of Development“ nebo „Director of UX“, kteří

pod sebou mohou mít jednoho nebo několik manažerů střední úrovně „Headů“ a další podpůrné role. Typickou podpůrnou rolí pro manažera první vrstvy oddělení vývoje aplikace „Director of Development“ je product owner.

V kontextu organizační struktury Quadientu, zvláště v oddělení vývoje aplikací, se role product ownera výrazně liší od tradičního chápání této role v agilním rámci. V Quadientu je role product ownera přizpůsobena tak, aby byla více orientována na interakci s vývojovými týmy, na rozdíl od běžné agilní praxe, kde product owner tráví většinu času komunikací se zákazníky, aby porozuměl jejich potřebám a požadavkům. Zatímco v klasickém agilním přístupu je poměr času stráveného mezi zákazníky a vývojovými týmy obvykle 80:20 ve prospěch zákazníka, v Quadientu lze pozorovat opačný trend. Zde product owner věnuje přibližně 80 % svého času práci s vývojovými týmy a pouze 20 % času je zaměřeno na interakci se zákazníky.

Tato specifická adaptace role product owner v Quadientu je pravděpodobně reakcí na potřebu intenzivnější spolupráce a koordinace v rámci interních vývojových procesů. Tento přístup odráží unikátní provozní dynamiku a strategické cíle společnosti a ukazuje, jak mohou být agilní principy flexibilně upraveny, aby vyhovovaly specifickým potřebám a strukturám v korporátním prostředí.

Další zajímavostí v Quadientu je role technického manažera produktu „technical product manager“, jež je klíčovou komponentou v rámci organizační struktury, která přináší unikátní perspektivu do procesu vývoje. Technický manažer produktu se primárně zaměřuje na sběr a analýzu potřeb zákazníků, což je nezbytné pro efektivní plánování a rozvoj produktů. Tato role vyžaduje hluboké porozumění technickým i obchodním aspektům produktů a vyžaduje schopnost transformovat rozsáhlé požadavky zákazníků, známé jako „epics“, do konkrétních a realizovatelných požadavků.

Jedinečnost této role v Quadientu spočívá v tom, že technický manažer produktu není jen prostředníkem mezi zákazníkem a vývojovým týmem, ale aktivně se podílí na definování obrysů produktu a jeho směřování. V agilní terminologii bychom mohli pozici technického manažera produktu přirovnat k rozšířené roli tradičního product ownera, nicméně s výraznějším technickým a strategickým zaměřením.

Techničtí manažeři produktu v Quadientu reportují přímo manažerům první úrovně „Director of Technical Product Management“, což je další významný aspekt, který

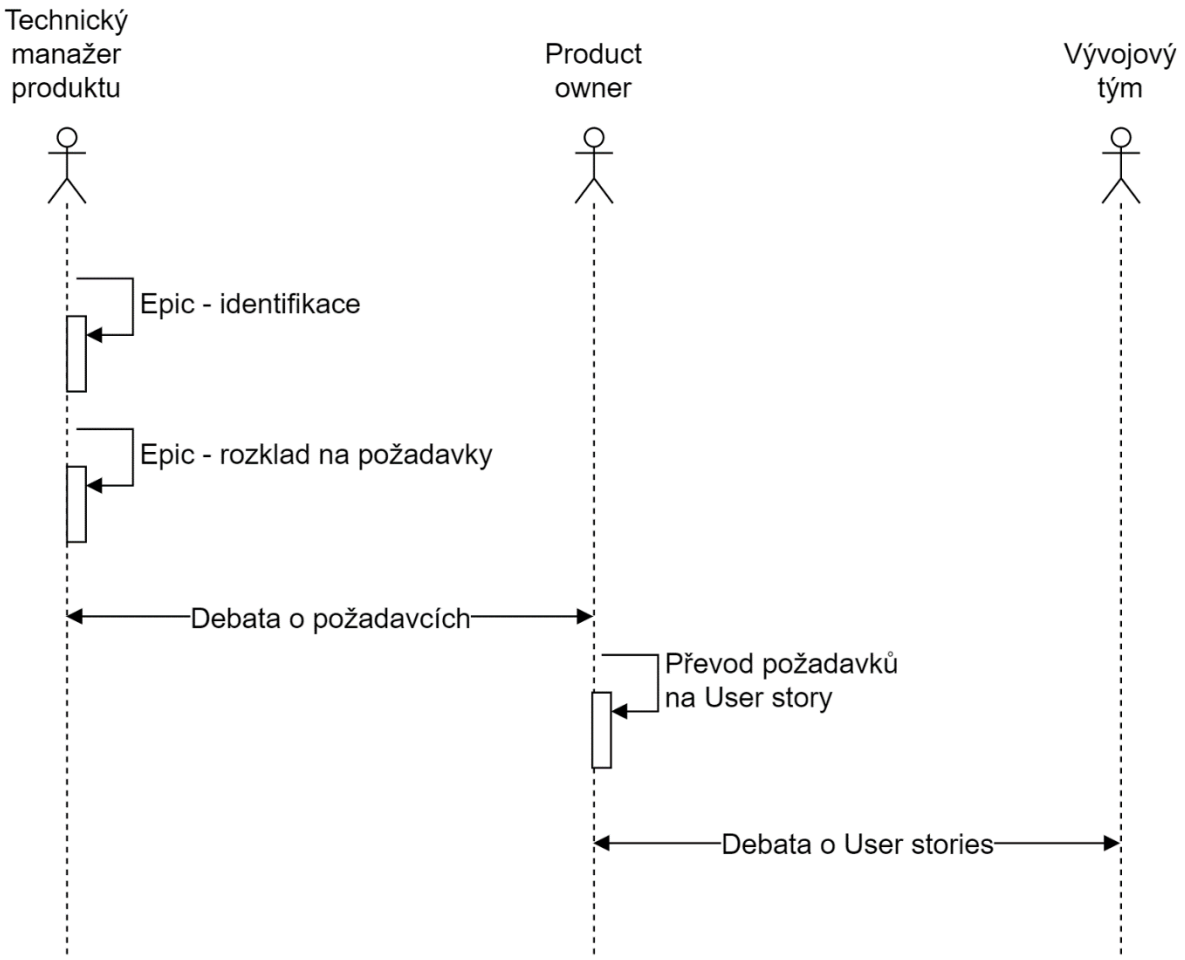
odráží důležitost a strategickou hodnotu této role. Tímto uspořádáním je zajištěno, že technické aspekty a zákaznické potřeby jsou řádně zastoupeny a integrovány do vývojového procesu. V rámci Quadientu tak technický manažer produktu představuje klíčový most mezi technologickým vývojem a obchodní strategií, což je nezbytné pro vyvážený a účelný vývoj produktů, které splňují jak technické, tak tržní požadavky.

V Quadientu, který je více zaměřen na prodej než na produkt, se proces definování a realizace produktových požadavků odvíjí od pečlivě provedeného průzkumu trhu. Tento průzkum má za cíl identifikovat příležitosti pro zvýšení tržního podílu společnosti a zachytit klíčové trendy a potřeby zákazníků, což je nezbytné pro udržení konkurenceschopnosti v dynamickém obchodním prostředí.

Role technického manažera produktu v Quadientu je zásadní v tomto procesu. Technický manažer produktu, s hlubokým porozuměním technickým i tržním aspektům, přeměňuje zjištěné epics, což jsou rozsáhlé a strategické požadavky odvozené z průzkumu trhu, na konkrétní požadavky. Tyto konkrétní požadavky jsou pak důkladně specifikovány tak, aby byly připraveny pro další fázi vývoje.

V Quadientu pak vstupuje do hry product owner, který přejímá tyto konkrétní požadavky od technických manažerů produktu. Product owner má za úkol transformovat je na user stories pro vývojové týmy. V tomto smyslu lze říci, že v Quadientu product owner funguje v interním prostředí, kde jeho „zákazníkem“ jsou pracovníci ve funkci technických manažerů produktu. Obrázek 2 ilustruje, jak tato vnitrofiremní dynamika umožňuje product owneru efektivně převádět strategické požadavky na konkrétní pracovní úkoly pro vývojové týmy, zajišťující, že vývoj produktu je v souladu s tržními požadavky a obchodními cíli společnosti.

Tento proces ilustruje, jak Quadient integruje agilní metodiky s tradičním obchodním plánováním, aby dosáhl optimalizace produktů v souladu s tržními trendy a zákaznickými potřebami.



