Univerzita Hradec Králové
Fakulta informatiky a managementu
Katedra informačních technologií

Agilní metodiky

Metodika SAFe a její integrace v rámci ŠKODA AUTO, a.s.

Diplomová práce

Autor: Bc. Lukáš, Smetana
Studijní obor: IM2-p

Vedoucí práce: RNDr. Petr Tučník, Ph.D.

Odborný konzultant: Ing. David Žid,
ŠKODA AUTO a.s.

Hradec Králové listopad 2019

Prohlášení:

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

vlastnoruční podpis

V Hradci Králové dne 28.11.2019 Bc. Lukáš Smetana

Poděkování:

Děkuji vedoucímu diplomové práce RNDr. Petrovi Tučníkovi, Ph.D. za metodické vedení práce a Ing. Davidovi Židovi ze společnosti ŠKODA AUTO a.s. za odborné rady a konzultace. Dále také děkuji své rodině a blízkým za podporu při studiu.

Anotace

Práce je zaměřena na agilní metodiky vývoje softwaru a trend posledních let, kterým je škálování těchto metodik ve velký společnostech a na velkých projektech. Práce předkládá teoretické vymezení agilních metodik porovnaných s metodikami, které jsou označovány jako tradiční. Dále jsou popsány tzv. large-scale agilní metodiky a jsou představeny nejznámější z nich. Praktický přínos práce je představení případové studie implementace metodiky Scaled Agile Framework do vybraného vývojového týmu společnosti ŠKODA AUTO a.s. a analýza rozdílů vnitropodnikového metodického rámce s metodikou Scaled Agile Framework. Součástí praktické části je přehled doporučení pro společnost ŠKODA AUTO a.s., která vyplývají z případové studie a analýzy rozdílů, pro budoucí implementace large-scale agile metodik.

Annotation

Title: Agile methodologies.

This thesis is focused on agile software development methodologies and the trend of recent years, which is the scaling of these methodologies in large companies and large projects. The thesis presents a theoretical definition of agile methodologies compared to those that are called traditional methods. Furthermore, the so-called large-scale agile methodologies are described and the best-known ones are introduced. The practical benefit of this thesis is a case study of the implementation of the Scaled Agile Framework methodology into a selected development team in ŠKODA AUTO a.s. and analysis of differences in the internal methodological framework with the Scaled Agile Framework. Finally, the paper presents recommendations for ŠKODA AUTO a.s., resulting from case studies and differences analysis, for the future implementation of large-scale agile methodologies.

Obsah

Seznam obrázků

Seznam tabulek

1 Úvod

V roce 2001 byl poprvé zveřejněn tzv. Agilní Manifest („Manifesto for Agile Software Development“, 2001). Tento manifest předních odborníků v oblasti softwarového inženýrství reagoval na situaci, která panovala ve vývojových týmech, softwarových společnostech a obecně v oblasti IT. Díky stále rychleji měnícímu se okolnímu světu, se rychle měnily také představy a požadavky zákazníků a vedení firem na vyvíjený software. Značná neefektivita, vysoké náklady, začínající nedostatek programátorů, chybějící metodické vedení. To všechno byly důvody, které daly za vznik novému pohledu na metodiku vývoje softwaru. A tak vznikla agilní filosofie a přístup, který směřoval vývojové týmy k tomu být flexibilnější, pružnější a otevřenější častějším změnám v zadání projektů, specifikací a funkcionalit, nebo třeba technologií. Agilní principy pomáhaly zvyšovat efektivitu a usnadňovaly spolupráci vývojových týmů se zákazníkem.

Dnes, téměř 19 let po vydání tohoto manifestu není agilita otázkou pouze programátorských týmů zaměřených na vývoj softwaru. Společnosti všech druhů a rozměrů se snaží adoptovat agilní metodiky i mimo vývojové týmy. Nutí je k tomu velice dynamické prostředí dnešní digitální ekonomiky. K udržení konkurenceschopnosti musí firmy pružně reagovat na technologické změny a změny požadavků zákazníků.

Agilní metody, které byly zaváděny do čistě softwarových společností, nebo do menších vývojových týmů, nejsou dnes již jediným klíčem k úspěchu. Pokud chce být společnost agilní, musí změnit svou kulturu, způsob práce a procesy napříč celou společností. Samotné implementování konkrétní metodiky do konkrétního týmu nezaručí dodání očekávaných hodnot.

Jednou ze společností, která prochází transformací a čelí výzvám spojeným s digitální ekonomikou a změnou vnitřního nastavení je i společnost ŠKODA AUTO a.s. (dále jen ŠKODA AUTO). Ta v dnešní době už dávno není pouze montovnou automobilů, ale mnohem více je technologickou společností s velkým důrazem na

moderní technologie ve svých automobilech. Různé typy a velký počet používaných softwarů a systémů se prolíná napříč celou společností. Jde o systém pro správu zaměstnaneckých účtů personálního oddělení, až po aplikaci eCall v autech, která dokáže přivolat pomoc v případě nehody automobilu. Moderní technologie jsou implementovány do všech oblastí, samotných automobilů, nebo je pomocí nich řízen pohyb materiálu ve výrobě a analyzována výrobní data. Digitální transformace společnosti je zkrátka vidět na každém kroku.

V teoretické části se tato práce zabývá objektivní analýzou, popisem a srovnáním nejrozšířenějších metodik projektového řízení a produktového vývoje a porovnává přístupy, které jsou označovány jako tradiční, s těmi, kterým se říká agilní. Agilní metodiky jsou navíc rozšířeny o část tzv. large-scale agile metodik, tedy agilní metodiky vhodné pro velké projekty a společnosti.

Praktická část je rozdělena na dvě části. První částí je popis prostředí produktového vývoje ve ŠKODA AUTO. Dále je provedena analýza rozdílů mezi vnitropodnikovým metodickým rámcem a rozšířenou metodikou Scaled Agile Framework (zkráceně SAFe), kterou ŠKODA AUTO vybrala jako jednu z metodik vhodných pro škálování agilních principů. Tato analýza je doplněna o případovou studii probíhající agilní transformace vybraného oddělení společnosti ŠKODA AUTO.

2 Cíl práce

Cílem této práce je teoretické srovnání tradičních a agilních metodik a stejně tak popis agilních metodik, které jsou využívané na velké projekty a obecně při škálování agilních principů ve větších organizacích. V této části je kladen důraz na popis metodického rámce Scaled Agile Framework, jeho hlavní principy, role, události a procesy.

Cílem praktické části práce je vytvořit ucelenou analýzu integrace metodiky Scaled Agile Framework v prostředí společnosti ŠKODA AUTO. Tato analýza se skládá z případové studie a analýzy rozdílů vnitropodnikového metodického rámce s metodikou Scaled Agile Framework. Případová studie je zaměřena na vybrané oddělení společnosti ŠKODA AUTO, které je v procesu implementování metodiky Scaled Agile Framework. Cílem je tuto probíhající transformaci analyzovat a definovat silná a slabá místa. Cílem analýzy rozdílů je definovat základní rozdíly mezi těmito dvěma metodikami. Na základě poznatků z těchto dvou částí je cílem identifikovat a definovat doporučení pro ostatní oddělení a týmy společnosti, kterých se v budoucnosti implementace metodiky Scaled Agile Framework bude týkat.

2.1 Metodika zpracování

Teoretická část je pojata jako literární rešerše z domácích i zahraničních zdrojů. Praktická část je vypracována na základě konzultací se zástupci společnosti ŠKODA AUTO a analyzování odpovědí na předem definované otázky. Cílem těchto konzultací bylo analyzovat probíhající agilní transformaci. Tyto otázky byly formou konzultací zodpovězeny několika pracovníky ŠKODA AUTO. Doplňující informace byly získány z různých podnikových materiálů jako jsou prezentace a konkrétní směrnice, či metodické pokyny.

3 Softwarové inženýrství a tradiční metody vývoje softwaru

Software je dnes všude kolem nás. Algoritmy a počítačové systémy nás ovlivňují každou sekundu našeho života. Přes jednoduché algoritmy v domácích elektronických spotřebičích až po ty komplikované, které řídí finanční burzy a veškeré procesy světové ekonomiky. Ať už se jedná o datové analyzování sociálních sítí za účelem reklamy, nebo algoritmus budíku, jedno má software společné. Je vytvářen lidmi. Proto než se pustíme do definování přístupů a metodik vývoje softwaru je potřeba definovat software a softwarové inženýrství jako takové.
Kadlec (Kadlec, 2004) ve své knize uvádí definici softwarového inženýrství podle Fritz Bauera.

„Softwarové inženýrství je zavedení a používání řádných inženýrských principů tak, abychom dosáhli ekonomické tvorby softwaru, který je spolehlivý a pracuje účinně na dostupných výpočetních prostředcích" (Kadlec, 2004).

Tato definice je velmi obecná, a proto je i dnes stále platná. Na rozdíl od své definice, se obor softwarového inženýrství za dobu své existence hodně změnil. Dnes již nejde o obor několika zapálených specialistů, ale o odvětví ekonomiky, které zaměstnává miliony lidí a na hrubém domácím produktu zemí se projevuje nezanedbatelným procentem (Somerville, 2013).

3.1 Historie softwarového inženýrství

Vývoj software a softwarové inženýrství je mladá větev inženýrských oborů. Za počátek softwarového inženýrství se dá označit první polovina 60. let dvacátého století. V té době se podoba vývoje software velmi lišila od té dnešní. Programování a obsluha počítače byla disciplína pro velmi úzce zaměřený okruh velmi vzdělaných lidí. S rozvojem hardware a klesajícími cenami počítačů se k programování a tvorbě software dostává více a více lidí. Termín „softwarové inženýrství" se neskloňuje pouze mezi vědci a úzce zaměřenými specialisty, ale také mezi běžnou populací a v roce 1969 se objeví jako téma na konferenci NATO (Kadlec, 2004). S tímto

rozvojem přicházejí první náznaky problémů projevujících se v nízké úspěšnosti dodávek softwaru a nedokončených projektech. Ty jsou zpravidla řízeny velmi chaoticky. Začínají se objevovat první náznaky metodik vývoje softwaru, aby se těmto problémům čelilo. V 80. letech se již dá o softwarovém inženýrství mluvit jako o zavedeném oboru i přesto, že USA jej uzná jako obor s certifikátem až v roce 1997 (Somerville, 2013; Pressman, 2010).

3.2 Softwarová krize

Tímto pojmem je označováno období konce šedesátých let minulého století, kdy firmy začaly ve větší míře pronikat do softwarového inženýrství a projektů značně přibývalo. Tyto projekty na dodávku softwaru však většinou nebyly dokončeny, termíny dokončení se stále vzdalovaly a celková efektivita práce programátorů a kvalita dodávaných systémů byla nízká. Jako mix příčin se uvádí špatná komunikace mezi vývojovým týmem a klientem a také uvnitř vývojového týmu (nebo firmy), nesprávné odhady a obecně špatné plánování a nízká produktivita práce. Tím, že byl obor stále v ranné fázi vývoje, projekty doprovázely také problémy spojené se špatně zvolenými technologiemi.

„Pro ilustraci softwarové krize bývá uváděna statistika z její doby, podle níž ze všech zakázek pro americkou vládu, které měly hodnotu mnoha milionů dolarů, bylo přes 40 % zakázek dodáno, ale nikdy nebylo úspěšně použito, zhruba 25 % zaplaceno, leč nebylo nikdy dodáno, asi 15 % bylo po drastických úpravách zahozeno. Pouze necelá tři procenta projektů byla po úpravách použita a jen asi dvě procenta produktů mohla být použita beze změny.“ (Kadlec, 2004, s. 27)

Toto období, ač znamenalo spoustu těžkostí a „vyhozených peněz“, přineslo jednu podstatnou věc. Manažeři společností a přední odborníci v oblasti programování si uvědomili potřebu definování standardizovaných postupů při vývoji. Postupně začaly vznikat nejrůznější metody a postupy projektového řízení navržené pouze pro vývoj softwaru a jeho životní cyklus (Somerville, 2013; Myslín, 2016).