Obrázek 2 Epic – proces rozpadu. Zdroj: vlastní zpracování

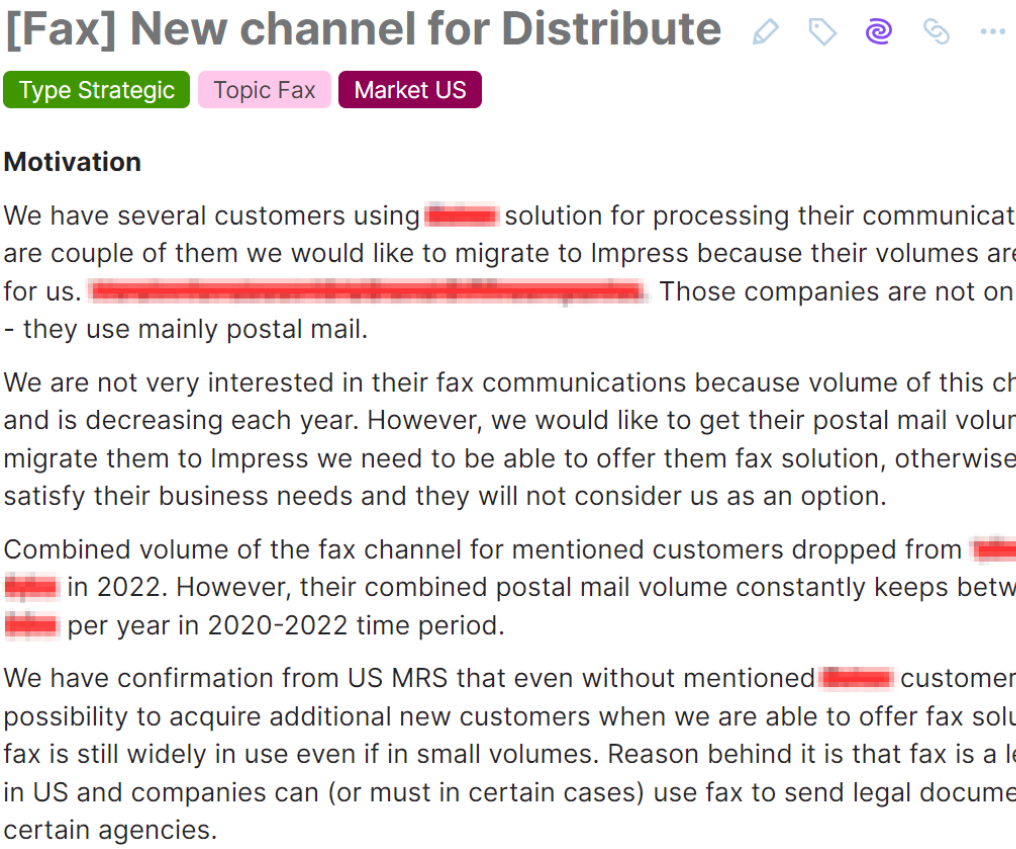
5.2 Úvod do projektu

V kontextu strategického rozhodování společnosti Quadient bylo zjištěno, že pro migraci některých potenciálních zákazníků využívajících konkurenční produkty na řešení Impress, jež se zabývá automatizací byznysových procesů, zejména pak fyzických i digitálních dokumentů a následné komunikace mezi firmami a koncovými uživateli, je klíčové rozšíření komunikačních kanálů. Impress, jakožto jeden z několika projektů společnosti Quadient, nabízí možnost využívání různých platforem pro distribuci obsahu, včetně klasické poštovní služby, emailu, SMS zpráv nebo zasílání dokumentů do klientských portálů, které lze přirovnat k osobním účtům koncových zákazníků.

Z hlediska finančního přínosu by migrace vybraných amerických a francouzských společností na řešení Impress znamenala významný nárůst příjmů a zisků pro Quadient. Tyto společnosti mají lákavý objem transakcí v oblasti poštovních služeb a Quadient generuje výnosy za každý zpracovaný kus komunikace nebo stránku

dokumentu. Přestože faxová komunikace není považována za moderní nebo vzrůstající technologii, zejména pak v oblasti České republiky, kde je spíše na ústupu, globální pohled odhaluje, že některé země a firmy stále považují fax za bezpečný způsob doručování dokumentů.

Podle analýzy popsané na obrázku 3 statistiky ukázaly, že objem faxové komunikace u potenciálních několika zákazníků klesl od roku 2020 do roku 2022, avšak objem poštovních zásilek zůstává stabilní. Tato situace naznačuje, že migrace na řešení Impress by byla pro Quadient výhodná i přes snižující se využívání faxů. Z agilního hlediska je zapojení faxového kanálu do Impress klíčové, aby zákazníci, kteří stále tento komunikační prostředek využívají, mohli přejít na nové řešení bez ztráty této možnosti. I když je objem faxové komunikace minimální, schopnost Impress podporovat tento kanál je zásadní pro uspokojení širšího spektra zákaznických potřeb.



Obrázek 3 Motivace implementace faxového kanálu. Zdroj: vlastní zpracování (YouTrack – software pro řízení projektů)

Quadient, díky své schopnosti rychle reagovat na dynamické potřeby zákazníků, demonstruje klíčové výhody agilního přístupu ve svém podnikání. Tato pružnost nejenže zvyšuje schopnost společnosti adaptovat se na měnící se tržní požadavky, ale také významně přispívá k posílení renomé značky. Když společnost prokáže, že dokáže

efektivně a rychle reagovat na specifické požadavky, ať už jde o integraci tradičních komunikačních kanálů jako faxu nebo o adaptaci na nové technologie, stává se pro zákazníky atraktivnější.

Agilní přístup umožňuje Quadientu nejen reagovat na současné potřeby trhu, ale také anticipovat budoucí trendy a přizpůsobit své produkty a služby tak, aby předběhly očekávání zákazníků. Taková strategie vede k vytváření hlubší důvěry a loajality mezi zákazníky, což je v dnešním konkurenčním prostředí neocenitelným aktivem.

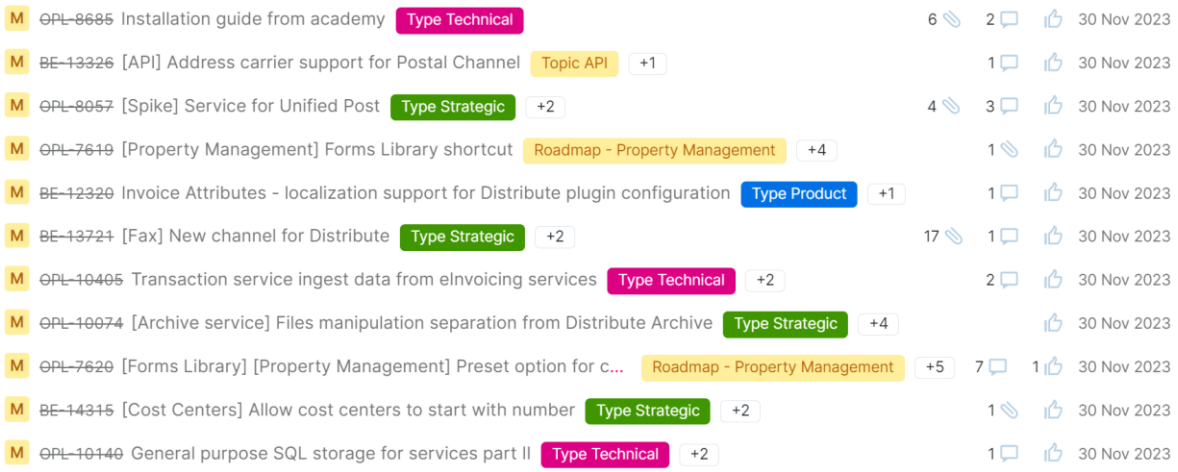
Dále, schopnost rychle inovovat a přizpůsobovat se, jak je vidět v agilním přístupu Quadientu, posiluje pozici firmy na trhu jako lídra v inovacích. To přináší nejen okamžité obchodní výhody, ale také dlouhodobě buduje pověst společnosti jako průkopníka ve svém oboru. Ve výsledku tento agilní přístup nejenže zajišťuje Quadientu konkurenční výhodu, ale také přispívá k udržitelnému růstu a rozvoji společnosti v globálním měřítku.

5.3 Začlenění do backlogu

V procesu definování vize pro další rozvoj produktů či cloudových aplikací v Quadientu není automaticky zaručeno, že každá nově navržená funkcionalita bude okamžitě zařazena do product backlogu. Nezbytným předpokladem pro zařazení je provedení důkladných interních analýz zaměřených na posouzení realizovatelnosti a vhodnosti dané funkcionality. Tyto analýzy zahrnují využití expertních systémů, interakci s interními vědomostními databázemi a diskuse s akcionáři, stakeholdery, představenstvem a managementem. Součástí tohoto procesu je také porovnání navrhovaných funkcionalit s již naplánovanými epics a stávajícím stavem backlogu.

Je důležité si uvědomit, že vývoj nových funkcionalit není jediným úkolem vývojových týmů v Quadientu. Vedle toho je kladen důraz na technické aspekty, jako jsou přechody na modernější technologie, aktualizace databázových řešení či inovace v oblasti ukládání dat. Tyto technické požadavky, ač nejsou přímo viditelné pro koncového zákazníka, hrají klíčovou roli v udržení kvality a efektivity softwaru. Další důležitou oblastí je optimalizace a zefektivnění stávajícího kódu vzhledem k rychlému vývoji programovacích jazyků. Neopomenutelnou součástí je také údržba aplikací, zahrnující včasné reagování na opravování chyb, nahlášených jak zákazníky, tak interně vývojovými týmy.

Product backlog v Quadientu je chápán jako „živý dokument“, neustále se měnící a aktualizovaný podle aktuálních potřeb a kapacit. Vizuální reprezentaci seznamu položek product backlogu odhaluje obrázek 4. V tomto kontextu je nezbytné pečlivě zvážit, zda existuje dostatečná kapacita pro zařazení dalších epics do backlogu a případně, co je potřeba upozadit nebo odstranit, aby bylo možné nové priority efektivně řešit.



Obrázek 4 Úryvek položek product backlogu. Zdroj: vlastní zpracování (YouTrack – software pro řízení projektů)

Pokud je rozhodnuto, že všechny podmínky a souvislosti umožňují přidání nového epic, v tomto případě kanálu pro fax, dochází k aktualizaci položek v product backlogu. Technický manažer produktu v této fázi přebírá epic za svůj a definuje konkrétní požadavky pro nižší úrovně, zejména pro product ownera. Avšak i přes aktualizaci backlogu je třeba si uvědomit, že kvůli dynamické povaze agilního přístupu a neustálým změnám v prioritách a kapacitách není zaručeno, že nová funkcionalita bude ihned začleněna do vývoje. Tento proces odhaluje, jak Quadient využívá agilní přístup k udržení flexibilního a adaptabilního plánování v rámci svých produktových strategií.