3.3 Proces vývoje softwaru

Proces vývoje softwaru je souhrnný pojem pro všechny aktivity vedoucí k vytvoření softwarového produktu. Toto jsou čtyři klíčové aktivity softwarového inženýrství, které se v různých podobách a určité formě vyskytují ve všech softwarových procesech (Somerville, 2013):

- Specifikace
- Návrh a implementace
- Validace
- Evoluce (údržba / rozšiřování)

V předchozí kapitole bylo popsáno období rozvoje softwarového inženýrství a období, kterému se někdy přezdívá „softwarová krize“. Toto období dalo vzniknout prvním uceleným procesům vývoje softwaru. Je logické, že se modely procesů a metodiky vývoje softwaru, které v tomto období vznikaly (přibližně sedmdesátá až osmdesátá léta minulého století) zaměřovaly hlavně na specifikace a analyzování zákaznických požadavků, protože jejich absence byla jedním z hlavních důvodů velké neúspěšnosti projektů. (Kadlec, 2004)

3.4 Tradiční vývoj softwaru

Existuje několik metodik vývoje softwaru, které jsou považovány a označovány jako tradiční. V základu se jedná o tyto metody (Kadlec, 2004):

- Vodopádový model
- Spirálový model
- Unified Software Development Process (používá se pouze Unified Process nebo zkráceně UP)

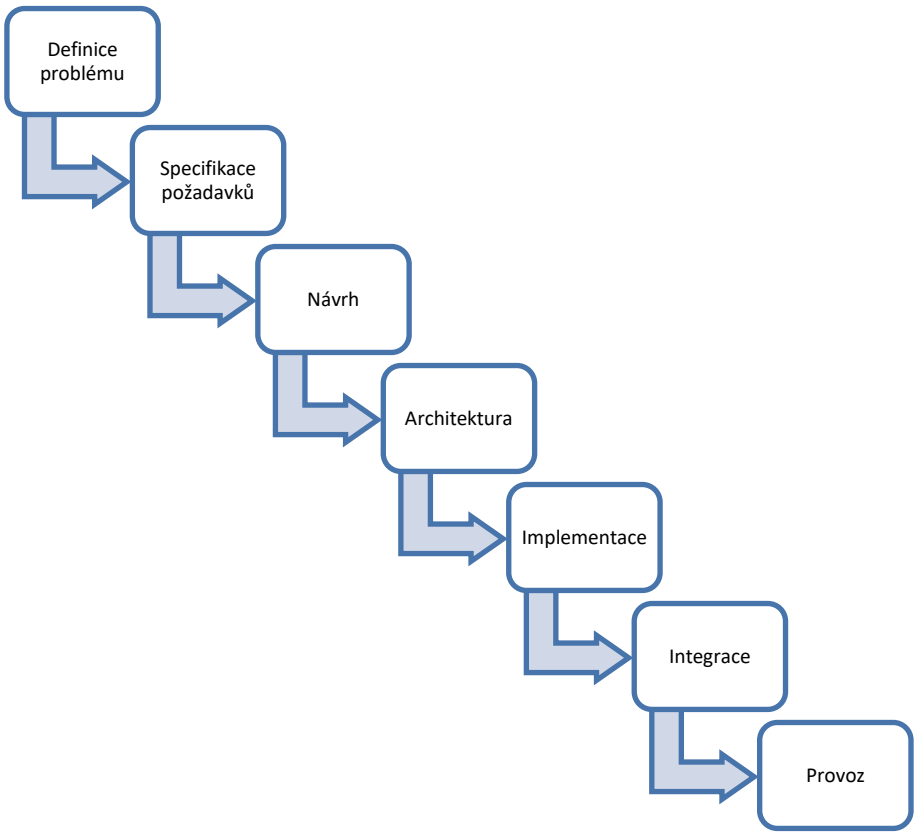
V odborné literatuře nebo na webu se dá najít velké množství dalších tradičních metodik, které jsou postaveny na těchto třech zmíněných, některé je kombinují, nebo rozšiřují. Pro základní přehled a principy tradičních metod budou popsány pouze tyto tři metodiky.

Obecně pro tradiční metodiky platí:

- sekvenčnost kroků
- důraz na dokumentaci
- úplné definování požadavků na začátku projektu.

3.5 Vodopádový model

Název tohoto modelu je odvozen od kaskádovitých kroků procesu, které připomínají vodopád. Vodopádový model vznikl jako reakce na nedostatečné metodické vedení vývoje softwaru v sedmdesátých letech. Je to velice jednoduchý model, který má několik po sobě jdoucích základních fází, tak, jako například proces výroby automobilu. (obrázek 1).



Obrázek 1 Schéma vodopádového modelu
Zdroj: Vlastní tvorba podle (Myslín 2016, s. 25)

Po dokončení jedné fáze se model přesouvá do další, nelze tak ale učinit, pokud předchozí fáze nebyla stoprocentně dokončena. Model klade důraz na

strukturovaný pokrok mezi definovanými fázemi. Musí být dokončeny všechny předem naplánované aktivity, až pak může přijít na řadu další. Například po úplném dokončení kódu (fáze implementace) může následovat testování (fáze integrace).

Silné stránky a vhodnost

Vodopádový model se hodí na jednoduché projekty menšího rozsahu, do kterých zasahuje menší počet spolupracujících stran. Je také vhodný na projekty, u kterých jsou pevně definovány nároky a požadavky a je velice malá šance, že se specifikace bude v průběhu měnit. Za silnou stránku se dá označit i jeho jednoduchost a jednoznačnost. Projekty vedené vodopádově jistě nepotřebují certifikované projektové manažery, protože každý jednoduše pochopí jeho principy a je jednoduché orientovat se ve fázích modelu. Vodopádový model může najít uplatnění i u projektů, které jsou prováděny například pro orgány státního aparátu, kde jsou tradičně vyžadovány rozsáhlé dokumentační materiály (Pressman, 2010).

Nevýhody

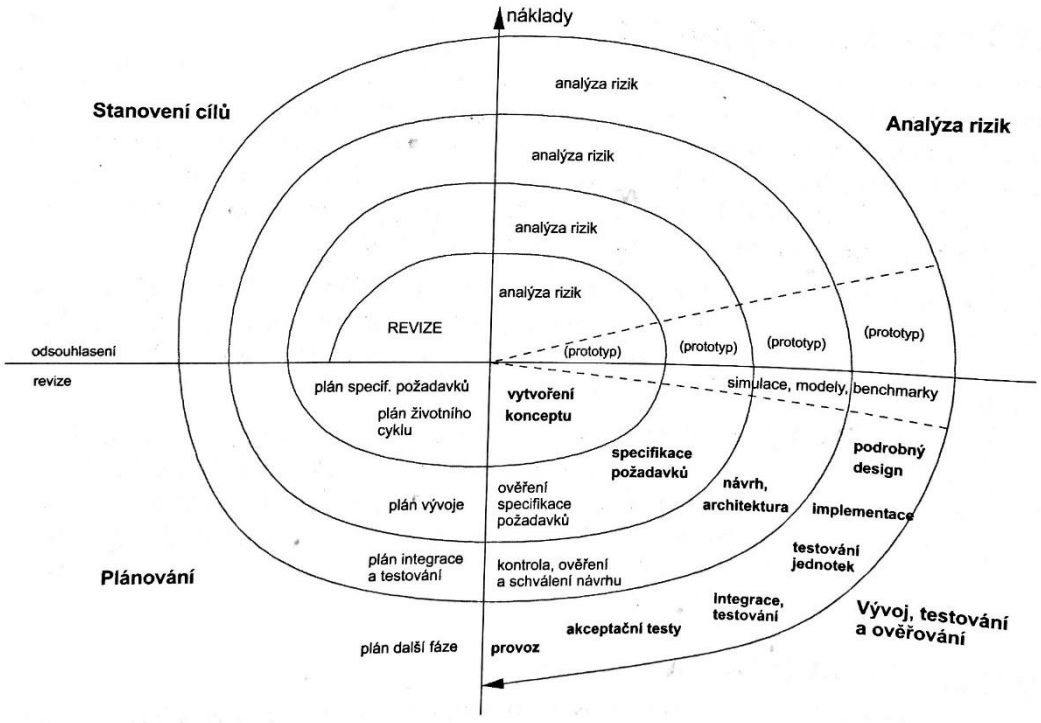
Je logické, že vodopádový model nereflektuje změny, které na poli softwarového inženýrství nastaly, a proto je pro dnešní dobu většinou nevhodný. Model je velice nepružný a nemá mechanismy, které by umožňovaly časté změny. Zákazníkovi není umožněno jakýmkoliv způsobem zasahovat do vývoje, kromě první fáze analýzy potřeb a definování požadavků. Hotová řešení se potom mohou často lišit od aktuálních potřeb klienta, protože ty se mohou v čase měnit. Je zde velký časový rozptyl mezi zadáním a viditelným produktem. Díky pevně dané sekvenčnosti kroků, je také problematické odstraňování chyb. Model je velice nevhodný pro rozsáhlé projekty, do kterých vstupuje velké množství stran, exponenciálně pak narůstá potřeba koordinace a řízení, která může v pokročilé fázi téměř vytlačit vlastní vývoj softwaru (Pressman, 2010).

3.6 Spirálový model

Značné nedostatky sekvenčního vodopádového modelu se snažil vyřešit Barry Boehm ve svém článku v roce 1988 (Boehm, 1988). Ten navrhl riziky řízený spirálový model, který vyvíjel na základě zkušeností s vodopádovými metodami

aplikovanými na velké vládní softwarové projekty. Nové principy, které tento model přináší, jsou iterativní přístup a opakovaná analýza rizik. V modelu jsou čtyři hlavní fáze, kterými projekt prochází stále dokola, tak jako zobrazená spirála. Tyto fáze jsou (Boehm, 1988):

- Stanovení cílů – jsou stanoveny konkrétní cíle projektu
- Posuzování a snižování rizik – klíčová rizika jsou identifikována a informace jsou použity ke snížení těchto rizik
- Vývoj a validace – pro další fázi je vybrán vhodný model vývoje
- Plánování – projekt je přezkoumáván a plány jsou zpracovávány pro další kolo spirály



Obrázek 2 Spirálový model
Zdroj: (Kadlec, 2004, s.68)

Silné stránky a vhodnost

Tím, že spirálový model reagoval na nedostatky vodopádového modelu, je zde hodně nedostatků vodopádu bráno jako silné stránky. Spirálový model je vhodný

k použití v rozsáhlých projektech, které se navíc často mění. Vlastnost iterace pomáhá řešit problémy, které se objevují až v průběhu vývoje. Model je vhodný pro nasazení ve společnostech, které vyvíjí vlastní produkt, méně vhodný je pro zákaznický vývoj (Myslín, 2016).

Nevýhody

Tak jako u vodopádového modelu je hlavní nevýhoda spirálového modelu dodání hotového projektu až po dokončení poslední fáze (poslední spirály). Neřeší tedy problém v dlouhé době dodání. Další otázkou je komplexita a možná přílišná byrokratická zátěž spojená s používáním tohoto modelu u menších projektů (Kadlec, 2004).

3.7 Unified Process

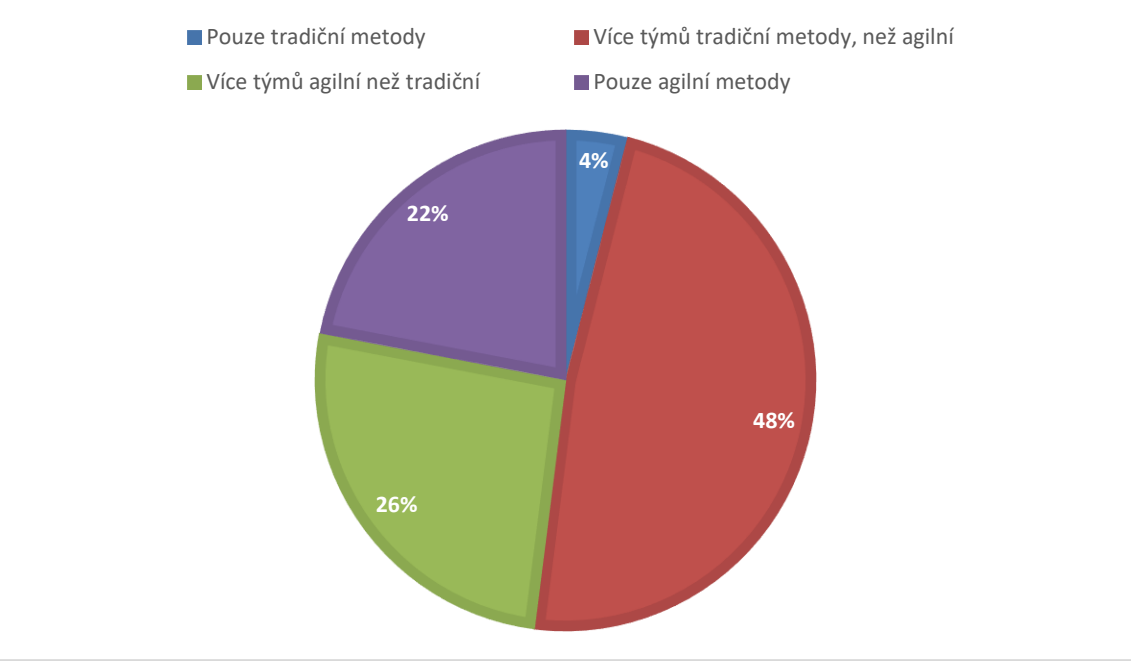
Unified Process není pouze proces nebo model, jak by mohl název napovídat. Jedná se mnohem více o ucelený rámec s více procesy, definovanými rolemi a nástroji. Jedná se o volně dostupnou verzi RUP (Rational Unified Process), na jehož základech byl UP o několik let později vyvinut. UP je tak jako i předchozí tradiční metody procesně orientovaná a má předpokládané fáze. Je to velice rozsáhlá a propracovaná iterativní metodika vývoje softwaru. UP je propojen s modelovacím jazykem UML, který mezi vývojáři nalezl široké uplatnění. UML se zaměřuje především na architekturu navrhovaného systému, a tak není překvapením, že i UP je na tuto oblast velice orientovaný. Vývoj softwaru pod Unified Process prochází iteracemi, které typicky zahrnují tyto oblasti (Kadlec, 2004):

- Plánování
- Analýza a návrh
- Implementace
- Testování a integrace
- Dodání

Dále je projekt v metodice UP rozdělen do čtyř fází. Fáze zahájení, projektování, realizace, předání. Každá fáze obsahuje několik iterací a má svoje mezníky (Kadlec, 2004).