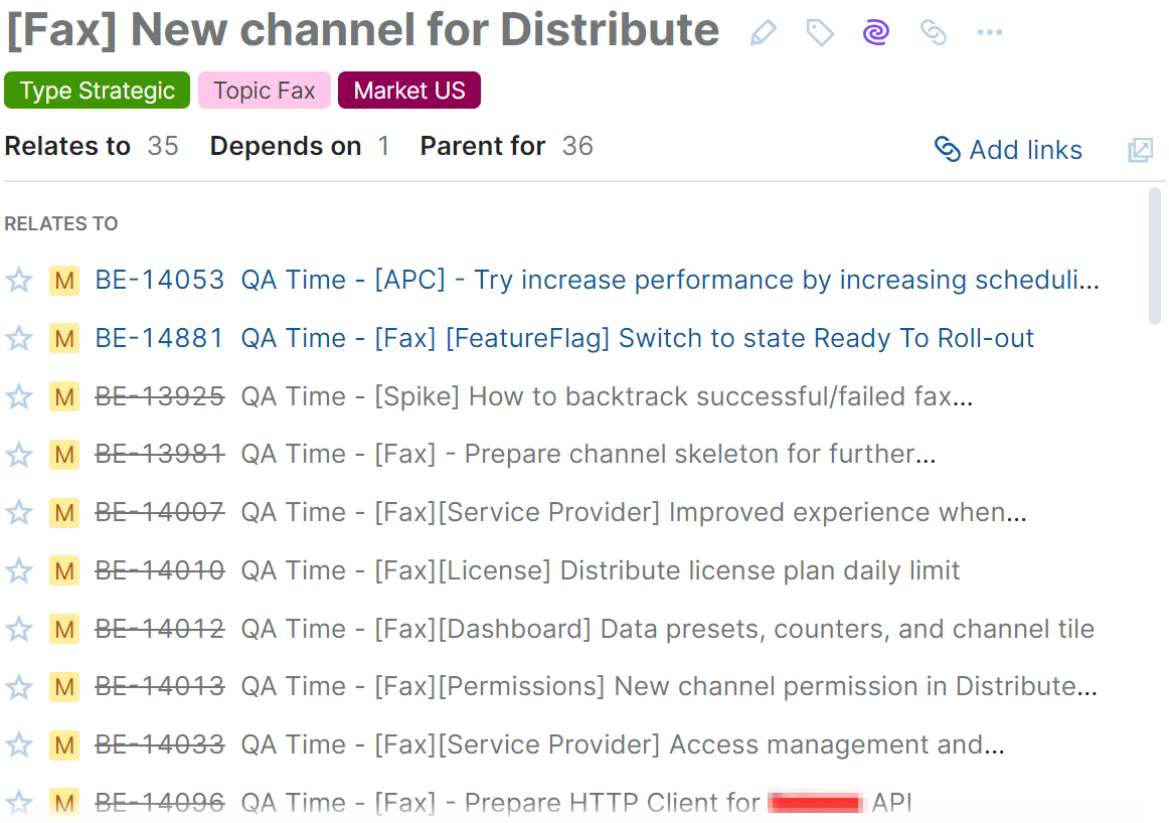
5.4 Plánování a zdroje

V procesu plánování a přidělování zdrojů v Quadientu, když před product ownerem stojí konkrétní požadavek, začíná fáze definování jednotlivých user stories. Tyto user stories atomizují konkrétní požadavek do nejelementárnějších úkonů. Je důležité poznamenat, že jeden epic může obsahovat jeden nebo více konkrétních požadavků a následně pod jeden konkrétní požadavek může spadat jeden nebo více user stories.

V některých situacích může jeden epic představovat pouze jediný konkrétní požadavek, který je rozdělen do jedné user story, avšak toto není obecné pravidlo.

Product owner analyzuje tyto konkrétní požadavky a rozhoduje, který z nich se začne vyvíjet jako první s cílem přinést maximální hodnotu zákazníkovi. Tato rozhodnutí a preference jsou koordinovány s technickým manažerem produktu. Paralelně k této aktivitě dochází k definování zdrojů, což zahrnuje výběr vhodného vývojového týmu pro danou funkčnost. Tento výběr je založen na analýze vytíženosti, odborných znalostí a úrovně vědomostí každého týmu v souvislosti s konkrétní částí aplikace.

V příkladu vývoje nového kanálu pro fax ukazuje obrázek 5, že jeden konkrétní požadavek zahrnoval celkem 72 dílčích úkonů, z nichž 36 je tvořeno user stories a 35 jsou související aktivity, jako jsou testovací nebo vývojové podpůrné úkoly. Posledním úkolem je nezbytná analýza hrozeb, která je klíčovou součástí vývoje softwaru a na které je konkrétní požadavek zcela závislý.



Obrázek 5 Úkony faxového kanálu. Zdroj: vlastní zpracování (YouTrack – software pro řízení projektů)

Je třeba zdůraznit, že product backlog je dynamicky se měnící seznam úkolů, což znamená, že počty a rozsah úkolů se mohou během vývoje měnit. Výčet hodnot uvedený v předchozím odstavci je konečný. Product owner není schopen, a bylo by to

i v rozporu s agilním přístupem, aby vymyslel všechny potřebné user stories pro celý konkrétní požadavek. Product owner postupně generuje a zadává práci, přičemž prioritizuje oblasti s nejvyšší přidanou hodnotou a postupně seznam doplňuje, upravuje nebo i odstraňuje některé položky. Vzhledem k orientaci product ownera v Quadientu na spolupráci s vývojovými týmy dochází k pravidelné iteraci a koordinaci ohledně plánů dodávek softwarových částí, a to jak z hlediska časového rámce, tak obsahu. Tento přístup, který zdůrazňuje interakci mezi product ownerem a vývojovými týmy, je v souladu s agilním myšlením a zároveň reflektuje specifické operační potřeby Quadientu.

5.5 Iterační proces

V kapitole o iterativním procesu v Quadientu je popsán mechanismus spolupráce mezi vývojovým týmem a product ownerem a také způsoby, jakými se informace sdílí a řídí v rámci týmu.

5.5.1 Spolupráce mezi vývojovým týmem a product ownerem

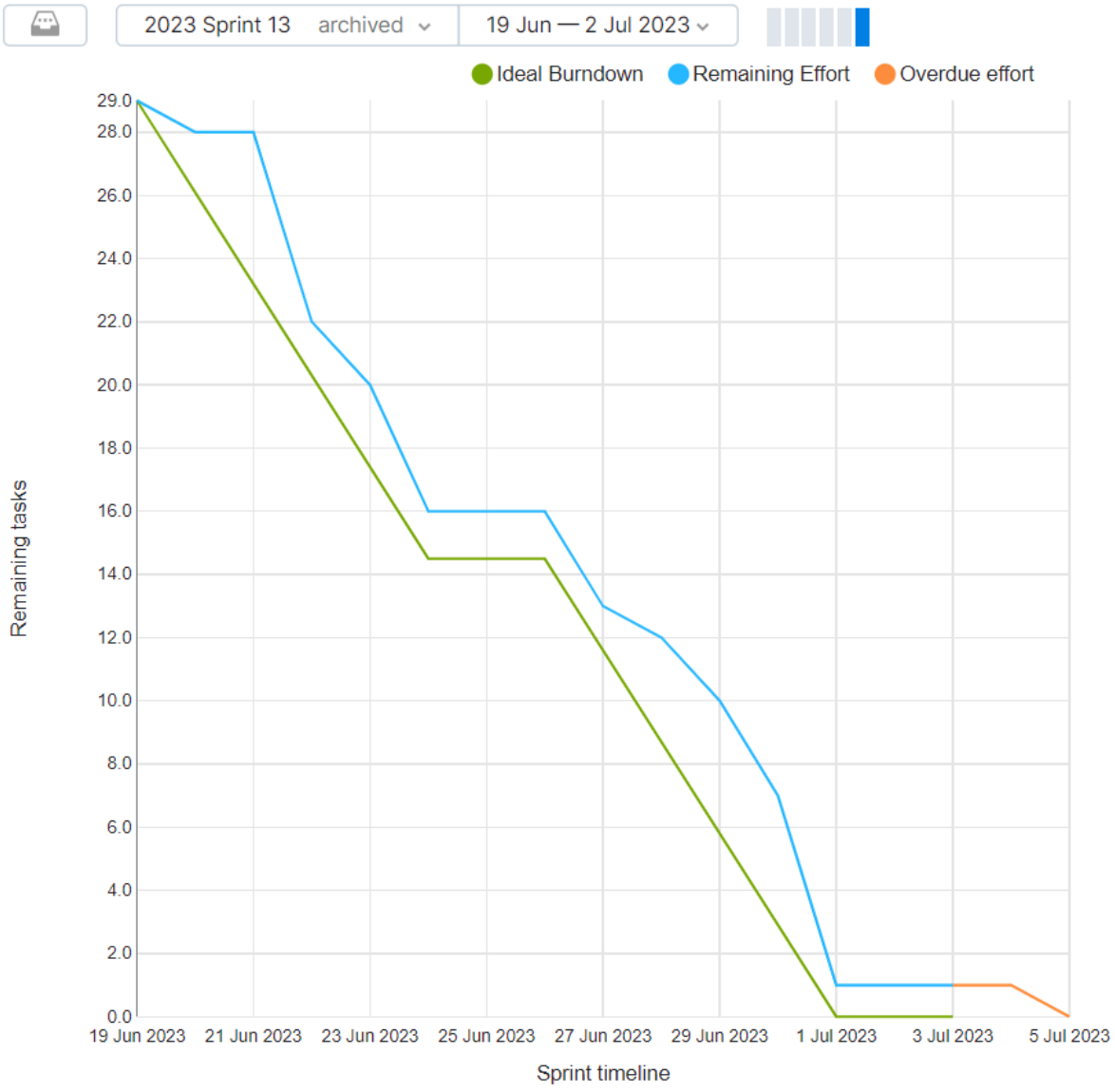
Na počátku fáze, kdy jsou formulovány user stories, se v Quadientu odehrává klíčové spojení mezi vývojovým týmem a product ownerem, a to prostřednictvím groomingů. V rámci těchto setkání product owner představuje definované úkoly a snaží se poskytnout širší kontext, známý jako „big picture“, což napomáhá vytvoření spojení mezi obchodním očekáváním a technickou realizací projektu.

V Quadientu se groomingy nekonají podle pevně stanoveného harmonogramu, ale jsou organizovány ad hoc, podle aktuálních potřeb vývojového týmu a dostupnosti product ownera. Důraz je přitom kladen na důkladný přenos myšlenek a konceptů, přičemž se od členů vývojového týmu očekává aktivní účast a dotazování. Cílem je zajistit, aby všichni měli jasnou představu o cílech a očekáváních projektu.

Zajímavým výsledkem těchto groomingů bývá také iniciativa ze strany vývojového týmu k doplnění nebo upřesnění user stories. Tím se nejen podporuje spoluvlastnictví procesu, ale také se zvyšuje zapojení a motivace týmu, což jsou klíčové aspekty úspěšného agilního prostředí.

5.5.2 Procesy uvnitř vývojového týmu

V Quadientu je základem udržení aktuálních informací a plynulého toku komunikace pravidelné konání denních standupů. Tyto krátké, ale strukturované schůzky poskytují členům týmu platformu pro sdílení informací o svém dosavadním pokroku, narazí-li na překážky a plány na další pracovní den. S nástupem pandemie COVID-19 a zvýšeným zaměřením na práci z domova se Quadient přizpůsobil těmto změnám a přesunul tyto standupy do online prostředí. Tento přístup umožňuje efektivnější správu projektů a lepší dostupnost informací prostřednictvím digitálních nástrojů, jako je YouTrack.



Obrázek 6 Trend plnění závazků sprintu. Zdroj: vlastní zpracování (YouTrack – software pro řízení projektů)

Tento přechod na digitální tabule úkolů, který zahrnuje nástroje jako YouTrack, přinesl týmům mnoho výhod. Díky digitálnímu formátu je možné získat lepší přehled

o úkolech nebo jejich trendu, jak je dokumentováno na obrázku 6. Zároveň je zajištěna snadná dostupnost informací i mimo kancelářské prostředí, což je zvláště důležité pro ty, kteří pracují na dálku. Navíc digitální platformy umožňují rychlejší a efektivnější interakci mezi členy týmu, což napomáhá udržet plynulost pracovních procesů a koordinace napříč celým projektem.

5.6 Hodnocení sprintu a částečné dodávky

V Quadientu je pravidelný sprintový cyklus nastaven na období dvou týdnů. V rámci tohoto cyklu se každé druhé pondělí konají prezentace, ve kterých vývojové týmy demonstrují výsledky své práce. Tato prezentace začíná krátkou motivací, následuje demonstrace funkčnosti přímo v aplikaci. Zde je důležité, že i malé přidání, jako například ikona faxu s průvodcem, může mít významný dopad. Tento způsob prezentace umožňuje ukázat i menší části práce v kontextu celkového vývoje.

Quadient má za cíl vydávat nové verze produktu každé čtyři týdny, což znamená, že uživatelé mají příležitost pravidelně zažívat nové funkce a vylepšení. V případě vývoje fax kanálu, který trval celkem 132 dní, tedy necelých deset sprintů, bylo důležité, aby byli zákazníci, v tomto případě techničtí manažeři produktu a produktoví manažeři, průběžně informováni a měli možnost vyjádřit svůj názor na postup práce. Poslední prezentace obvykle obsahuje scénář „end-to-end“, který ukazuje funkčnost ve své celistvosti.

S ohledem na agilní přístup je nezbytné zmínit sprintovou retrospektivu. Stejně jako jsou pravidelně prezentovány dílčí části práce na aplikaci, je důležité se věnovat i interní reflexi v týmu. V rámci retrospektivy se provádí diskuse o úspěších, které jsou dokumentovány na obrázku 7, debatovaných tématech, jež jsou podrobněji ilustrovány na obrázku 8 a příležitostech pro zlepšení, na nichž se zaměřuje obrázek 9. Tato diskuse zahrnuje širokou škálu témat, od práce týmu přes funkčnost aplikace a organizaci práce až po osobní rozvoj. Tyto retrospektivy jsou klíčové pro kontinuální zlepšování a týmovou spolupráci. Vývoj fax kanálu nebyl výjimkou a přinesl mnoho příležitostí pro růst a vzájemné učení se v týmu. Momentky z těchto setkání, plné jak úspěšných, tak náročnějších okamžiků, jsou často zdrojem motivace a týmové soudržnosti, což je neocenitelné pro úspěch jakéhokoli agilního projektu.

AKTUÁLNÍ POZITIVA	
kdo	**co**
D	Rozjíždí se konzultační skupina na řešení testovací strategie. Jsem součástí, takže testování nabyde nového směru :)
D	Martin je zpátky, vítej a ve zbytku času nám o tom můžeš povyprávět
P	~~Martin~~ Libor je zpátky, vítej a ve zbytku času nám o tom můžeš povyprávět
M	dík za veškeré zástupy během prázdnin

Obrázek 7 Týmová retrospektiva – aktuální pozitiva. Zdroj: vlastní zpracování

AKTUÁLNÍ TÉMATA	
kdo	**co**
D	Nefunkční API klíč Um96bHXEjWthIHMgRGF2aWRlbS4gS2R5PyBQZW7DrXplPw==
D	Reportování času v YT
P	QA Time o opuštěných storek - řeší se nějak?
P	Zbývající storky na FAX comming soon
M	chceme dalsi setkani Exit+Martina?
H	Requirementy bez groomingů
Lib	Má další cesta

Obrázek 8 Týmová retrospektiva – aktuální témata. Zdroj: vlastní zpracování

AKTUÁLNÍ PROBLÉMY	
	problemy/stiznosti
kdo	**co**
Lu	Nedostatečně zadaný REQ na [redacted]
P	Čí role je vyjednávání s externímy [redacted]
D	[redacted] support a vše okolo

Obrázek 9 Týmová retrospektiva – aktuální problémy. Zdroj: vlastní zpracování

5.7 Změny v požadavcích

V Quadientu, stejně jako v jakémkoliv jiném agilně řízeném vývoji, je product backlog vnímán jako živý dokument, který reaguje na aktuální potřeby zákazníků. V důsledku toho může dojít k situacím, kdy požadovaná funkcionalita během vývoje ztratí na významu nebo se dokonce stane nepotřebnou. V agilním světě je toto scénář, s nímž se musí počítat a který může být pro nováčky v oboru překvapivý. Zkušenost ukazuje, že teorie a realita se mohou výrazně lišit a až praktické prožití těchto situací poskytuje plné porozumění agilního přístupu.

Při vývoji faxového kanálu v Quadientu došlo k několika situacím, kdy se vývoj ubíral mírně odlišným směrem, než bylo původně plánováno. V některých případech to vedlo k nutnosti „utnout“ určité vývojové větve a přizpůsobit směr práce aktuálním potřebám. Tyto změny byly často iniciovány na základě zpětné vazby z hodnocení sprintů, což zdůrazňuje význam pružného přístupu a schopnosti rychle reagovat na změny.

Komplikovanější situace nastala, když se ukázalo, že faxový kanál nebude možné dodat v původně stanoveném termínu. Příčinou nebyly plánovací chyby, ale neočekávané technické problémy. Před týmem stála volba mezi zahozením celého projektu nebo jeho dodáním s omezeným rozsahem funkcí. Po konzultaci se zákazníkem bylo rozhodnuto dodat produkt i s těmito omezeními. Výsledkem bylo, že zákazník nevyjádřil zájem o zbývajících 10 % funkcí a byl spokojen s dodaným řešením.

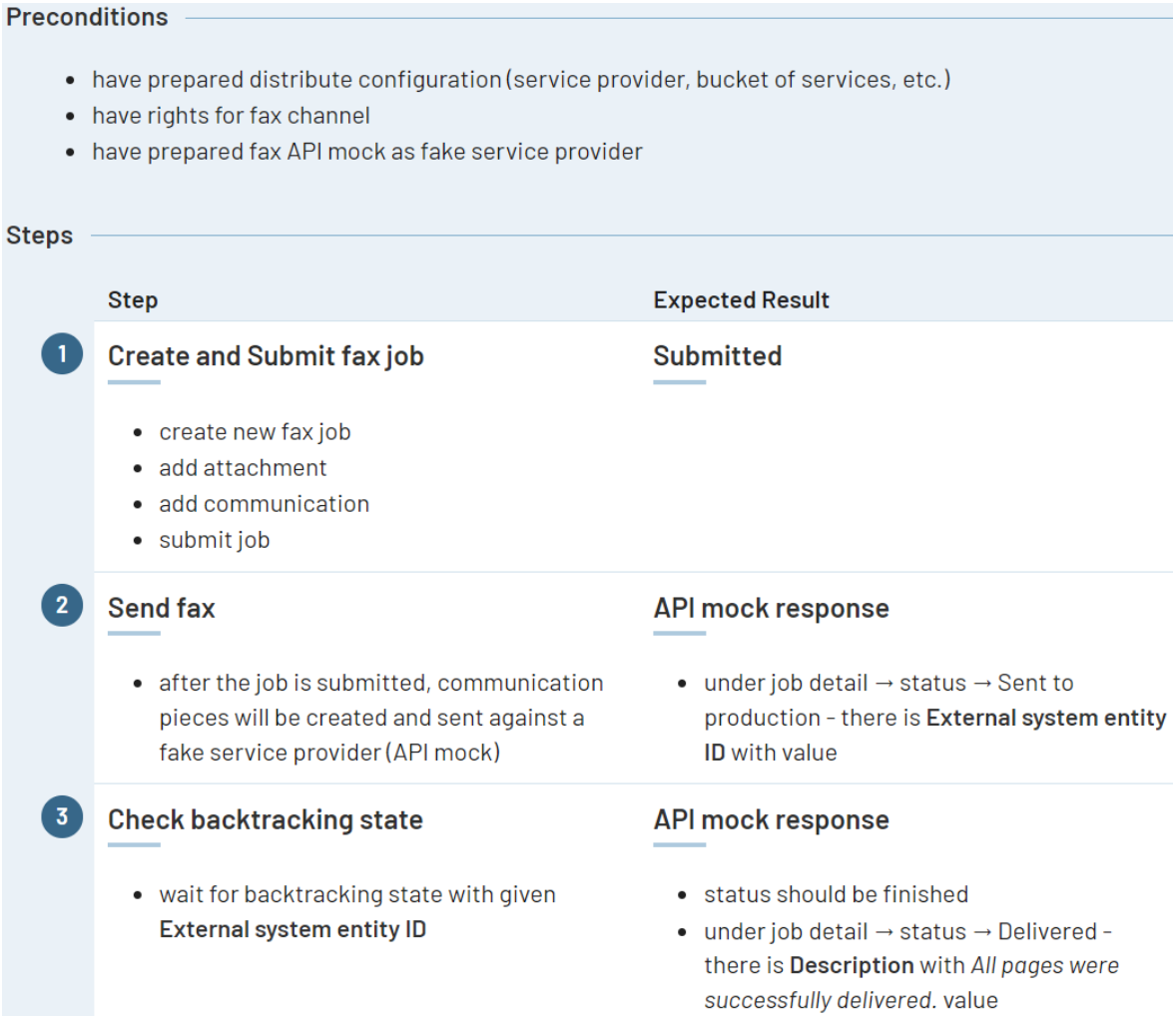
Tento případ poukazuje na důležitost transparentnosti a komunikace mezi vývojovým týmem a zákazníkem. Vzájemná otevřenost a upřímnost v komunikaci pomohla zachránit 132 dní vývoje a v konečném důsledku dodat produkt, který odpovídal skutečným potřebám zákazníka. Příklad jasně ukazuje, jak agilní přístup umožňuje flexibilně reagovat na měnící se požadavky a zajistit, že konečné výsledky jsou v souladu s očekáváními a potřebami zákazníků.