3.8 Využití tradičních metod v dnešním světě

Tradiční metody v dnešní době stále hrají při vývoji softwaru velkou roli. I přesto, že iniciativy spojené s přechodem na agilní metody fungují již několik let. Společností, které používají stále pouze tradiční metody je velmi málo. Dotazníkové šetření State of Agile americké společnosti Version One, která se průzkumy v oblasti vývoje softwaru zabývá již od roku 2006, říká, že pouze 3 % firem v roce 2018 neuplatňovaly agilní metody vývoje softwaru (Version One, 2019). Zbytek firem implementoval v různém rozsahu agilní metody. Avšak podle statistiky, do které se zapojilo na 1300 respondentů, pouze 22 % z těchto firem udává, že všechny jejich vývojové týmy jsou řízeny agilně (Version One, 2019). V praxi to proto znamená, že uvnitř společností mohou některé týmy pracovat agilně a jiné mohou svůj projekt stále řídit tradičně. Což je logické, protože společnosti mají projekty různé velikosti a různé komplexity a přechod na agilní metody nemusí vždy dávat smysl. Některé důvody nevhodnosti agilních metodik budou popsány v další kapitole.



Obrázek 3 Podíl tradičních a agilních metod uvnitř společností
Zdroj: Vlastní zpracování podle (Version One, 2019)

3.9 Potřeba agility

Jestliže Kadlec (Kadlec, 2004) ve své publikaci, která je jedním z prvních českých zdrojů věnujících se agilním metodikám, připisuje rychle měnícímu se světu zásluhu na vzniku agilních metodik, musíme dnes potvrdit tuto myšlenku. Rychle měnící se svět, rychle měnící se požadavky klientů a technologických společností, jsou stále jedním z hlavních důvodů tolik potřebné agility. V poslední době k tomu také velice přispívá tlak a situace na trhu práce. Speciálně v oblasti informačních technologií, a tedy vývoje softwaru, je dlouhodobý nedostatek lidí. Podle statistiky Českého Statistického Úřadu z roku 2018 počet ICT specialistů za pět let vzrostl o polovinu. Statistika dále říká, že poptávka po ICT odbornících rok od roku roste. Mezi lety 2014 a 2017 stoupl podíl firem, které již nějaké ICT odborníky zaměstnávají a zároveň mají problém najít další vhodné kandidáty, o 12 procentních bodů („ICT odborníci v České republice“, 2018).

Další důvody přechodu na agilní metody popisuje Cottemeyer (Cottemeyer, 2011):

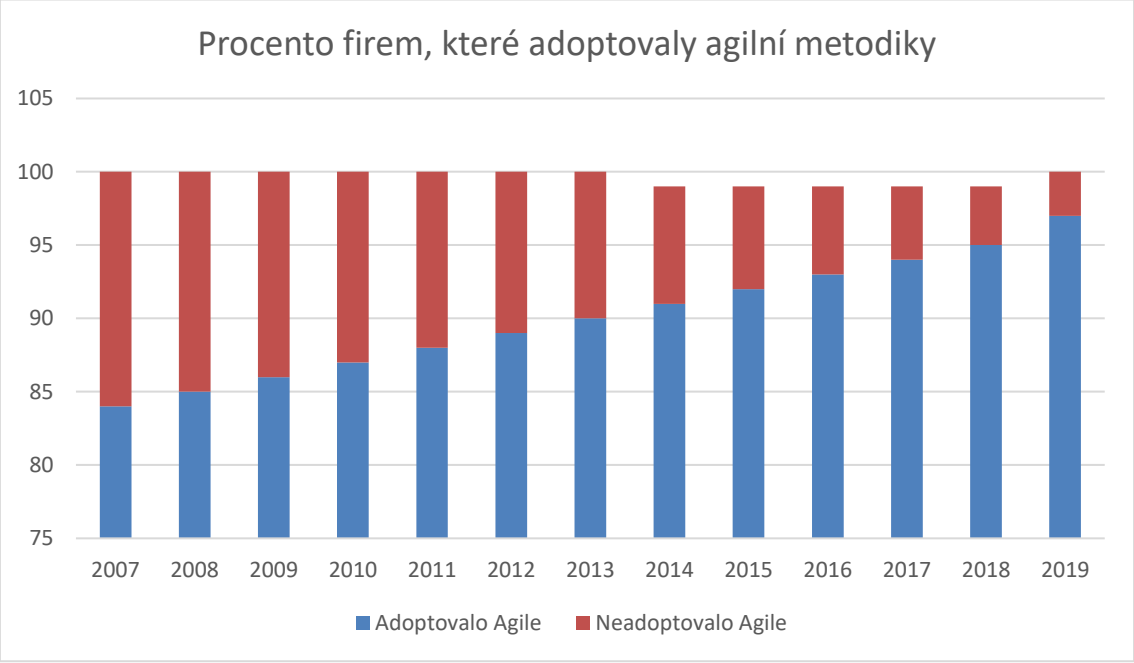
- Rychlejší dodání produktu na trh
- Přímá zpětná vazba od zákazníků
- Tvorba správného produktu
- Brzké snižování rizik
- Vyšší kvalita
- Vyšší efektivita

4 Agilní vývoj softwaru

V této sekci jsou uceleně shrnuty informace a principy agilních metodik, vycházejících z Agilního Manifestu („Manifesto for Agile Software Development“, 2001). Dále jsou popsány známé agilní metodiky, jakožto alternativy k tradičnímu vývoji softwaru. Poslední část této kapitoly se zabývá přenosem agilních principů z menších do větších projektů a celofiremních prostředí a představení několika metodik na to přizpůsobených. Pro tento proces adoptování agilní metodiky ve velkém se zažil termín „large-scale“.

Společnosti se snaží být agilní, aby mohly doručovat kvalitnější produkty za kratší čas s menším úsilím. Toto však není možné bez agilní firemní kultury. Podle Kunce a Šochové (Kunce, Šochová, 2014) nestačí certifikovaný kouč, ale právě celofiremní filosofie k tomu, aby firma byla agilní.

Obrázek číslo 4 demonstruje odklon firem od tradičního vývoje softwaru a adopci moderních agilních technik, jak bude později demonstrováno, nejen pro vývoj softwaru.



Obrázek 4 Vzestup agilních metod
Zdroj: Vlastní zpracování podle (Version One, 2007-2019)

Nelze opomenout fakt, že oproti tradičním metodám práce v agilním prostředí přináší pracovníkům lepší pocit a větší radost z práce (Šochová, Kunce, 2014).

4.1 Manifest Agilního vývoje softwaru

Principy agility vycházejí z Agilního Manifestu („Manifesto for Agile Software Development“, 2001). Ten byl sepsán sedmnácti odborníky v oblasti softwarového inženýrství v roce 2001 v lyžařském resortu Snowbird v Utahu, USA. Tento manifest reagoval na nefunkční prostředí vývoje softwaru, vycházející často z tradičních metodik. Společnosti nadměrně plánovaly a dokumentovaly cykly vývoje softwaru, a ztratily tak ze zřetele to, na čem opravdu záleželo – uspokojení zákazníků kvalitním softwarovým produktem. Agilní manifest se skládá pouze z 68 slov (v anglické originále) a dvanácti principů, nejedná se o žádný složitý dokument, jak by se možná dalo očekávat. Původní webová stránka agilemanifesto.com byla k dnešnímu dni přeložena do více než padesáti jazyků („Manifesto for Agile Software Development“, 2001). České znění manifestu, převzaté z originálního webu je k vidění na obr. číslo 5.

Manifest agilního vývoje

Objevujeme lepší způsoby vývoje software tím, že jej tvoříme a pomáháme při jeho tvorbě ostatním. Při této práci jsme dospěli k těmto hodnotám:

Jednotlivci a interakce před procesy a nástroji
Fungující software před vyčerpávající dokumentací
Spolupráce se zákazníkem před vyjednávání o smlouvě
Reagování na změny před dodržením plánu

Jakkoliv jsou body napravo hodnotné, bodů nalevo si ceníme více.

Obrázek 5 Manifest agilního vývoje
Zdroj: Vlastní zpracování podle („Manifesto for Agile Software Development“, 2001)

Tyto čtyři hlavní hodnoty jsou dále autory rozvedeny do dvanácti principů. Prvním a nejdůležitějším principem je důraz na uspokojení klienta. Klientem může být i obchodní oddělení uvnitř společnosti. Všechny další principy jsou tomuto podřízené, nebo představují nástroje k jeho dosažení. Na tomto bodu lze ilustrovat hlavní rozdíl oproti tradičním metodikám, kde bylo plánování celého softwarového procesu na první místě. Oproti tradičním metodám také principy v agilním manifestu hovoří o pravidelném dodávání funkčního softwaru v krátkých časových intervalech, ve kterých jsou vítány požadavky na změny. Tento princip je součástí i tzv. inkrementálního vývoje a spirálového modelu. Dodaný software je jediným měřitelným pokrokem. Všech dvanáct principů, tak jak jsou definovány v manifestu pro agilní vývoj, je zde („Manifesto for Agile Software Development“, 2001):

1. Nejvyšší prioritou je vyhovět zákazníkovi časným a průběžným dodáváním hodnotného softwaru.
2. Změny v požadavcích jsou vítány, a to i v pozdějších fázích vývoje. Agilní procesy podporují změny vedoucí ke zvýšení konkurenceschopnosti zákazníka.
3. Dodáván je fungující software v intervalech týdnů až měsíců, s preferencí kratší periody.
4. Lidé z byznysu a vývoje musí spolupracovat denně po celou dobu projektu.
5. Projekty jsou budovány kolem motivovaných jednotlivců. Je jim vytvářeno prostředí, podporovány jejich potřeby a je jim důvěřováno, že odvedou dobrou práci.
6. Nejúčinnějším a nejefektnějším způsobem sdělování informací vývojovému týmu z vnějšku i uvnitř něj je osobní konverzace.
7. Hlavním měřítkem pokroku je fungující software.
8. Agilní procesy podporují udržitelný rozvoj. Sponzoři, vývojáři i uživatelé by měli být schopni udržet stálé tempo trvale.
9. Agilitu zvyšuje neustálá pozornost věnovaná technické výjimečnosti a dobrému designu.
10. Jednoduchost-umění maximalizovat množství nevykonané práce-je klíčová.

11. Nejlepší architektury, požadavky a návrhy vzejdou ze samo-organizujících se týmů.
12. Tým se pravidelně zamýšlí nad tím, jak se stát efektivnějším, a následně koriguje a přizpůsobuje své chování a zvyklosti.

Tvůrci Agilního Manifestu jsou („Manifesto for Agile Software Development“, 2001): Kent Beck, 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. I v této práci je možné si všimnou, že velké množství publikací, které se věnují agilním metodám pochází od někoho z původních tvůrců tohoto manifestu.

4.2 Co je agilní vývoj softwaru

Být agilní znamená být dynamický a přizpůsobivý, být interaktivní a dokázat rychle reagovat na změnu, která přichází. Je to vlastnost, která je v dnešní době velice žádaná. Agilní metody vývoje softwaru jsou často popisovány jako hravé, zbavené nudných procesů a otravné byrokracie. Agilnímu týmu jsou vytyčeny hranice, nejsou mu přesně definovány jednotlivé směry (procesy), uvnitř těchto hranic se tým může chovat tak, jak uzná za vhodné, s cílem doručení produktu. Nejedná se o chaos, ale spíše o pravidla hry, na rozdíl od pevně stanovených procesů. Agilní metodiky upřednostňují reálný výsledek, kterého se dá dosáhnout častými změnami před nedodělaným produktem, který následoval předem vytyčený plán. V agilním prostředí je plánování důležitá součást, ale je potřeba ho často opakovat, měnit, vracet se k němu. Není agilní následovat jednou vytvořený plán beze změn (Myslín, 2016).

V oblasti IT systémů nemusí být agilní jen samotný vývoj softwaru. Tato problematika se úzce prolíná také s agilním a štíhlým provozem a údržbou IT systémů (Procházka, Klimeš, 2011).

4.2.1 Přehled metodik

Z principů agilního manifestu vychází nemalé množství metodik a návodů, jak tyto principy ve firmě adaptovat. Některé se navzájem doplňují, některé slouží pouze jako nástroj určité aktivity. Cílem této kapitoly je obecný popis některých z nich. Podle již několikrát zmíněného průzkumu společnosti Version One může být uveden seznam nejpoužívanějších agilních metodik v roce 2018 v byznysovém prostředí. Jsou jimi (Version One, 2019):

- Scrum
- XP – extrémní programování
- Kanban
- Iterative development
- Scrum/XP hybrid
- Scrumban
- Lean startup

Podle průzkumu vévodí agilním metodikám jasně Scrum a dohromady s hybridní verzí, která je spojena s praktikami s XP, tzv. Scrum-XP hybrid jsou využívány ve 64 % organizací. Nezanedbatelná je také část firem, která používá mix několika metodik dohromady (14 %). Ostatní metodiky jako Kanban nebo Lean Startup, jsou v menšině (Version One, 2019).