5.8 Testovací strategie

V Quadientu je testování softwaru považováno za klíčovou disciplínu, která má stejnou důležitost jako vývoj samotného softwaru. To je zásadní pro udržení vysoké kvality produktů a budování dobrého jména značky. Vývoj faxového kanálu byl také podroben intenzivnímu testovacímu procesu, který ilustruje přístup společnosti k testování.

Je důležité podotknout, že dříve v Quadientu existovalo izolované oddělení pro testování, což vedlo k neefektivní komunikaci mezi vývojáři a testery. Před zhruba třemi lety došlo k významné změně, kdy byli testeři integrováni přímo do vývojových týmů, což umožnilo zlepšit komunikaci, zkrátit reakční doby a zefektivnit vývoj i opravy chyb v softwaru. Díky této změně se vývojáři mohou občas zapojit do testování a naopak zkušení testeři mohou pomáhat s vývojem, což předtím nebylo možné.

Testování v Quadientu lze rozdělit na dvě hlavní kategorie: manuální a automatizované. Manuální testování zahrnuje ruční procházení aplikace, kde tester simuluje chování reálného uživatele a hledá chyby nebo neobvyklé chování aplikace. Automatizované testování na druhou stranu spočívá ve vytváření testů založených na logických scénářích, jako je například celý proces odeslání faxu a sledování jeho stavu, což je detailně vystiženo na obrázku 10.



Obrázek 10 Testovací scénář. Zdroj: vlastní zpracování (TestRail – platforma pro správu testů)

Při vývoji faxového kanálu bylo důležité, aby se již během prvních groomingů začalo s konceptualizací automatizovaných testů. Jakmile byla vytvořena první „big picture“, začalo se s přemýšlením nad testovacím scénářem pro novou funkčnost ze strany testerů. To vedlo k vytvoření směrodatného testovacího plánu, který sloužil jako vodítko pro další fáze vývoje. S dodávkami nových funkcionalit byly vyvíjeny automatizované testy, což umožnilo efektivní dokončení testování souběžně s vývojem softwaru. Tento proces, označovaný jako „done-done“, zajišťuje, že jak vývojáři, tak testeři dokončují svou práci téměř ve stejnou dobu, což vede k efektivnějšímu využití času a zdrojů. Ukázka kódu funkčního testu je zachycena na obrázku 11.

```
[Test]
[Parallelizable]
simple enough (5%)  ◇  ? usages  ? overrides  Daniel Hanák
public async Task GivenFaxConfiguration_WhenJobIsTransmitted_ShouldReceiveFinishedBacktracking()
{
    using var faxMock = new FaxMock(Company, GetFaxMockFilePath());
    var contact = await PrepareConfigurationAsync(faxMock);

    var distributeMessage = await CreateJobInTransmittedStateAsync(contact.Id);

    await PrepareMockForBacktracking(distributeMessage, faxMock, contact.Data.FaxNumber);
    await distributeMessage.WaitForStateAsync(MessageStateDto.Finished);
}
```

Obrázek 11 Kód funkčního testu faxového kanálu (C#). Zdroj: vlastní zpracování (Rider – vývojové prostředí)

5.9 Dodání a uvedení do provozu

V současné době Quadient primárně vyvíjí aplikace v cloudové podobě, což přináší řadu výhod, jako jsou časté dodávky a snížení nutnosti vlastnit drahý hardware přímo zákazníky. Proces vydání nové verze softwaru v Quadientu prochází několika fázemi, které zajistí hladké a efektivní uvedení nových funkcionalit do provozu.

Prvním krokem po vydání nové funkcionality, například faxového kanálu, je nasazení na vývojová testovací prostředí. Do těchto prostředí mají přístup především všechny vývojové týmy, což umožňuje nahodilé testování nových funkcí a identifikaci případných chyb. Toto vývojové testovací prostředí slouží jako bezpečná platforma pro prvotní ověření funkčnosti a stability nových prvků aplikace předtím, než se dostanou do dalších fází.

Následuje fáze, kdy je nová verze softwaru zpřístupněna internímu personálu, převážně z oddělení marketingu a prodeje. Tato fáze je důležitá, protože umožňuje

těmto týmům seznámit se s novými prvky aplikace a připravit se na jejich prezentaci a propagaci mezi zákazníky. Tato interní fáze testování a přípravy slouží k zajištění, že všechny relevantní oddělení jsou dobře připraveny na uvedení nové funkcionality na trh.

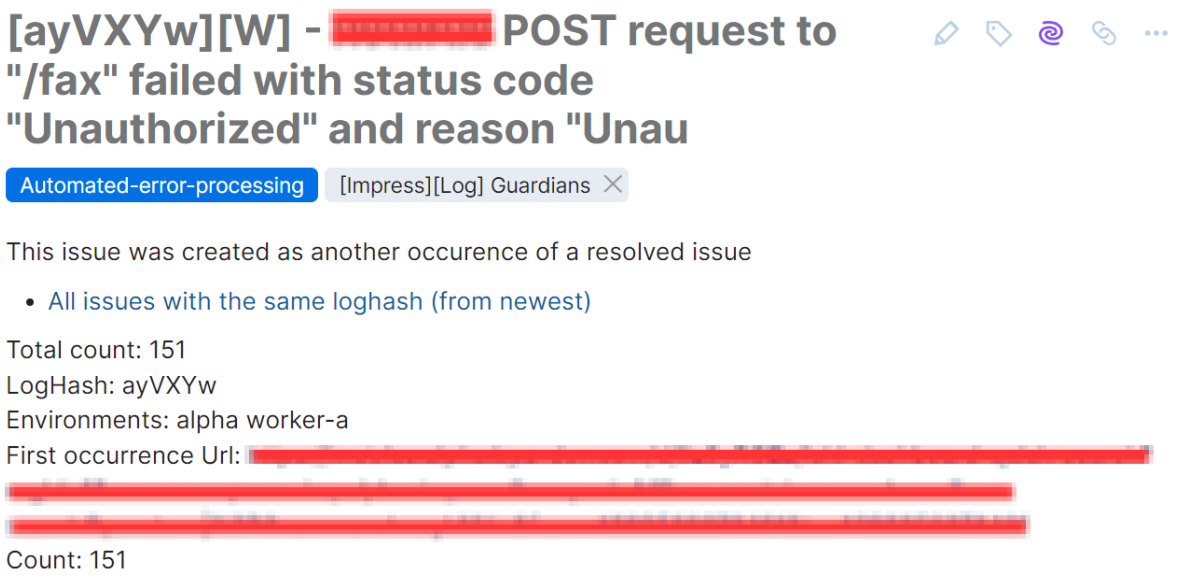
Posledním krokem v procesu uvedení do provozu je nasazení do produkčního prostředí, kde nová funkcionalita je konečně dostupná skutečným uživatelům, tedy zákazníkům. Toto je klíčová fáze, kdy se nové funkce stávají plně operativními a začínají být využívány v běžném provozu.

Celkově, od nasazení na vývojová prostředí až po spuštění v produkčním prostředí, je časový interval potřebný pro kompletní proces nasazení přibližně čtyři až šest týdnů. Toto období je nezbytné pro zajištění hladkého přechodu a zahrnuje čas potřebný pro testování, interní přípravu a zajištění, že všechny nové prvky jsou plně funkční a připravené pro finální uživatele.

Dokončení nasazení nové funkcionality na produkční prostředí ve společnosti Quadient není konečnou fází v životním cyklu produktu. Po uvedení do skutečného provozu mohou nastat situace, kdy uživatelé narazí na problémy nebo nejasnosti, které vyžadují podporu. V Quadientu je oddělení podpory rozděleno do tří úrovní, aby efektivně reagovalo na různé typy problémů.

První úroveň podpory zahrnuje produktovou podporu, kde členové týmu nebo interní zaměstnanci pomáhají zákazníkům s jednoduššími problémy. Pokud tato úroveň není schopna problém vyřešit, předává se situace technické podpoře, která se skládá ze zkušenějších uživatelů s technickým zázemím. V případě, že i tato úroveň nenalezne řešení, je problém předán vývojovému týmu, který pracoval na dané funkcionalitě nebo je s ním alespoň komunikováno o možných opravách.

Kromě reaktivní podpory se vývojové týmy v Quadientu také zabývají proaktivní, v některých situacích by se dalo mluvit i o prediktivní, údržbou produktu. To zahrnuje automatické reportování neobvyklých stavů aplikace, jak je dokumentováno na obrázku 12. Tyto úkony jsou následně analyzovány a rozebírány na denních standupech. Vývojové týmy se pak snaží v co nejkratším čase provést potřebné nápravy. Tento proces nejen zajišťuje rychlou reakci na problémy, ale také přispívá k udržení vysoké kvality a spolehlivosti produktů Quadientu.