Další metodiky, které v této statistice mohou být schované pod zmíněných 14 % nebo je dotazovaní neuvedli, protože tvoří pouze doplněk Scrum a rozšiřují ho, jsou (Somerville, 2013):

- Test Driven Development
- Feature-Driven Development
- Crystal
- Adaptive Software Development
- a další

4.3 Scrum

První metodika, která bude popsána je Scrum. Ten je v dnešní době široce rozšířen (Version One, 2019) a jeho počátky vznikly ještě před definováním agilního manifestu roku 2001. Tvůrci metodiky Scrum jsou Ken Schwaber a Jeff Sutherland (Sutherland, 2014). Oba tito pánové přispěli svými myšlenkami i k již zmíněnému agilnímu manifestu.

Scrum je jednoduchý a poměrně stručný metodický rámec pro řízení komplexních projektů. Definuje několik málo procesů a praktiky, týmové role a další artefakty a má pouze několik daných zákonitostí. Díky tomu, že není rozsáhlý, Scrum je velice lehký na naučení a pochopení. Scrum nepředepisuje procesy a strategie na každou možnou situaci, je modelovaný na komplexní projekty, ve kterých se nedá naplánovat vše do posledního detailu, a tak se víc než na pevně definované předpisy, co a jak dělat v jaké situaci, spoléhá na selský rozum a několik základních pravidel, která mohou být použity jako vodítka v komplexních situacích.

Paradoxně z důvodu své jednoduchosti, může být adopce Scrumu velmi složitá. Pokud se tým, který je zvyklý na direktivní přístup shora, ve smyslu „udělej toto, pak tamto“, rozhodne implementovat Scrum, může mít ze začátku více problémů než užitku.

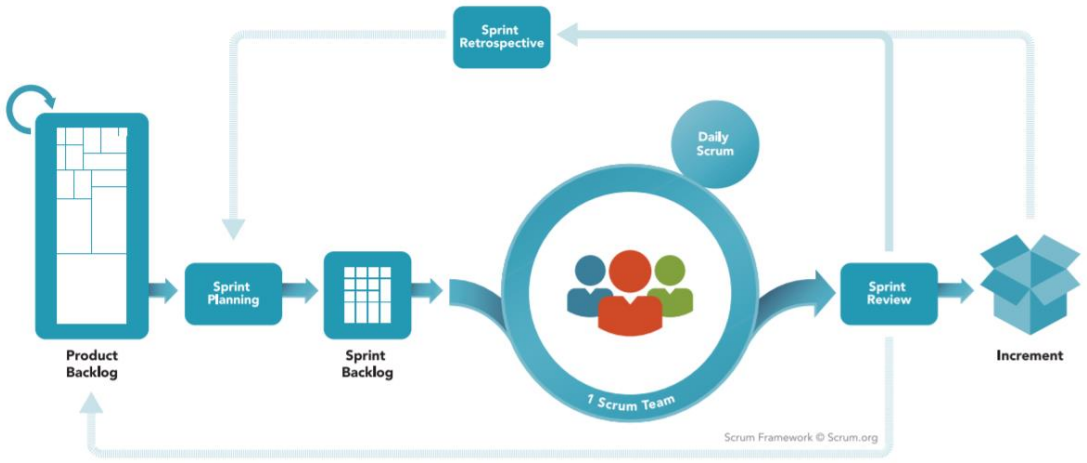
Jak uvádí jeden z tvůrců této metodiky Ken Schwaber – Scrum je přesný opak tradičních metod řízení projektu, které používají detailní plány, Ganttovy diagramy a pracovní rozpisy. Narozdíl od těchto nástrojů, které jsou prakticky v rozporu s přirozeným spádem projektu, Scrum ukazuje, jak řídit projekt podle jeho optimálního směru, který se odvíjí od průběhu projektu (Schwaber, 2004).

Jak již bylo řečeno pro projekt řízený Scrumem je definováno pouze několik základních pravidel. Jsou definovány role členů týmu, způsob práce takového týmu, události a další artefakty, které provedou tým projektem od začátku až do konce.

Pro obecný vhled a pochopení metodiky je potřeba uvést základní pravidla Scrum projektu:

- Jsou definovány tři role
  - Product Owner
  - Scrum Master
  - Team (členové vývojového týmu)
- Je definováno pět událostí:
  - Sprint
  - Daily Scrum Meeting
  - Sprint Planning Meeting
  - Sprint Review Meeting
  - Sprint Retrospctive Meeting
- Jsou definovány tři artefakty:
  - Product Backlog
  - Sprint Backlog
  - Product Increment

Výše zmíněné role, události a artefakty, jsou základy metodiky Scrum. Pokud se tým rozhodne řídit svůj projekt pomocí Scrumu, nemůže vynechat jedinou ze zmíněných věcí.



Obrázek 6 Metodika Scrum, průběh vývojového cyklu
Zdroj: www.scrum.org

Obrázek číslo 6 ilustruje, jak jednoduchý Scrum je. Pro přehlednost bude nyní popsáno několik základních principů metodiky Scrum a jak je zacházeno s definovanými rolemi, událostmi a artefakty. V dalších kapitolách se budeme těmto oblastem věnovat detailněji. Základní principy Scrumu jsou:

- Scrum tým se řídí sám
- Členové vývojového týmu nejsou specializováni na jednu konkrétní činnost a sami si volí na čem pracují podle aktuální potřeby
- Product owner ani Scrum master nejsou formálními nadřízenými členů vývojového týmu
- Požadované chování výsledného produktu je rozděleno na jednotlivé funkcionality, které jsou shromážděny v tzv. Product Backlogu
- Jednotlivé funkcionality jsou na planning meetingu rozděleny na tzv. „User Stories“, které jsou shromážděny v tzv. Sprint Backlogu
- Tým v rámci sprintu pracuje na jednotlivých User Stories a o svém postupu se členové informují na pravidelných denních Scrum meetinzích
- Hotové User Stories jsou na review meetingu označeny jako hotové a jsou shromážděny v product increment, nebo vráceny na planning meeting
- Jeden sprint je jedna iterace a jeho délka je nastavena podle povahy produktu
- Každý sprint je zakončen retrospektivou, na které se navrhují možná zlepšení organizace práce, nebo třeba technologických postupů

4.3.1 Role v týmu

Product owner

Product owner odpovídá za zastupování zájmů společnosti a všech, kteří jsou do projektu a jeho výsledku zapojeni. Product Owner má na starost počáteční a průběžné financování projektu vytvořením původních obecných požadavků projektu neboli vize projektu. Product owner je jediný ze Scrum týmu, kdo se podílí na finanční problematice projektu. Tvoří analýzu návratnosti investic (ROI) a plány dokončení produktu. Seznam požadavků na produkt se nazývá Product Backlog.

Product Owner zajišťuje, aby byl produkt budován podle Product Backlogu a zajišťuje, že nejcennější funkce jsou vyvinuty prioritně a dále je na nich stavěno. Toho je dosaženo častým re-prioritizováním Product Backlogu (Schwaber, 2004).

Product Owner má na starost (Myslín, 2016):

- Definice vize projektu
- Definice úkolů (tvorba Product Backlogu)
- Prioritizace úkolů
- Případné zrušení sprintu
- Komunikaci se zákazníkem

Scrum Master

Scrum Master odpovídá za to, že projekt následuje metodiku Scrum. Vzdělává členy týmu a všechny zapojené v týmu do pravidel Scrumu a je hlavním garantem správného nakládání se Scrumem. Má na starosti to, aby procesy definované v Scrum zapadly do kultury organizace a týmu, ale stále přinášely očekávané přínosy. V neposlední řadě zajišťuje, aby všichni následovali pravidla a postupy Scrumu, například dohlíží na to, aby se uskutečnily všechny naplánované meetingy a ty, aby měly konkrétní cíle programu (Schwaber, 2004).

Správný Scrum Master dokáže odfiltrovat veškeré nepodstatné věci, se kterými se může vývojový tým potýkat při projektu. Zůstává neutrálním hráčem a mezičlánkem ve vztahu vývojového týmu a Product Ownera a svými aktivitami přispívá, aby obě strany měly nejlepší možné projektové zázemí pro efektivní vývoj produktu. Jeho práce se skládá také z motivace týmu.

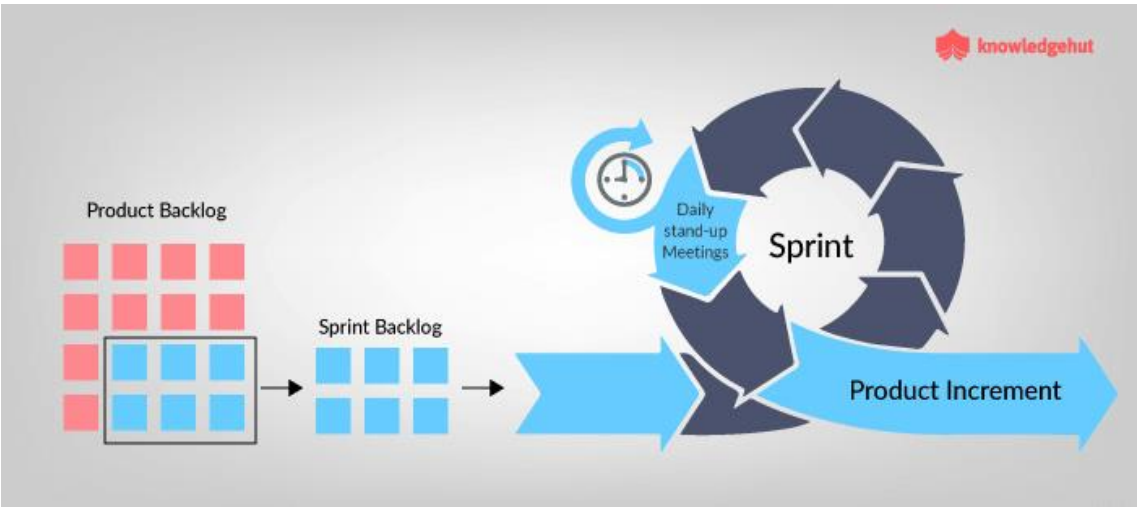
Scrum Master není obvykle nejzkušenější programátor, nebo obecně technicky nejlepší člen týmu (Šochová, Kunce, 2014). Mnohem víc než tvrdé technické dovednosti, jsou pro práci Scrum Mastera vyžadovány měkké dovednosti. Komunikace, naslouchaní, přirozená schopnost řešit problémy lidského charakteru a zároveň perfektní znalost metodiky Scrum.

Scrum Master také musí být skvělý kouč a umět tým povzbudit k překonání překážek. Potřebuje rozumět fázím formování týmů, poznat problémy v atmosféře a kultuře společnosti. To všechno jsou potřebné ingredience úspěšného Scrum Mastera (Šochová, 2018).

Tým

Tým je zodpovědný za vývoj funkcionalit. Týmy ve Scrum jsou samosprávné, samoorganizované a „cross-functional", tedy žádný z členů týmu by neměl být nezastupitelný. V realitě je samozřejmě každý člen týmu silnější v jiné oblasti z důvodu rozdílného kariérního vývoje. Někdo má větší zkušenost s programováním webových aplikací, někdo má větší zkušenost s automatizovaným testováním. Oproti tradičním metodám se ale ve Scrumu nedělí tradičně role a práce všech členů týmu by měla být zastupitelná dalšími členy týmu. Členové vývojového týmu jsou zodpovědní za přeměnu Produktového Backlogu v Sprint Increment, tedy budování toho, co bylo definováno na Planning meetingu. Členové týmu jsou společně odpovědní za úspěch každé iterace a celého projektu (Schwaber, 2004).

4.3.2 Artefakty Scrum projektu



Obrázek 7 Artefakty ve Scrum projektu
Zdroj: www.knowledgehut.com

Product Backlog

Požadavky na systém, nebo produkt, který je vyvíjen Scrum týmem, jsou shromážděny v Product Backlog. Je tvořen primárně Product Ownerem, který může spolupracovat s celou řadou dalších lidí, záleží na komplexnosti. Požadavky jsou do Backlogu zaznamenávány ve formě User Stories. Tento pojem, ač není základním artefaktem Scrumu, je velice důležitý. User Story pochází přímo od klienta, pro kterého je produkt vyvíjen. Definuje, co má systém dělat, doslovně jde o uživatelský příběh, jak uživatel pracuje se systémem a co očekává, že bude dělat, ne jak to bude dělat (Myslín, 2016). Jednotlivé User Stories se definují za použití otázek. Příklad podle Myslína:

> „*Abych dosáhl <užitku> jako <role>, chci <cíl>. Příklad: Abych mohl najít duplicitu, jako administrátor, chci vypsat na obrazovku seznam uživatelů.* “ (Myslín, 2016).

Product Backlog, tedy slouží jako seznam plánovaných funkcionalit, které má vyvíjený systém mít. Jak již bylo řečeno, je úkolem Product Ownera prioritizovat tyto User Stories, podle toho, které mají být vyvíjeny první.

Sprint Backlog

Jak naznačuje obrázek č.7 Sprint Backlog je podmnožina úkolů, které byly vybrány pro daný vývojový cyklus (Sprint). Zdrojem pro Sprint Backlog je Product Backlog. Na výběru úkolů pro Sprint a zařazení do Sprint Backlogu se podílí celý tým a probíhá výhradně na Planning meetingu. Sprint Backlog pro tým představuje množství práce, které má být dokončeno a zároveň pomáhá všem členům týmu vidět, jak si tým při plnění úkolů aktuálně vede (Schwaber, 2004).

Product Increment

Scrum vyžaduje, aby týmy každý Sprint zvyšovaly funkčnost produktu, tedy aby byl dokončen vývoj nějaké části produktu. Tento přírůstek (Increment) musí být možné vypustit jako hotovou část produktu (je tzv. „shippable“), protože Product Owner se může rozhodnout tuto novou funkcionalitu okamžitě implementovat do zákaznického řešení. To vyžaduje, že každý Increment je důkladně testovaný, jeho

kód je dobře strukturovaný a revidovaný a celkově je připravený na implementaci. Aby Increment mohl být na Review meetingu označen jako hotový, musí být tato nová uživatelská funkcionalita náležitě zdokumentována (Schwaber, 2004).

4.3.3 Události Scrum projektu

Sprint

Základní orientační časová jednotka Scrum projektu je Sprint. Jedná se o jednu iteraci vývojového cyklu. Všechny další zmíněné události jsou součástí každého Sprintu. Délka Sprintu se udává v týdnech a odvíjí se od produktu a celkového kontextu. Obecně se doporučuje délka Sprintu od jednoho do čtyř týdnů. Sprint by měl být tak dlouhý, aby měl tým dostatek času na dokončení nové funkcionality. Delší Sprinty nejsou doporučovány, protože se tím tým okrádá o pravidelnou zpětnou vazbu, jak od klienta, tak od členů týmu samotných (Šochová, Kunce, 2014).

Daily Scrum Meeting

Pro tuto schůzi, je používán také výraz „standup meeting“, tedy schůze ve stoje (Šochová, Kunce, 2014). Je to z toho důvodu, že Daily Scrum Meeting je rychlá schůze, kde se členové týmu informují o postupu a překážkách u svých úkolů, na kterých pracují. Daily Scrum Meeting by se z pravidla měl konat každý pracovní den ve stejný čas. Každý člen týmu mluví samostatně a informuje ostatní o svém postupu. Jeho řeč zpravidla odpovídá na tyto otázky:

- Co jsem udělal od posledního Daily Scrum Meetingu na tomto projektu
- Co budu dělat na tomto projektu do dalšího Daily Scrum Meetingu
- Co mi brání od plánované práce

Tato základní sada otázek by měla ostatní dostatečně informovat o progresu u jednotlivých kolegů a pomoci týmu se samostatně organizovat. Poslední bod, tedy co brání v postupu, je důležitý, aby se členové týmu vzájemně neblokovali nedokončenými nebo rozdělanými částmi úkolů (Schwaber, 2004).

Sprint Planning Meeting

Důležitá událost, která předchází každému sprintu. Jsou zde plánovány a hodnoceny jednotlivé User Story, které budou v následujícím sprintu vyvíjeny a implementovány. Hodnocení jednotlivých User Story probíhá za přítomnosti celého týmu a využívá se tzv. Planning Poker (Sutherland, 2014). Jde o aktivitu, kde jednotliví členové týmu přiřazují každému User Story počet bodů podle toho, jak si myslí, že je daný User Story náročný na implementaci. K hodnocení se používá Fibonacciho sekvence, tedy sekvence, kde každé číslo je součet dvou předchozích (Sutherland, 2014). User Story tak dostávají bodové hodnocení 1, 2, 3, 5, 8, 13, 21, 35. Vyšší čísla se z pravidla nepoužívají.

Podle Sutherlanda (Sutherland, 2014) je pro úspěšné naplánování nadcházejícího sprintu důležité plánovat konkrétní User Story, aby bylo co nejjednodušší na implementaci a komplexní funkcionality se skládaly z více User Stories.

Sprint Review Meeting

Tato událost je podle Schwabera (Schwaber, 2004) často opomíjena a pokud tým nestíhá dodat naplánované funkcionality je často i rušena na úkor vývoje. To je špatný přístup a Scrum Master by měl dohlédnout na to, že každý sprint je zakončen Sprint Review Meetingem. Slouží především k (‘Scrum.org: Home of Scrum‘, 2019):

- Revizi hotových a nedokončených User Story
- Hodnocení sprintu, co šlo dobře a kde je prostor pro zlepšení fungování celého týmu
- Hodnocení a definici dalších kroků, prioritizace položek Backlogu - tyto výstupy slouží k dalšímu Planning Meetingu

Sprint Retrospective Meeting

Jak je popsáno v oficiálním průvodci Scrum (‘Scrum.org: Home of Scrum‘, 2019), Sprint Retrospective Meeting je příležitostí pro tým Scrum, aby zkontroloval svoje aktivity a vytvořil plán pro zlepšení, která budou uzákoněna během příštího Sprintu.

Typická retrospektiva nastává po Sprint Review a před dalším plánováním sprintu. Scrum Master zajišťuje, že se událost koná a že tým rozumí jejím účelu. Primárně se jedná o příležitost pro tým, aby se zlepšil v různých oblastech a všichni členové by se měli účastnit.
Během retrospektivy Sprint tým diskutuje:

- Co se ve sprintu dařilo
- Co by se mohlo zlepšit
- Co chce zlepšit v příštím sprintu

4.3.4 Průběh projektu

Jádrem Scrum je iterace. Tým v rámci každé iterace na analyzuje požadavky, zvažuje dostupnou technologii a hodnotí své vlastní dovednosti a schopnosti. Poté kolektivně určuje, jak budovat funkčnost (tvoří User Stories), upravuje svůj přístup každý den (Daily Scrum Meeting), když naráží na nové složitosti, potíže a překvapení. Tým zjistí, co je třeba udělat, a vybere nejlepší způsob, jak toho dosáhnout. Tento kreativní proces je podle (Schwaber, 2004) srdcem a hlavním klíčem produktivity Scrumu.

Projekt vedený Scrumem začíná tvorbou Backlogu. Pokud jde o nový projekt, je důležité začít tvořit od začátku reálná, zpracovatelná User Stories, aby mohla být následně rozpracována do jednotlivých úkolů. Pokud jde o projekt, který již běží, jen bude nově veden Scrumem, větší důraz je potřeba u prioritizace User Stories v Backlogu (Pham A., Pham P., 2012). Dále je potřeba podle povahy projektu a dalších okolností nastavit vhodnou délku sprintu. Jakmile je připraven produktový Backlog a stanovena délka vývojového cyklu (sprint) Scrum tým může začít pracovat podle pravidel daných Scrumem. Průběh sprintu je k vidění na obr. číslo 6.

S prací na novém projektu se pojí spousta doprovodných aktivit, jako připravení vývojového prostředí, seznámení se s technologiemi atd. Některé zdroje (Šochová, Kunce, 2014) uvádějí pro tyto aktivity vytvořit tzv. nulový sprint, tedy období, ve kterém se tým připraví na práci a není v tomto období očekáván žádný výstup.

Podle Šochové a Kunce (Šochová, Kunce, 2014) je však z níže uvedených důvodů vhodnější, začít „naostro“ a definovat očekávané výstupy už v první iteraci, tedy prvním sprintu, ale snížit očekávaný inkrement. Pokud se například podle počtu a zkušeností týmu obecně předpokládá, že za jeden sprint budou průměrně dokončovat úkoly (User Stories) v hodnotě 30 bodů, doporučuje se dostat se na tuto hodnotu během prvních 3-4 sprintů. Myšlenka je nechat týmu čas na rozjezd a na zmíněné seznámení se s technologiemi a nastavení vývojového prostředí, ale očekávat výstupy, i když se sníženým očekáváním. Podle Šochové a Kunce se v praxi tým na svůj předpokládaný potenciál dostane rychleji a celý začátek projektu je tak efektivnější než s plánovaným nulovým sprintem. Postup projektu, a to jak se týmu daří dokončovat User Stories může Scrum Master měřit pomocí tzv. velocity diagramu (Nicolette, 2015).

4.4 XP – extrémní programování

Extrémní programování není ucelená metodika vývoje softwaru, ale spíše sada nástrojů, hodnot a best practises ulehčujících práci programátorům na komplexních projektech, kde špatným stylem práce a častými změnami v zadání vzniká velké procento neefektivity. Na rozdíl od praktik Scrumu je mnoho praktik XP specifických pro programování a jejich cílem je řešit nejčastější problémy, které způsobují týmům sestavení nekvalitního kódu. I proto vznikl mýtus toho, že XP je pouze pro ty nejlepší programátory, i když to není pravda. (Stellman, Greene, 2015). XP vyznává 13 hlavních praktik a prvním, kdo je zformuloval, byl Beck a tým jeho spolupracovníků (Beck, 2002).

V současné praxi se využívá XP nejčastěji v kombinaci s metodikou Scrum. Jak naznačuje výzkum (Version One, 2019) samotný XP je dnes spíše ojedinělá záležitost a pouze 1 % dotazovaných týmů používá pouze XP.

Extrémní programování používá objektově orientovaný přístup a zahrnuje soubor pravidel a praktik, které se vyskytují v rámci čtyř rámcových činností: plánování, návrh, kódování a testování. (Pressman, 2010)

4.5 Škálování agilních metodik

Adopce agilního vývoje softwaru je v dnešní době již rozšířené téma a nejedná se o žádnou novou praktiku. Není pravdou, že by se implementace agility týkala pouze IT vývojových týmů. Metody, které byly zmíněny v předchozích částech, byly designovány pro využití v malém měřítku, nebo dokonce pouze pro jednotlivé týmy. Velké společnosti adoptují metodiky, které jim pomáhají škálovat zavedené agilní procesy i mimo prostředí vývoje softwaru s cílem reprodukce pozitivních dopadů agilních metodik. Tzv. large-scale agile principy pomáhají firmám například i s řízením mezinárodních distribuovaných týmů, se kterým si vodopádové metodiky nedokážou tak dobře připravit a nejsou pro ně designované (Papadopolulos 2015), Tento proces adopce agilních metodik většími týmy, nebo celými společnostmi, je velice komplexní a setkává se s mnoha výzvami. Větší týmy a společnosti vyžadují v tomto směru větší míru koordinace.

Scrum je designován pro použití v týmu o velikosti 3-9 členů. Velké společnosti však často ve svém vývojovém oddělení zaměstnávají stovky programátorů, testerů, analytiků a jiných odborníků. Myšlenky a principy agility jsou zde však stejné. Každý tým se v rámci daných pravidel (např. Scrum) řídí sám, jeho členové jsou navzájem zastupitelní a znají produktový Backlog, tedy připravené úkoly, na kterých daný tým má pracovat. Tyto úkoly neboli User Stories, by měly být nezávislé na ostatních, a to i v případě, že na jednom produktu pracuje více týmů. Jednotlivé týmy se samozřejmě musejí mezi sebou domlouvat např. na technickém řešení. Fungovat by zde měla hlavně spolupráce mezi Product Owners a také Scrum Mastery na tzv. Scrum of Scrums meetingu (Šochová, Kunce, 2014).

Je samozřejmě velký rozdíl v uplatňování agilních principů ve společnosti o dvou nebo dvou stech Scrum týmech. Agilní týmy dnes ve velké míře řeší, jak rozšířit agilní metody v rámci celé společnosti (Stellman, Greene, 2015). I přesto, že teoreticky by mělo jít pouze o jakési přebrání agilních principů, v praxi s velikostí firmy roste i náročnost implementace agilních metodik. Touto problematikou se

dnes zabývá řada metodik. Většina z nich vychází právě z metodiky Scrum a bere ji jako základ pro své procesy. Některé metodiky vznikly uvnitř společností (Spotify), některé jsou vytvořené obecně, jako metodiky, které se specializují na large-scale agile. Podle (Version One, 2018) a (Alqudah, Razali, 2016) jsou v současnosti rozšířeny hlavně tyto metodiky:

- DAD – Disciplined Agile Delivery
- SAFe – Scaled Agile Framework
- LeSS – Large Scaled Scrum
- Spotify
- Nexus
- RAGE – Recipes for Agile Governance in the Enteprise
- A další.