Obrázek 12 Automaticky vytvořený úkon z abnormálního stavu aplikace. Zdroj: vlastní zpracování (YouTrack – software pro řízení projektů)

Tímto způsobem Quadient udržuje kontinuální zlepšování a podporu svých produktů, což je v souladu s agilním přístupem, který zdůrazňuje neustálou adaptabilitu a odpověď na měnící se potřeby uživatelů.

5.10 Budoucí kroky

V rámci údržby konkrétního kusu kódu, jako byl vývoj faxového kanálu v Quadientu, pokračuje práce vývojového týmu a dalších podpůrných týmů až do okamžiku, kdy daná funkcionalita dosáhne konce svého životního cyklu. Během tohoto období se tým může setkat s řadou výzev, které vyžadují neustálou pozornost a zásahy.

Tyto výzvy zahrnují řešení problémů a chyb, které mohou být nahlášeny jak zákazníky, tak interně zaměstnanci Quadientu. Dalším aspektem údržby je neustálé zlepšování a adaptace na měnící se bezpečnostní hrozby, které vyžadují aktualizace a zvýšení úrovně zabezpečení softwaru. Vývojový tým se také může zabývat restrukturalizací kódu, což zahrnuje refaktoring (proces zvyšování čitelnosti kódu bez změny jeho chování) nebo přepis kódu pro různé typy softwarové architektury. Tento neustálý vývoj a úpravy jsou nezbytné pro udržení kvality, efektivity a bezpečnosti softwaru.

Jakmile však společnost jako celek dospěje k závěru, že určitá část produktu již není rentabilní nebo se již nepoužívá, což lze posoudit na základě sledování různých metrik, nastává rozhodovací proces o možném odstranění této funkce. Po důkladném zhodnocení a schvalovacím procesu může dojít k odstranění daného kusu kódu, čímž daná funkcionalita zcela zaniká. Tento proces odstranění je pečlivě řízen

a koordinován, aby se zajistilo, že zákazníci nebudou negativně ovlivněni a aby byla zachována celková integrita a funkčnost produktu.

6 Shrnutí

V této bakalářské práci bylo podrobně zkoumáno, jak společnost Quadient aplikuje agilní metodiky v procesu softwarového vývoje s důrazem na to, jak jsou tyto praktiky a principy upravovány a integrovány do jejího korporátního prostředí. Bylo zjištěno, že Quadient upravuje a přizpůsobuje tradiční agilní role a procesy, aby vyhověly specifickým potřebám a zároveň zachovávaly agilní flexibilitu a inovativnost.

V rámci agilního přístupu v Quadientu byla zvláště zajímavá adaptace a interpretace rolí technický manažer produktu, neboli „TPM“, a product owner, zkráceně „PO“. TPM se soustředí více na technické aspekty a přeměnu zákaznických požadavků do technických specifikací, zatímco PO má větší zaměření na interakci s vývojovými týmy, což je odlišné od tradičního agilního modelu. Zajímavým aspektem je také absence role scrum mastera, což ukazuje na řízení agilních týmů více tradiční hierarchickou managementovou strukturou.

Práce je strukturována tak, aby poskytla podrobný pohled na životní cyklus vývoje konkrétní funkcionality v Quadientu. Tento přístup umožňuje hlouběji proniknout do specifik procesů a postupů, které jsou využívány při vývoji softwaru v rámci agilního prostředí společnosti. Od počátečního začlenění funkcionality do backlogu, přes plánování a vývoj, až po testování, dodání a uvedení do provozu, je každá fáze v této závěrečné bakalářské práci zkoumána. Tento pohled poskytuje ucelený obraz o tom, jak Quadient řídí a realizuje softwarové projekty, od nápadu až po jeho realizaci a následnou údržbu. Tento způsob prezentace umožňuje čtenáři lépe pochopit komplexitu a dynamiku agilního vývoje softwaru v korporátním prostředí a zdůrazňuje význam každého kroku v procesu vývoje softwaru.

Během výzkumu byla zaznamenána integrace testerů přímo do vývojových týmů, což vede k lepší komunikaci a spolupráci mezi členy týmu a zvyšuje efektivitu vývojového procesu. Quadient rovněž uplatňuje běžné agilní postupy, jako jsou groomingy a standupy, avšak s určitými modifikacemi pro lepší vyhovění interním procesům a požadavkům.

Práce dále poukazuje na důležitost pružnosti a schopnosti rychle reagovat na měnící se tržní podmínky a potřeby zákazníků, což je klíčové pro udržení konkurenceschopnosti v dynamickém softwarovém průmyslu. Přístup Quadientu,

který umožňuje rychlou adaptaci a inovaci, umožňuje reagovat na aktuální tržní výzvy a předvídat budoucí trendy.

Závěrem, tato práce poskytuje komplexní přehled o tom, jak Quadient přizpůsobuje a implementuje agilní metodiky ve své praxi a jak adaptace agilních principů a metod může přispět k efektivitě, inovativnosti a úspěšnosti softwarových projektů v korporátním prostředí. Tímto přístupem společnost zdůrazňuje význam flexibility a přizpůsobení v agilním řízení projektů, což je zásadní pro jejich úspěšné vedení v rámci korporátního kontextu.

7 Závěry a doporučení

V rámci této bakalářské práce bylo provedeno podrobné zkoumání aplikace agilních metodik ve společnosti Quadient. Bylo zjištěno, že Quadient efektivně integruje agilní metodiky do svého softwarového vývoje, ale s významnými úpravami a přizpůsobeními, aby vyhověly specifickým potřebám a struktuře podniku. Tyto změny zahrnují modifikaci rolí jako technický manažer produktu a produkt owner a provoz bez tradiční role scrum mastera. Tato práce rovněž ukazuje, že integrace testerů přímo do vývojových týmů v Quadientu výrazně zlepšuje komunikaci a efektivitu vývojového procesu, což přispívá k lepší spolupráci a rychlejšímu řešení problémů.

Na základě těchto zjištění je doporučeno, aby Quadient pokračoval v rozvoji a přizpůsobování agilních metodik, které by ještě více vyhovovalo specifickým podnikovým potřebám. Je důležité, aby se společnost zaměřila na další vylepšení role product ownera, ve smyslu lépe vyvážené komunikace se zákazníky a současně podpory účinné interakce s vývojovými týmy. Toto by mohlo zahrnovat školení a rozvoj dovedností pro product ownera, aby firma ještě více zlepšila své pochopení zákaznických potřeb.

Quadient by měl také zvážit zavedení role scrum mastera, aby podpořil lepší samoorganizaci týmů a efektivnější řízení agilních procesů. Tato role by mohla přinést další strukturu a podporu pro týmy, což by mohlo zlepšit celkovou efektivitu a průběh projektů.

Průběžné vzdělávání a školení zaměstnanců v oblasti agilního myšlení by mělo být rovněž prioritou, aby se zajistilo, že všechny týmy jsou neustále informovány o nejnovějších agilních metodikách a nejlepších praktikách. Toto nejen podporuje inovativní a efektivní pracovní prostředí, ale také pomáhá udržet zaměstnance motivované a zapojené.

V neposlední řadě je pro Quadient klíčové udržet otevřený postoj k inovacím a změnám, aby mohl efektivně reagovat na nové technologické trendy a požadavky trhu. V dynamicky se vyvíjejícím softwarovém odvětví je schopnost rychle se přizpůsobit a inovovat zásadní pro dlouhodobý úspěch. Tyto kroky umožní Quadientu udržet si svou pozici na špičce inovací v oblasti softwarového vývoje a zároveň zajistit, že jejich produkty a služby nadále vyhovují potřebám a očekáváním zákazníků.

8 Seznam použité literatury

ABBAS, Noura, Andrew M. GRAVELL a Gary B. WILLS, 2008. Historical Roots of Agile Methods: Where Did "Agile Thinking" Come From? In: Pekka ABRAHAMSSON, Richard BASKERVILLE, Kieran CONBOY, Brian FITZGERALD, Lorraine MORGAN a Xiaofeng WANG, ed. *Agile Processes in Software Engineering and Extreme Programming* [online]. Berlin, Heidelberg: Springer, s. 94–103. ISBN 978-3-540-68255-4. Dostupné z: doi:10.1007/978-3-540-68255-4_10

AHMAD, Muhammad Ovals, Denis DENNEHY, Kieran CONBOY a Markku OIVO, 2018. Kanban in software engineering: A systematic mapping study. *JOURNAL OF SYSTEMS AND SOFTWARE* [online]. **137**, 96–113. ISSN 0164-1212, 1873-1228. Dostupné z: doi:10.1016/j.jss.2017.11.045

ALZAMIL, Zakarya A., 2019. Integrated Quality Model for Flexible Quality Management System. *QUALITY-ACCESS TO SUCCESS*. **20**(173), 3–8. ISSN 1582-2559.

BATRA, Dinesh, Weidong XIA, Debra VANDERMEER a Kaushik DUTTA, 2010. Balancing Agile and Structured Development Approaches to Successfully Manage Large Distributed Software Projects: A Case Study from the Cruise Line Industry. *COMMUNICATIONS OF THE ASSOCIATION FOR INFORMATION SYSTEMS*. **27**, 379–394. ISSN 1529-3181.

BECK, Kent a Cynthia ANDRES, 2012. *Extreme programming explained: embrace change*. 2. ed., 11. print. Boston: Addison-Wesley. The XP Series. ISBN 978-0-321-27865-4.