Alqudah a Razali ve své práci (Alqudah, Razali, 2016) uvedli srovnání těchto nejznámějších large-scale agilních metodik. Tabulka číslo 1 přehledně porovnává tyto metodiky podle:

- velikosti a typu organizace, pro které jsou vhodné
- metod, ze kterých vycházejí
- požadavků na technické a možnosti, jak se v dané metodice vzdělávat

Tabulka 1 Srovnání large-scale metodik
Zdroj: Vlastní zpracování podle (Alqudah, Razali, 2016)

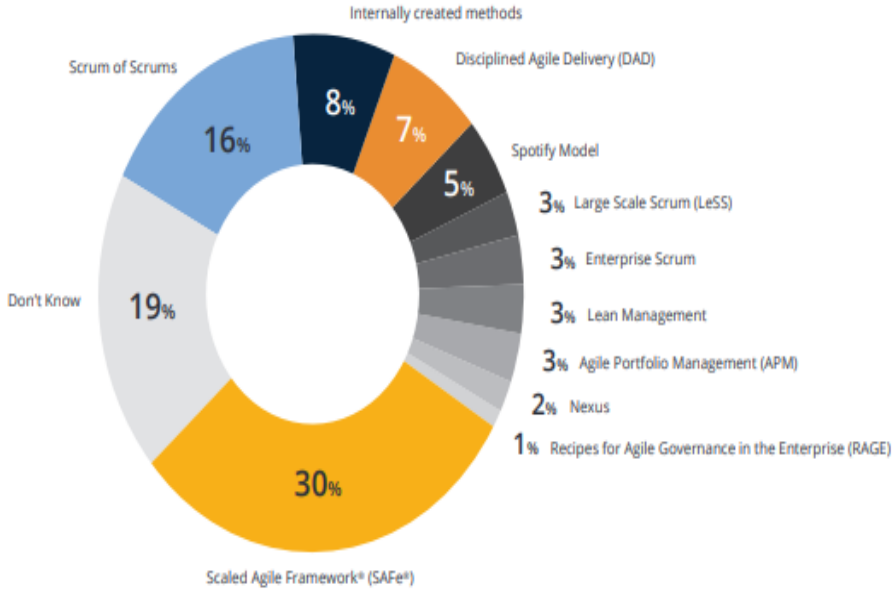
Velikost a typ organizace	
DAD	200 a více lidí, různorodé organizace a podniky
SAFe	50 – 125 lidí na jeden agilní vlak, střední až velké mezinárodní společnosti s portfoliem produktů
LeSS 1	Maximálně 10 agilních týmů, tradiční velké organizace

LeSS 2	Více než 1000 lidí, nadnárodní korporace
Spotify	250–300 lidí, přímo ve firmě Spotify je to 30 týmů, vhodné pro společnosti typu Spotify, tedy softwarové společnosti s jedním klíčovým digitálním produktem
Nexus	3–9 Scrum týmů, středně velké společnosti se začínajícím portfoliem produktů
RAGE	Velikost týmu není specifikována, vhodná pro tradiční i agilní korporace
Metodiky, ze kterých vychází	
DAD	Vybrané praktiky Kanban, SCRUM, Extreme Programming, Test Driven Development
SAFe	Scrum, Lean, Kanban, SCRUMban, DevOps a vybrané praktiky Extreme Programming
LeSS 1	Scrum
LeSS 2	Scrum
Spotify	Kanban, SCRUM, DevOps a Lean Startup
Nexus	Scrum a obecně praktiky, které řeší závislosti mezi produkty
RAGE	Scrum a Kanban, projektové plánování rozvoje a další hybridní metodiky
Technická náročnost a možnosti vzdělávání	
DAD	Technicky náročné (velké množství procesů a praktik). Oficiální kniha metodiky a certifikované workshopy.

SAFe	Středně těžká technická náročnost, náročnější na vyšších manažerských úrovních. Interaktivní web se všemi informacemi. Několik oficiálních i neoficiálních knih a příruček. Oficiální certifikované školení různých rozměrů. Certifikovaní instruktoři s mezinárodním působením.
LeSS 1	Méně náročné pro manažerské úrovně, nízké pro pracovní týmy. Možnost oficiálního školení v USA a několika dalších zemích. Neoficiální online materiály.
LeSS 2	Středně náročné pro manažerské úrovně, nízké pro pracovní týmy. Možnost oficiálního školení v USA a několika dalších zemích. Neoficiální online materiály.
Spotify	Středně náročné. Nedostatek informací a neexistence oficiálního tréninku pro ostatní společnosti.
Nexus	Středně náročné. Potřeba certifikovaných trenérů Scrum.
RAGE	Nízké až středně náročné. Tréninky formou online webinářů a oficiálních prezentací.

Na obr. číslo 8 je pak možné vidět jaké metody jsou nejčastěji používané v roce 2018 podle známého The State of Agile reportu (Version One, 2018). V tomto obrázku jsou uvedeny i obecné praktiky škálování jako je Scrum of Scrum. Ta nevychází z žádného specializovaného large-scale agile metodického rámce. Jedná se o nadstavbu

standartního Scrum, se kterou přišli tvůrci Scrum (Sutherland,2001). Jedná se o dodatek, který reaguje na současné trend škálování ještě předtím, než se trendem stal.



Obrázek 8 Large-scale agile metodiky a praktiky
Zdroj: (Version One, 2018)

4.5.1 Large-scale agile

Předchozí kapitola popisuje obecně škálování agility u větších projektů a týmů. Co je ale vlastně považováno za větší projekt, případně společnost? Podle Dikerta a kolektivu (Dikert, Paasivaara a Lassenius, 2016) není definice tzv. large-scale jednoznačná a velice se liší podle osoby, která je dotazovaná a podle kontextu. Obecně je však pro definování velikosti potřeba brát v úvahu počet osob pracujících na projektu, počet týmů, do kterých je projekt rozdělen, rozpočet projektu, počet řádků kódu, nebo délka trvání projektu. Na základě několika studií pak Dikert a kolektiv (Dikert, Paasivaara a Lassenius, 2016) definovali large-scale projekty tak, že musejí mít alespoň padesát pracovníků (nemusí se jednat pouze o programátory) v alespoň šesti týmech.

4.5.2 Výzvy se zaváděním large-scale agilních metodik

Implementovat agilní metodiky do malého vývojového týmu plného specialistů zaměřených na jednu oblast je jedna věc. Pokusit se o to samé v prostředí velké

společnosti je velmi odlišná výzva s mnoha bariérami. Jedná se o bariéry komunikační, byrokratické, nebo jen prostý problém s tím, jak dlouho a kolik peněz stojí proškolit stovky zaměstnanců na novou metodiku vývoje produktu. Zvlášť problémový je při zavádění agilních metodik na větší projekty způsob, jak zvládnout koordinaci mezi desítkami týmů, které navíc mohou pracovat na rozdílných produktech. Další výzvou agility ve velkých společnostech je zapojení rozdílných organizačních jednotek, jako jsou lidské zdroje, marketing, obchod atd. Avšak i přes nemalé množství výzev a problémů spojených se zaváděním agilních metodik a transformací velkých společností, existuje trend přijímaní agilních metodik ve velkém měřítku (Dikert, Paasivaara a Lassenius, 2016).

Jako hlavní překážky a výzvy při procesu implementace a transformace podniku na large-scale agilní metodiky byly identifikovány (Kalenda, Hyna a Rossi, 2018):

- Odpor ke změně
- Distribuované týmy
- Snížení kvality kódu
- Integrace s ne-agilními částmi společnosti
- Nedostatek odhodlání
- Zvýšení objemu práce a tlaku
- Nedostatek znalostí, koučování a tréninku k dané metodice

Podle analýzy, kterou provedl tým (Kalenda, Hyna a Rossi, 2018), jsou tyto překážky nejčastěji vyskytující se v odborné literatuře a případových studiích zabývajících se integrací largce-scale metodik. Body jsou seřazeny podle počtu výskytů od nejvyskytovanějšího po nejméně. Není překvapivé, že první místo obsadil odpor ke změně. Odhodlání vrcholného managementu a vytvoření agilního nastavení mysli je již několikrát zmíněnou výzvou úspěšného adoptování agility. Bohužel spousta společností je přijímá pouze proto, že jde o trendovou záležitost, nebo třeba z důvodu tlaku mateřské společnosti.

5 Scaled Agile Framework

Scaled Agile Framework, zkráceně SAFe, je jeden z nejrozšířenějších metodických rámců pro škálování agility. SAFe synchronizuje práci velkého množství agilních týmů pomocí různých nástrojů zaměřených na sjednocení cílů, spolupráci a společné doručení produktu. Firmám pomáhá vyvíjet menší softwarové produkty, ale také velké komplexní systémy, na kterých pracují tisíce lidí. SAFe byl vyvinut tak, aby pomáhal řešit nejnáročnější problémy spojené se škálováním ve firmách.

První verze metodického rámce Scaled Agile Framework, byla uvedena v roce 2011. Od té doby se tato metodika šířila velmi rychle a dnes ji využívá přes 70 % společností na seznamu Fortune 100 („Scaled Agile Framework – SAFe for Lean Enterprises“, 2019).

Zakladatelem a hlavním metodikem SAFe je Dean Leffingwell. Metodika SAFe navazuje na myšlenky a principy publikované v jeho knize Scaled Sotfware Agility: Best Practies for Large Enterprises (Leffingwell, 2007). Ucelená metodika byla pak vytvořena Leffingwellem a týmem odborníků, pracujících ve stejnojmenné americké společností Scaled Agile („Scaled Agile Framework – SAFe for Lean Enterprises“, c2019). Kromě původního týmu má metodika spoustu přispěvatelů z řad odborné veřejnosti, klientů a dalších odborníků z oblasti agilních metodik a projektového managementu. Proto se také samotná metodika v čase mění, je aktualizována a upravována. V době psaní této práce byla aktuální verze označena jako SAFe 4.6, byla aktualizována v listopadu 2018. Principy a informace k metodice jsou zdarma k nalezení na webové stránce scaledagileframework.com („Scaled Agile Framework – SAFe for Lean Enterprises“, c2019), která je zároveň jediným originálním a úplným zdrojem informací. Společnost nabízí placenou možnost certifikací a školení k zavádění SAFe ve společnostech všech druhů a velikostí.

SAFe si klade za cíl řešit problémy podniků a nalézat odpovědi na tento typ otázek:

- Jak sladit technické a obchodní cíle podniku?

- Jak dodat nové hodnoty (produkty) v předvídatelném rozvrhu tak, aby zbytek podniku mohl plánovat a provádět své aktivity?
- Jak může podnik škálovat agilní praktiky z týmu na větší programovou a obchodní jednotku a v rámci celého podniku, abych dosáhl lepších výsledků?
- Jak měřit, že nové způsoby práce jsou efektivnější?
- Jak vědět, co konkrétní agilní týmy dělají a jak měřit, jak dobře si vedou?
- Jak lze pomoci z pozice managementu týmům zlepšit se, aniž by se dostali do cesty jejich práce?

5.1 Východiska SAFe

Východiska pro SAFe, tedy oblasti, na jejichž základech tato metodika stojí, se dají rozdělit do tří oblastí. Jsou jimi:

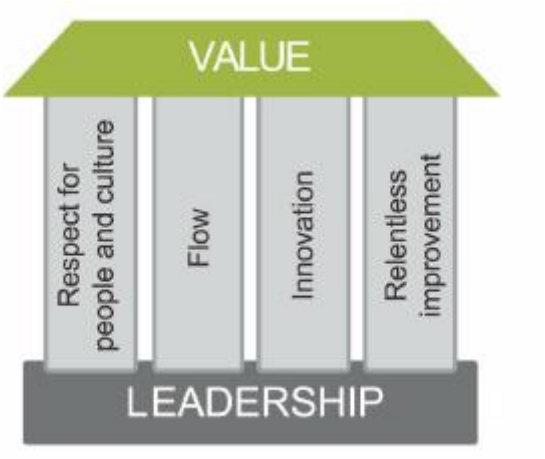
- agilní principy vývoje software
- systémové myšlení
- lean produktový vývoj

Postavena na těchto třech oblastech, SAFe je metodika, která byla navržena pro vývoj komplexních systémů štíhlým a agilním způsobem. Kombinuje sílu agile se štíhlým produktovým vývojem a systémovým myšlením. Synchronizuje a směřuje spolupráci několika agilních týmů na stejný cíl. Na základě několika desítek případových studií dosahují společnosti, které metodiku SAFe integrují, výrazná zlepšení. Výsledkem takové implementace je, že se dramaticky zlepšuje obchodní agilita urychlením produktivity, doby uvedení na trh, kvality produktu a zapojením zaměstnanců („Scaled Agile Framework – SAFe for Lean Enterprises“, c2019).

Obrázek 9 Výsledky implementace SAFe
Zdroj: („Scaled Agile Framework – SAFe for Lean Enterprises", c2019).

Společnost, která to s implementací SAFe myslí vážně, musí nejprve přijmout zmíněné Lean-Agile myšlení. Obě tyto oblasti – lean vývoj a agilní principy, byly popsány obecně v předchozích kapitolách. Metodika SAFe vychází z těchto oblastí a ty jsou v ní reprezentovány následovně:

- House of Lean (viz obrázek číslo 9)
- Agilní manifest (viz kapitola 4.1.)



Obrázek 10 House of Lean
Zdroj: (Knaster, 2017, s.)

5.2 Základní informace

Metodický rámec SAFe, tak jako třeba agilní metodika Scrum, popisuje role, jejich zodpovědnosti, artefakty, události, aktivity a další nezbytnosti. Oficiální web (https://www.scaledagileframework.com) nabízí detailní popis všech náležitostí, každá ikona na plánu SAFe se dá rozkliknout a nabízí vyčerpávající popis. Cílem této kapitoly, je přehledně charakterizovat nejdůležitější aspekty metodiky SAFe.

Grafické schéma metodiky SAFe je uvedeno jako příloha č.1 této práce.

5.3 Konfigurace SAFe

Jak bylo již zmíněno, metodika SAFe je vyvíjena tak, aby byla uplatnitelná na široké množství problémů se škálováním, a tedy i do prostředí různě velkých společností. Společnosti jsou řízeny různými způsoby s více či méně manažerskými úrovněmi, s menšími či většími týmy. Metodika SAFe je rozdělena do několika úrovní, které na sebe navazují a různé konfigurace těchto úrovní, jsou vhodné pro různé typy implementací. SAFe nabízí tyto konfigurace („Scaled Agile Framework – SAFe for Lean Enterprises“, c2019):

- Essential SAFe
    - Toto je nejzákladnější konfigurace a nejryzejší podoba metodiky SAFe. Definuje pouze nejzákladnější principy a potřebné minimum k tomu, aby metodika byla funkční a její implementace úspěšná.
- Large Solution SAFe
    - Tato modifikace je pro společnosti, které vytvářejí větší systémy a komplexní řešení, avšak nepotřebují nástroje, používané ve verzi Portfolio SAFe. Agile Release Trains jsou zde spojovány do tzv. Solution Trains.
- Portfolio SAFe
    - Stejně jako Large Solution SAFe tak i Portfolio SAFe je navrženo na větší projekty, na které je Essential SAFe již nedostačující řešení. Rozdíl oproti Large Solution je, že agilní vlaky nejsou spojovány do jednotlivých Solution Trains.

- Full SAFe
    - Full SAFe představuje nejrozsáhlejší konfiguraci tohoto metodického rámce. Podporuje vývoj rozsáhlých a složitých řešení, které typicky vyžadují práci stovek a více lidí ve vývoji i údržbě.

5.4 Úrovně SAFe

SAFe je metodický rámec, který je dělaný tak, aby pokryl celou organizaci. Celkem SAFe operuje na čtyřech úrovních, seřazeno od nejnižší.

- Team
- Program
- Large Solution
- Portfolio

Jednotlivým úrovním budou věnovány samostatné části. Pro obecné uvedení do metodiky SAFe je potřeba uvést, že týmová úroveň je nejčastěji spojována s používáním technik popsaných ve Scrum, nebo Kanban. Programová úroveň je zaměřena na tzv. Relase Train., který je složen ze sprintů jednotlivých týmů. Release Train je jedna z hlavních entit celé SAFe metodiky. Nejvyšší úroveň se nazývá Portfolio. Tato úroveň poskytuje procesy na základě lean principů, které pomáhají vedoucím pracovníkům a exekutivnímu managementu identifikovat a prioritizovat funkce vyvíjeného řešení. Ty lze poté rozdělit do programových úrovní a naplánovat Release Train, který se dále rozpadá na jednotlivé úkoly jednotlivých vývojových (nejčastěji Scrum) týmů.

Popis všech rolí, událostí, artefaktů a procesů metodiky SAFe na všech jejich úrovních by byl velice rozsáhlý a není předmětem této práce. Proto je v následujících kapitolách vždy uvedena přehledná tabulka se všemi rolemi, událostmi a artefakty dané úrovně. Následný text poté představuje procesy dané úrovně a kontext v rámci celé metodiky SAFe.

V příloze této práce je uveden grafický model celého metodického rámce SAFe.

5.4.1 Team level

Úroveň Team se věnuje práci jednotlivých vývojových týmů. Na této úrovni týmy pracují téměř stejně, jako standartní Scrum týmy. Některé týmy však mohou svou práci řídit pomocí metody Kanban. SAFe, stejně jako Scrum, definuje cross-funkční týmy. Ty společně spolupracují na doručení funkčních částí systému na konci každé iterace. Iterace (Sprint) v SAFe trvá zpravidla 2 týdny. Náplň práce jednotlivých iterací je určena Product Ownerem, který má na starost Team Backlog. Každá iterace začíná událostí Iteration Planning, tedy plánováním nadcházející iterace, kde se sami členové týmu rozhodnou, na jakých Story budou pracovat tak, aby je dokončili do konce iterace. Stejně jako ve Scrum, se členové týmu každý den scházejí na několika minutové schůzi, kde probírají svůj progres. Každá iterace je zakončena ukázkou hotových funkcionalit a schůzí, kde se tým ohlíží a hodnotí uběhlou iteraci (Iteration Retrospective). Toto vše je opět jako ve Scrum prováděno za dozoru Scrum Mastera, který dohlíží, aby vše probíhalo hladce a odstraňuje případné problémy.

Tabulka 2 SAFe Team level
Zdroj: ("Scaled Agile Framework – SAFe for Lean Enterprises", 2019)

Role	Události	Artefakty
Scrum Master / Agilní coach	Iteration Planning	Story
Product Owner	Iteration Review	Enabler stories
Vývojový tým	Iteration Execution	Iteration goals
	Iteration Retrospective	Team Backlog
	Backlog refinement	Team PI objective
	Innovation and Planning Iteration	

Obecně se dá říct, že Team level je základ pro všechny další úrovně, bez vývojových týmů by nebylo co řídit a nikdo by nepracoval na samotném vývoji řešení, ať už se jedná o software, hardware nebo jiný typ produktu. Týmy svou prací napájí agilní vlak, který je popsán v následující kapitole.

Týmová úroveň se výrazně neliší od samostatných metodik Scrum, Kanban a Extreme Programming. Základem všeho je inkrementální cyklický vývoj. SAFe navíc detailně definuje například proces testování hotových řešení a obecně velmi dbá na kvalitu vývoje. Týmy v SAFe aplikují tzv. Behavior-Driven-Development. Jedná se o způsob vývoje, kdy se veškerá Story, která jsou týmem vyvíjena, odvíjí od reálného chování uživatelů systému. Cílem je vytvářet kvalitní zadání pro vývojové týmy, které navíc implementují Test-Driven-Development, tedy vývoj řízený testy. V praxi to vypadá tak, že dříve, než je vyvíjena samotná funkcionalita, je definován test, který musí tato funkcionalita splnit a tím dosáhnout akceptačních kritérií. Až potom může být nová funkcionalita vypuštěna.

Kromě základních nástrojů a funkcí Scrum, vývojové týmy v SAFe používají také vybrané nástroje používané v Extreme programming.

5.4.2 Program level

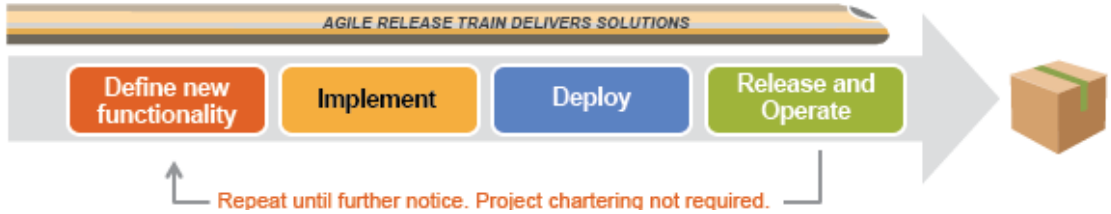
Úroveň Program, je jádrem metodiky SAFe. Je nadstavbou úrovně Team, která je velmi podobná standartnímu Scrumu. Dalo by se říct, že jde o rozšířený Scrum of Scrum. Na programové úrovni SAFe definuje tým složený z více týmů pracujících na doručení řešení (systému, nebo části systému) společně. Pro tento tým se v SAFe používá označení Agile Release Train (ART). Ten se skládá z 5 až 12 týmů, a to odpovídá počtu 50 až 125 lidí.

Tabulka 3 SAFe Program level
Zdroj: ("Scaled Agile Framework – SAFe for Lean Enterprises", 2019)

Role	Události	Artefakty
Product Management	PI Planning	Features
System Architect/Engineer	System Demo	Program Epics
Release Train Engineer	Inspect and Adapt	Program Backlog
Business Owners		Program Kanban
		PI Objective
		Architectural Runway

Programová úroveň obsahuje role a činnosti potřebné k nepřetržitému doručování zákaznických řešení prostřednictvím Agile Release Train.

Agile Release Train je tým agilních týmů (Scrum nebo Kanban) s dlouhou životností, který spolu s dalšími zúčastněnými stranami postupně vyvíjí, dodává a případně provozuje jedno, nebo i více řešení. Tyto dodávky jsou orientovány podle PI (Program Increment), který trvá pět vývojových iterací (pět sprintů).



Obrázek 11 Agile Release Train
Zdroj: ("Scaled Agile Framework – SAFe for Lean Enterprises", 2019)

Agile Release Train lze přeložit jako agilní vlak, který vypouští jednotlivé verze systému(releasy) a metafora spočívá v tom, že všechny vyvíjené funkcionality systému nemusí být v jednom vlaku (v jednom „releasu"), ale ke klientovi se mohou dostat až v těch dalších (v dalších Program Incrementech).

Obsah každého Program Incrementu, tedy funkcionalit, které v následujícím období budou vyvíjeny, je definován Product Owners na tzv. PI Planning Meetingu. Tyto funkcionality mají v SAFe označení Business Feature a jsou shromažďovány v Program Backlog, z těch následně přerozdělovány do jednotlivých agilních týmů, které své úkoly shromažďují v Team Backlog.
Každý agilní vlak je řízen Release Train Engineerem (RTE). Jeho role se dá přirovnat k roli Scrum Mastera jednotlivého Scrum týmu, s tím rozdílem, že tady je řízen celý tým týmů. Hlavní zodpovědností RTE je usnadňovat události a procesy ART a pomáhat týmům při dosahování cílů. RTE komunikuje se všemi zúčastněnými stranami, odbourává překážky, pomáhá řídit rizika a vyvíjejí neustálé zlepšování.

5.4.3 Large Solution level

Tato úroveň zahrnuje silnější zaměření na požadavky celkového záměru řešení a koordinaci více agilních vlaků, které k tomuto řešení směřují. Large Solution úroveň obsahuje další role, artefakty a procesy potřebné k vytváření opravdu velkých a komplexních řešení.

Tabulka 4 SAFe Large Solution level
Zdroj: ("Scaled Agile Framework – SAFe for Lean Enterprises", 2019)

Role	Události	Artefakty
Customer	Pre- and Post-PI Planning	Capabilities
Solution Architect/Engineer	Solution Demo	Solution Epics
Solution Management	Inspect and Adapt	Nonfunctional Requirements
Solution Train Engineer		Solution Backlog
Supplier		Solution Kanban

Hlavním pojmem této úrovně, podle které se vše točí, je tzv. Solution Train. Ten je považován za organizační konstrukt používaný k vytváření velkých a komplexních řešení, která vyžadují koordinaci více vlaků Agile Release Train (ART), jakož i příspěvky dodavatelů (externí nebo interní přispěvatelé projektu).

Obrázek 12 Solution Train složený z několika ART
Zdroj: ("Scaled Agile Framework – SAFe for Lean Enterprises", 2019)

Solution Train je určen k vývoji komplexních řešení označovaných jako systém systémů. Ty mohou vyžadovat spolupráci stovek až tisíců spolupracovníků.

Příkladem takových řešení mohou být zdravotní systémy, automobily, letadla a letecké systémy atd.

Solution Train sjednocuje obchodní i technologické cíle jednotlivých vlaků Agile Release Train. Solution Train je opět řízen ekvivalentem Scrum Mastera v tomto případě Solution Train Engineerem. Solution management se stará o tzv Capability, nadstavba Features (viz 5.4.6).

5.4.4 Portfolio level

Na této úrovni jsou definovány strategie a financování jednotlivých řešení. Tato úroveň slouží ke sjednocení celopodnikové strategie a jejího reálného provádění na nižších úrovních. Je zde zajišťována tvorba rozpočtů pro jednotlivá řešení a jejich Solution Train tak, aby plnily strategické cíle.

Úroveň Portfolio může v malých a středních podnicích spravovat veškerá technická řešení společnosti. Ve větších společnostech je pak tato úroveň myšlena pro řízení jedné oblasti produktů.

Tabulka 5 SAFe Portfolio level
Zdroj: ("Scaled Agile Framework – SAFe for Lean Enterprises", 2019)

Role	Události	Artefakty
Lean Portfolio Management		Business Epics
Epic Owners		Enabler epics
Enterprise Architect		Strategic themes
		Portfolio Backlog

5.4.5 Role jednotlivých úrovní

V tabulce číslo 6 jsou uvedeny všechny role tak, jak jsou definovány v metodice SAFe. Role jsou rozděleny ve sloupcích podle úrovně, ve které mají hlavní pole působnosti.