BECK, Kent, Mike BEEDLE, Arie VAN BENNEKUM, Alistair COCKBURN, Ward CUNNINGHAM, Martin FOWLER, James GRENNING, Jim HIGHSMITH, Andrew HUNT, Ron JEFFRIES, Jon KERN, Brian MARICK, Robert C. MARTIN, Steve MELLOR, Ken SCHWABER, Jeff SUTHERLAND a Dave THOMAS, 2001. *Manifesto for Agile Software Development* [online] [vid. 2023-12-29]. Dostupné z: https://agilemanifesto.org/

BERGER, Hilary a Paul BEYNON-DAVIES, 2009. The utility of rapid application development in large-scale, complex projects. *INFORMATION SYSTEMS JOURNAL* [online]. **19**(6), 549–570. ISSN 1350-1917, 1365-2575. Dostupné z: doi:10.1111/j.1365-2575.2009.00329.x

BOLLATI, Veronica A., German GAONA, Liliana CUENCA PLETSCH, Silvio GONNET a Horacio LEONE, 2017. The state of agile development adoption in argentine software companies. In: H. MONTEVERDE a R. SANTOS, ed. *43rd Latin American Computer Conference (CLEI)*: *2017 XLIII LATIN AMERICAN COMPUTER CONFERENCE (CLEI)* [online]. New York: IEEE [vid. 2024-03-16]. ISBN 978-1-5386-3057-0. Dostupné z: https://www.webofscience.com/wos/alldb/full-record/WOS:000426922000025

BRECHNER, Eric, 2015. *Agile project management with Kanban*. Redmond, Washington: Microsoft Press. Best practices. ISBN 978-0-7356-9895-6.

BURAGGA, Khalid a Noor JHANJHI, 2013. *Software Development Techniques for Constructive Information Systems Design*. ISBN 978-1-4666-3679-8.

COOPER, Robert G. a Anita F. SOMMER, 2016. From Experience: The Agile-Stage-Gate Hybrid Model: A Promising New Approach and a New Research Opportunity. *JOURNAL OF PRODUCT INNOVATION MANAGEMENT* [online]. **33**(5), 513–526. ISSN 0737-6782, 1540-5885. Dostupné z: doi:10.1111/jpim.12314

COOPER, Robert G. a Anita Friis SOMMER, 2018. Agile-Stage-Gate for Manufacturers Changing the Way New Products Are Developed Integrating Agile project management methods into a Stage-Gate system offers both opportunities and challenges. *RESEARCH-TECHNOLOGY MANAGEMENT* [online]. **61**(2), 17–26. ISSN 0895-6308, 1930-0166. Dostupné z: doi:10.1080/08956308.2018.1421380

CRAFT.CO, 2023. Quadient Company Profile - Office Locations, Competitors, Revenue, Financials, Employees, Key People, Subsidiaries | Craft.co. *Craft.co* [online] [vid. 2024-01-21]. Dostupné z: https://craft.co/quadient

CRISPIN, Lisa a Janet GREGORY, 2009. *Agile testing: a practical guide for testers and agile teams*. Upper Saddle River, NJ: Addison-Wesley. The Addison-Wesley signature series. ISBN 978-0-321-53446-0.

DÍAZ, Jessica, Rubén ALMARAZ, Jennifer PÉREZ a Juan GARBAJOSA, 2018. DevOps in practice: an exploratory case study. In: *Proceedings of the 19th International Conference on Agile Software Development: Companion* [online]. New York, NY, USA: Association for Computing Machinery, s. 1–3 [vid. 2023-12-28]. XP '18. ISBN 978-1-4503-6422-5. Dostupné z: doi:10.1145/3234152.3234199

DOLEŽAL, Jan, 2022. *Agilní přístupy vývoje produktu a řízení projektu: komplexně, prakticky a dle světové praxe*. První vydání. Praha: Grada. ISBN 978-80-271-3705-3.

DRURY, Meghann, Kieran CONBOY a Ken POWER, 2012. Obstacles to decision making in Agile software development teams. *JOURNAL OF SYSTEMS AND SOFTWARE* [online]. **85**(6), 1239–1254. ISSN 0164-1212, 1873-1228. Dostupné z: doi:10.1016/j.jss.2012.01.058

DYBA, Tore a Torgeir DINGSOYR, 2008. Empirical studies of agile software development:: A systematic review. *INFORMATION AND SOFTWARE TECHNOLOGY* [online]. **50**(9–10), 833–859. ISSN 0950-5849, 1873-6025. Dostupné z: doi:10.1016/j.infsof.2008.01.006

FRITZSCHE, Martin, 2008. Agile methods and requirements engineering in change intensive projects. In: C. GONZALEZPEREZ a S. JABLONSKI, ed. *3rd International Conference on Evaluation of Novel Approaches to Software Engineering*: *ENASE 2008: PROCEEDINGS OF THE THIRD INTERNATIONAL CONFERENCE ON EVALUATION OF NOVEL APPROACHES TO SOFTWARE ENGINEERING* [online]. Setubal: Insticc-Inst Syst Technologies Information Control & Communication, s. 81–88 [vid. 2024-03-16]. ISBN 978-989-8111-28-9. Dostupné z: https://www.webofscience.com/wos/alldb/full-record/WOS:000258473500016

GLIGOR, David M., Carol L. ESMARK a Mary C. HOLCOMB, 2015. Performance outcomes of supply chain agility: When should you be agile? *JOURNAL OF OPERATIONS MANAGEMENT* [online]. **33–34**, 71–82. ISSN 0272-6963, 1873-1317. Dostupné z: doi:10.1016/j.jom.2014.10.008

GOMES SILVA, Francisco Jose, Konstantinos KIRYTOPOULOS, Luis Pinto FERREIRA, Jose Carlos SA, Gilberto SANTOS a Maria Carolina CANCELA NOGUEIRA, 2022. The three pillars of sustainability and agile project management: How do they influence each other. *CORPORATE SOCIAL RESPONSIBILITY AND ENVIRONMENTAL MANAGEMENT* [online]. **29**(5), 1495–1512. ISSN 1535-3958, 1535-3966. Dostupné z: doi:10.1002/csr.2287

HOHL, Philipp, Jil KLÜNDER, Arie VAN BENNEKUM, Ryan LOCKARD, James GIFFORD, Jürgen MÜNCH, Michael STUPPERICH a Kurt SCHNEIDER, 2018. Back to the future: origins and directions of the "Agile Manifesto" – views of the originators. *Journal of Software Engineering Research and Development* [online]. **6**(1), 15. ISSN 2195-1721. Dostupné z: doi:10.1186/s40411-018-0059-z

IBRAHIM, Ku Saimah Ku, Jamaiah YAHAYA, Zulkefli MANSOR a Aziz DERAMAN, 2019. The Emergence of Agile Maintenance: A Preliminary Study. In: *2019 International Conference on Electrical Engineering and Informatics (ICEEI)*: *2019 International Conference on Electrical Engineering and Informatics (ICEEI)* [online]. Bandung, Indonesia: IEEE, s. 146–151 [vid. 2023-12-27]. ISBN 978-1-72812-418-6. Dostupné z: doi:10.1109/ICEEI47359.2019.8988815

JIANG, Li a Armin EBERLEIN, 2009. An analysis of the history of classical software development and agile development. In: *2009 IEEE International Conference on Systems, Man and Cybernetics*: *2009 IEEE International Conference on Systems, Man and Cybernetics* [online]. s. 3733–3738 [vid. 2023-12-27]. ISSN 1062-922X. Dostupné z: doi:10.1109/ICSMC.2009.5346888

JUSTICE, Joe, 2021. *Scrum master: the agile training seminar for business performance . complete training for instructors, professionals, business leaders, and the professionally curious*. Book version 2.0A. s.l.: Agile Business Institute. ISBN 9798704954057.

KEITH, Clinton, 2020. *Agile game development: iterative, lean, and scaled practices for game development*. 2. vyd. Hoboken: Pearson Education, Inc. ISBN 978-0-13-652781-7.

KLUNDER, Jil, Felix TROMMER a Nils PRENNER, 2022. How agile coaches create an agile mindset in development teams: Insights from an interview study. *JOURNAL OF SOFTWARE-EVOLUTION AND PROCESS* [online]. **34**(12), e2491. ISSN 2047-7473, 2047-7481. Dostupné z: doi:10.1002/smr.2491

LARMAN, Craig, 2004. *Agile and iterative development: a manager's guide* [online]. B.m.: Addison-Wesley Professional [vid. 2024-03-16]. Dostupné z: https://www.google.com/books?hl=cs&lr=&id=76rnV5Exs50C&oi=fnd&pg=PA1&dq=larman+c+(2004)+agile+and+iterative+development:+a+manager%E2%80%99s+guide.+addison-wesley,+boston&ots=ocU_XpESPZ&sig=P62b8AohRsB_AI-NY6eOIg2DzE4

LEACH, Lawrence P., 2000. *Critical chain project management*. Boston: Artech House. Artech House technology management and professional development library. ISBN 978-1-58053-074-3.

LEI, Howard, Farnaz GANJEIZADEH, Pradeep Kumar JAYACHANDRAN a Pinar OZCAN, 2017. A statistical analysis of the effects of Scrum and Kanban on software

development projects. *ROBOTICS AND COMPUTER-INTEGRATED MANUFACTURING* [online]. **43**, 59–67. ISSN 0736-5845, 1879-2537. Dostupné z: doi:10.1016/j.rcim.2015.12.001

MAHANTA, Prabal, Anil Kumar POLE, Vittalraya Shenoy ADIGE a M. RAJKUMAR, 2016. DevOps Culture and its impact on Cloud Delivery and Software Development. In: *IEEE International Conference on Advances in Computing, Communication and Automation (ICACCA)*: *2016 INTERNATIONAL CONFERENCE ON ADVANCES IN COMPUTING, COMMUNICATION AND AUTOMATION (ICACCA 2016)* [online]. New York: IEEE, s. 260–265 [vid. 2024-03-16]. ISBN 978-1-5090-0673-1. Dostupné z: https://www.webofscience.com/wos/alldb/full-record/WOS:000391239500051

MARTIN, Robert C., 2003. *Agile software development: principles, patterns, and practices*. Upper Saddle River, N.J: Prentice Hall. Alan Apt series. ISBN 978-0-13-597444-5.

MCGREAL, Don a Ralph JOCHAM, 2018. *The professional product owner: leveraging Scrum as a competitive advantage*. Boston: Addison-Wesley. The professional scrum series. ISBN 978-0-13-468647-9.

MONTGOMERY, Douglas C. a Connie M. BORROR, 2017. Systems for modern quality and business improvement. *QUALITY TECHNOLOGY AND QUANTITATIVE MANAGEMENT* [online]. **14**(4), 343–352. ISSN 1684-3703, 1811-4857. Dostupné z: doi:10.1080/16843703.2017.1304032

MORDI, Azuka a Mareike SCHOOP, 2021. Scaling with an Agile Mindset - A Conceptual Approach to Large-Scale Agile. In: *27th Annual Americas Conference on Information Systems (AMCIS)*: *DIGITAL INNOVATION AND ENTREPRENEURSHIP (AMCIS 2021)* [online]. Atlanta: Assoc Information Systems [vid. 2024-03-16]. Dostupné z: https://www.webofscience.com/wos/alldb/full-record/WOS:000672599802029

MOREIRA, Mario E., 2017. *The agile enterprise: building and running agile organizations*. Winchester, Mass, USA: Apress. ISBN 978-1-4842-2390-1.

MYSLÍN, Josef, 2016. *Scrum: průvodce agilním vývojem softwaru*. 1. vydání. Brno: Computer Press. ISBN 978-80-251-4650-7.

PIKKARAINEN, M., J. HAIKARA, O. SALO, P. ABRAHAMSSON a J. STILL, 2008. The impact of agile practices on communication in software development. *EMPIRICAL SOFTWARE ENGINEERING* [online]. **13**(3), 303–337. ISSN 1382-3256, 1573-7616. Dostupné z: doi:10.1007/s10664-008-9065-9

PROJECT MANAGEMENT INSTITUTE, ed., 2021. *A guide to the project management body of knowledge (PMBOK® guide) and the standard for projekt management*. Seventh Edition. Pennsylvania: Project Management Institute, Inc. ISBN 978-1-62825-664-2.

PROJECT MANAGEMENT INSTITUTE a AGILE ALLIANCE, ed., 2017. *Agile practice guide*. Newton Square, Pennsylvania: Project Management Institute, Inc. ISBN 978-1-62825-199-9.

QUADIENT, 2024a. About us | Quadient. *Quadient* [online] [vid. 2024-01-21]. Dostupné z: https://www.quadient.com/en/about-us

QUADIENT, 2024b. Quadient Company Overview | Quadient. *Quadient* [online] [vid. 2024-01-21]. Dostupné z: https://www.quadient.com/en/resources/quadient-company-overview

RIGBY, Darrell, Steve BEREZ a Sarah ELK, 2020. *Doing agile right: transformation without chaos*. Boston, MA: Harvard Business Review Press. ISBN 978-1-63369-870-3.

SAIER, Martin Christopher, 2017. Going back to the roots of W. A. Shewhart (and further) and introduction of a new CPD cycle. *INTERNATIONAL JOURNAL OF MANAGING PROJECTS IN BUSINESS* [online]. **10**(1), 143–166. ISSN 1753-8378, 1753-8386. Dostupné z: doi:10.1108/IJMPB-11-2015-0111

SHARMA, Shruti a Nitasha HASTEER, 2016. A Comprehensive Study on State of Scrum Development. In: P. N. ASTYA, A. SWAROOP, V. SHARMA a M. SINGH, ed. *IEEE International Conference on Computing, Communication and Automation (ICCCA)*: *2016 IEEE INTERNATIONAL CONFERENCE ON COMPUTING, COMMUNICATION AND AUTOMATION (ICCCA)* [online]. New York: IEEE, s. 867–872 [vid. 2024-03-16]. ISBN 978-1-5090-1666-2. Dostupné z: https://www.webofscience.com/wos/alldb/full-record/WOS:000393168500157

SHORE, James, Diana LARSEN, Gitte KLITGAARD a Shane WARDEN, 2022. *The art of agile development*. Second edition. Beijing : Boston: O'Reill. Theory in practice. ISBN 978-1-4920-8069-5.

SCHONBERGER, Richard J., 2007. Japanese production management: An evolution - With mixed success. *JOURNAL OF OPERATIONS MANAGEMENT* [online]. **25**(2), 403–419. ISSN 0272-6963, 1873-1317. Dostupné z: doi:10.1016/j.jom.2006.04.003

STELLMAN, Andrew a Jennifer GREENE, 2015. *Learning agile: understanding Scrum, XP, Lean, and Kanban*. 1st ed. Beijing Köln: O'Reilly. ISBN 978-1-4493-3192-4.

SUBAIR, Saad, 2014. The Evolution of Software Process Models: From the Waterfall Model to the Unified Modelling Language (UML). **3**, 2277–9825.

ŠOCHOVÁ, Zuzana, 2017. *The great scrummaster: #scrummasterway*. Boston: Addison-Wesley. The Addison-Wesley Signature Series. ISBN 978-0-13-465711-0.

ŠOCHOVÁ, Zuzana a Eduard KUNCE, 2019. *Agilní metody řízení projektů*. 2. vydání. Brno: Computer Press. ISBN 978-80-251-4961-4.

THESING, Theo, Carsten FELDMANN a Martin BURCHARDT, 2021. Agile versus Waterfall Project Management: Decision Model for Selecting the Appropriate Approach to a Project. In: M. M. CRUZCUNHA, R. MARTINHO, R. RIJO, N. MATEUSCOELHO, D. DOMINGOS a E. PERES, ed. *International Conference on ENTERprise Information Systems (CENTERIS) / International Conference on Project MANagement (ProjMAN) / International Conference on Health and Social Care Information Systems and Technologies (HCist)*: *INTERNATIONAL CONFERENCE ON ENTERPRISE INFORMATION SYSTEMS / INTERNATIONAL CONFERENCE ON PROJECT MANAGEMENT / INTERNATIONAL CONFERENCE ON HEALTH AND SOCIAL CARE INFORMATION SYSTEMS AND TECHNOLOGIES 2020 (CENTERIS/PROJMAN/HCIST 2020)* [online].

Amsterdam: Elsevier Science Bv, s. 746–756 [vid. 2024-03-16]. Procedia Computer Science. ISSN 1877-0509. Dostupné z: doi:10.1016/j.procs.2021.01.227

THOMAS, Joseph C. a Steven W. BAKER, 2008. Establishing an agile portfolio to align IT investments with business needs. In: G. MELNIK, P. KRUCHTEN a M. POPPENDIECK, ed. *AGILE 2008 Conference*: *AGILE 2008, PROCEEDINGS* [online]. Los Alamitos: IEEE Computer Soc, s. 252–258 [vid. 2024-03-16]. ISBN 978-0-7695-3321-6. Dostupné z: doi:10.1109/Agile.2008.29

TODARO, Dave, 2019. *The epic guide to Agile: more business value on a predictable schedule with Scrum*. North Hampton, NH: R9 Publishing LLC. ISBN 978-1-73300-040-6.

TURNER, Richard, 2004. Balancing Agility and Discipline: Evaluating and Integrating Agile and Plan-Driven Methods. *… of the 26th international Conference on …* [online]. [vid. 2024-03-16]. Dostupné z: https://www.academia.edu/18859063/Balancing_Agility_and_Discipline_Evaluating_and_Integrating_Agile_and_Plan_Driven_Methods

VAN MERODE, Henry, 2023. *Continuous Integration (CI) and Continuous Delivery (CD): a practical guide to designing and developing pipelines*. New York, NY: Apress. ISBN 978-1-4842-9227-3.

WEBER, D. W., 1999. CM strategies for RAD. In: J. ESTUBLIER, ed. *9th International Symposium on System Configuration Management*: *SYSTEM CONFIGURATION MANAGEMENT* [online]. Berlin: Springer-Verlag Berlin, LECTURE NOTES IN COMPUTER SCIENCE, s. 204–216 [vid. 2024-03-16]. ISBN 978-3-540-66484-0. Dostupné z: https://www.webofscience.com/wos/alldb/full-record/WOS:000170800400014

WILLIAMS, Laurie, 2010. Agile Software Development Methodologies and Practices. In: M. V. ZELKOWITZ, ed. *ADVANCES IN COMPUTERS, VOL 80* [online]. San Diego: Elsevier Academic Press Inc, s. 1–44 [vid. 2024-03-16]. ISBN 978-0-12-381025-0. Dostupné z: doi:10.1016/S0065-2458(10)80001-4

9 Seznam obrázků

10 Zadání práce z IS (eVŠKP)

Univerzita Hradec Králové
Fakulta informatiky a managementu

Zadání bakalářské práce

Autor:	**Daniel Hanák**
Studium:	I2100124
Studijní program:	B0688A140001 Informační management
Studijní obor:	Informační management
Název bakalářské práce:	**Vývoj softwaru pomocí agilních metodik**
Název bakalářské práce AJ:	Agile Software Development

Cíl, metody, literatura, předpoklady:

Cílem práce je objasnit rozdíly mezi teorií a praxí za použití agilních přístupů při vývoji softwaru.

Rámcová osnova

1. Úvod - motivace, cíl práce
2. Teoretická část - literární rešerše, pojmy, definice
3. Praktická část - představa vs realita fungování agilu, příklady používání agilních nástrojů, návrhy na zlepšení
4. Závěr

- DOLEŽAL, Jan. Agilní přístupy vývoje produktu a řízení projektu: komplexně, prakticky a dle světové praxe. Praha: Grada, 2022. ISBN 978-80-271-3705-3.
- ŠOCHOVÁ, Zuzana a Eduard KUNCE. Agilní metody řízení projektů. 2. vydání. Brno: Computer Press, 2019. ISBN 978-80-251-4961-4.
- MYSLÍN, Josef. Scrum: průvodce agilním vývojem softwaru. Brno: Computer Press, 2016. ISBN 978-80-251-4650-7.

Zadávající pracoviště:	Katedra informačních technologií, Fakulta informatiky a managementu
Vedoucí práce:	prof. Ing. Vladimír Bureš, Ph.D., MBA
Datum zadání závěrečné práce:	15.10.2021