Tabulka 6 Přehled rolí v SAFe
Zdroj: ("Scaled Agile Framework – SAFe for Lean Enterprises", 2019)

Team	Program	Large Solution	Portfolio
Scrum Master/ Agile Coach	Release train engineer	Solution train engineer	Epic Owners
Product owner	Product management	Solution management	Lean Portfolio management
Vývojový tým	System architect	Solution architect	Enterprise architect
	Business owners	Zákazník	

5.4.6 Model požadavků v SAFe

SAFe ve své plné čtyř úrovňové podobě definuje také čtyři vrstvy artefaktů, které definují, jaké jsou požadované funkce vyvíjeného řešení. Tyto čtyři artefakty, které jsou na obr. číslo 14, tvoří spolu s tzv. nefunkčními požadavky Backlog jednotlivých úrovní. Díky jejich rozpadu na menší podmnožiny je možná nepřímá komunikace mezi klientem a technickými specialisty ve vývojových týmech. Nejvyšší vrstva, která se pojí k Portfolio úrovni je Epic. Capability je spojena s definováním řešení pro Solution Trains na úrovni Large Solution. Na nižší programové úrovni používají Agile Release Train k definování své práce a rozdělení funkcionalit systému tzv. Features. A nejnižší vrstvou pracovních položek je Story. Tato hierarchie umožňuje definovat jednotlivé úkoly pro vývojový tým tak, aby se daly dokončit během jedné iterace vývojového cyklu a zároveň byly orientovány na stejný cíl.

Obrázek 13 Work Items v SAFe
Zdroj: Vlastní tvorba dle ("Scaled Agile Framework – SAFe for Lean Enterprises", 2019)

Epic – jedná se o nejabstraktnější vyjádření zamýšlené funkcionality systému. Epic je možno rozložit do menších celků (uvedené níže). Lze si jej představit jako kontejner, který obsahuje vše, co je třeba pro zamýšlenou iniciativu. K implementaci

dochází ve více programových přírůstcích (PI) a každý Epic následuje cyklus Lean startup „build-measure-learn“.

Capability – je chování daného řešení vyšší úrovně, které zahrnuje více než jeden Agile Release Train (ART). Capability se dělí na Feature aby se usnadnila jejich implementace a vývoj a šli dimenzovat do období jednoho Program Increment.

Feature – je položka, která splňuje potřeby zúčastněných stran. Každá Feature obsahuje dvě složky. Akceptační kritérium a hypotézu výhod, tedy to, co její implementace přinese za funkcionalitu systému. Feature je rozdělena a dimenzována pro jednotlivé agilní vlaky Agile Release Train (ART) v každém Program Increment období.

User Story – jsou primárním artefaktem používaným k definování chování systému napříč všemi agilními metodikami. Jde o krátké jednoduché popisy funkcí, ne požadavky. Každý Story má umožnit implementaci malého výřezu funkcionality systému. Příběhy jsou koncipovány v jazyku klienta tak, aby techničtí specialisté z vývojového týmu porozuměli záměru klienta. Story jsou hodnocena body podle toho, kolik času jejich implementace zabere. Zdrojem pro User Story jsou Feature.

Epic, Capability a Feature jsou používány na popis zamýšleného chování větších funkčních celků v rámci celých řešení. Ty jsou pak rozpadnuty až na dílčí jednotlivá User Story, která popisují detailní záměry implementace a jsou shromážděna v Team Backlog.

5.5 Principy SAFe

Scaled Agile Framework je založen na principech štíhlé agility. Tyto principy představují základní myšlenky a ekonomické hodnoty, které udávají efektivitu rolí a praktiky. Tyto principy mohou být použity univerzálně, bez ohledu na to, jaká je situace ve firmě, která chce metodiku SAFe použít. Tvůrci metody SAFe jsou si vědomi toho, že praxe, která funguje v jedné situaci, nemusí nutně platit nebo fungovat v jiné. Lean-Agilní transformace může přinést každému podniku značné

výhody. Je to však významná změna a každá implementace je poněkud odlišná. SAFe je však dostatečně škálovatelný a konfigurovatelný rámec, který umožňuje každé organizaci aplikovat jej na svůj vlastní obchodní model. Proto dříve, než podnik může účinně uplatňovat postupy SAFe, je nutné porozumět základním principům. Tato kapitola popisuje devět principů SAFe Lean-Agile. SAFe je založen na devíti principech štíhlé agility (Leffingwell, 2007):

- Je třeba uplatňovat ekonomický pohled.
- Aplikujte systémové myšlení.
- Předpokládejte variabilitu; zachovejte možnosti.
- Vyvíjejte postupně pomocí rychlých integrovaných výukových cyklů.
- Zaveďte základní milníky pro objektivní hodnocení systémů.
- Vizualizujte a omezte WIP, zmenšete velikosti šarží a spravujte délky front.
- Použijte kadenci; synchronizovat s plánováním napříč doménami.
- Odemkněte vnitřní motivaci pracovníků.
- Decentralizujte rozhodování.

6 Praktická část

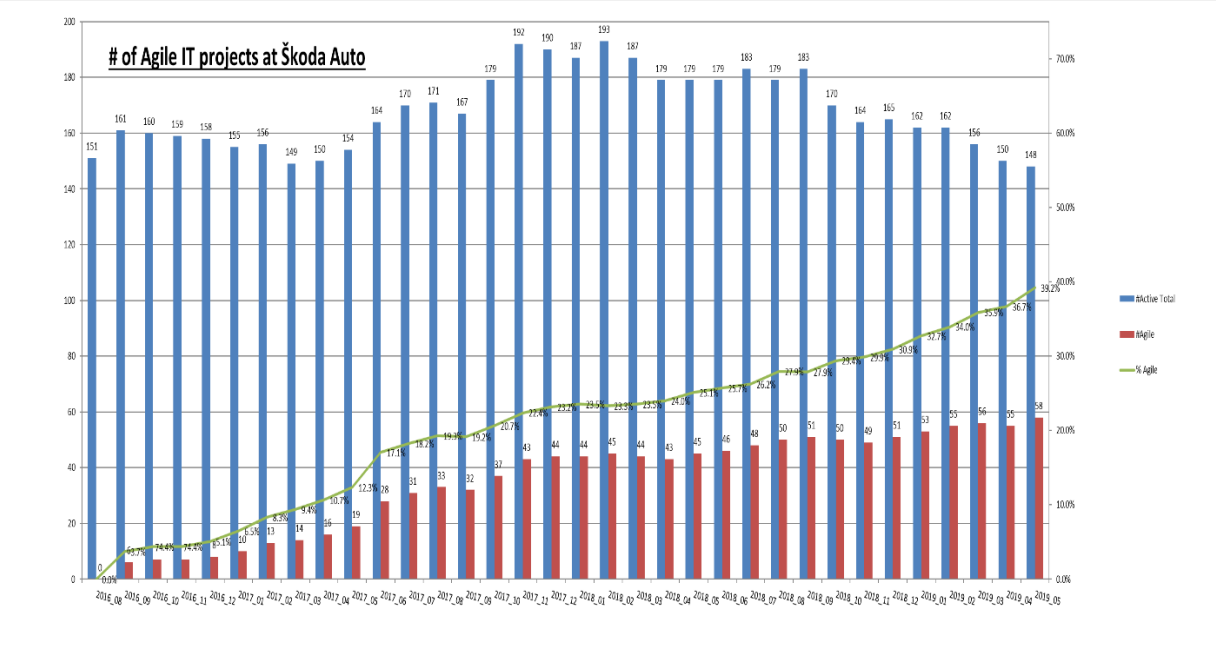
Praktická část práce se zabývá integrací popsané metodiky SAFe ve společnosti ŠKODA AUTO a.s. (dále jen ŠKODA AUTO). V této části je uveden popis prostředí a seznámení s kontextem (6.1), analýza situace ve ŠKODA AUTO (6.2), analýza rozdílů mezi metodikou SAFe a vnitropodnikovým metodickým rámcem IT-PEP (6.3) Dále jsou na základě objektivní analýzy definovány závěry a sady doporučení pro optimalizaci adaptace metodiky SAFe v této organizaci (6.4). Praktickou část práce doplňuje případová studie a analýza probíhající implementace škálovatelné agility za použití prvků metodiky SAFe do vybraného vývojového oddělení ŠKODA AUTO (7).

6.1 Představení společnosti ŠKODA AUTO a.s.

Historie společnosti sahá až do roku 1895, kdy ještě pod názvem Laurin&Klement, byla založena. Od roku 1991 je ŠKODA AUTO součástí německého koncernu Volkswagen. V dnešní době má ŠKODA AUTO jen v České republice bezmála třicet čtyři tisíc zaměstnanců. Aby si společnost ŠKODA AUTO udržela své místo na trhu, prochází v dnešní době digitální transformací. Z klasické výrobny automobilů se stává digitální mezinárodní společnost využívající moderní technologie jak ke svému řízení, tak ve svých produktech. K tomu jí slouží ŠKODA Strategie 2025. Jedná se o dlouhodobý, pružně se přizpůsobující koncepční plán, který je realizovaný konkrétními strategickými projekty. Společnost bedlivě sleduje vývoj v automobilovém odvětví a ve světě, promýšlí budoucí dopady nástupu umělé inteligence a autonomního řízení na populaci a své strategické plány adekvátně přizpůsobuje dynamickému vývoji. V průběhu roku 2018 došlo například k úpravě strategie mobility a konektivity, interní digitalizace, či k dalšímu rozpracování tématu udržitelného rozvoje, společenské odpovědnosti a integrity.

Právě oblasti mobility a konektivity, digitálních služeb a interní digitalizace jsou úzce spojeny s tématem agilních projektů a agilním vývojem komplexních digitálních produktů. O plošné zavádění agilních metodik vývoje se ŠKODA AUTO snaží u svých projektů již od roku 2016. K 05/2019 se celkový počet aktivních IT

projektů ve ŠKODA AUTO pohyboval kolem sto padesáti. Téměř 40 % projektů využívalo ke svému řízení agilní metodiky. Na obr. číslo 15 je vidět poměrně rychlý nárůst podílu agilních projektů ve ŠKODA AUTO.



Obrázek 14 Nárůst IT projektů vedených agilní metodikou v ŠKODA AUTO
Zdroj: ŠKODA AUTO a.s.

Zákazníkem jsou firmy a obyvatelé zemí, kam ŠKODA AUTO vyváží své automobily. V kontextu IT projektů jsou však zákazníky těchto projektů jednotlivé business jednotky uvnitř společnosti, jako například technický vývoj, digitální marketing, výroba, HR atd. Tito koncoví zákazníci analyzují a popisují požadavky (User Stories), které pak vývojové týmy realizují. Zde se míchají různorodé profese IT ŠKODA AUTO. Interní vývojáři a analytici, externí vývojáři nebo konzultanti apod. Paradoxně tedy koncový zákazník, který kupuje vůz může do procesu vstoupit pouze během nějakého průzkumu. Priority tedy vznikají na vysoké úrovni strategického managementu.

6.2 Analyzování situace

Na základě dostupných informací je v této části zpracována SWOT analýza přijetí metodiky SAFe do prostředí ŠKODA AUTO. Analýza se opírá o základní teoretická

východiska metodiky SAFe a dostupné informace o společnosti. Analýza vychází taktéž z odborných konzultací, které poskytli vybraní zaměstnanci, v čele s Ing. Davidem Židem a experti pro danou oblast ve ŠKODA AUTO. Zdrojem pro tuto analýzu jsou i odborné podklady uvedené v teoretické části práce.

V současnosti jsou projekty a produktový vývoj ve ŠKODA AUTO řízeny dle tzv. IT-PEP 2.0 ('IT-PEP 2.0: Metodika řízení produktů a projektů, 2019). Jedná se o metodický rámec pro vedení projektů a produktového vývoje a je mu věnována samostatná kapitola v následujících částech práce (6.4). Tato metodika odkazuje a do jisté míry integruje vybrané části metodik Scrum, Less a SAFe. Právě integrace metodiky SAFe ve ŠKODA AUTO je předmětem této práce, a to z toho důvodu, že její přijetí a používání je nyní ve ŠKODA AUTO aktuální výzvou. Byly definovány dvě oblasti možného využití nové integrace frameworku SAFe do stávajícího IT-PEP 2.0 a týmy, kterých se tato problematika napřímo týká.

1. Týmy společnosti ŠKODA AUTO, které se podílejí na koncernových projektech. To znamená na projektech, které jsou vedeny ve společnosti VW a jednotlivé vývojové týmy ŠKODA AUTO jsou přímo součástí těchto projektů, nebo produktového vývoje. Tyto projekty jsou řízeny metodikou SAFe. Ta je v metodice IT-PEP zmíněna, ale slučuje se pouze na úrovni vývojového týmu a delivery managementu.
2. Vodopádově vedené týmy, které chtějí metodiku SAFe přijmout jako hlavní metodiku pro řízení dodávek produktů nebo produktových portfolií.

6.2.1 SWOT analýza

V následující části je zpracována klasická SWOT analýza přijetí metodiky SAFe a její integrace do stávající metodiky IT-PEP 2.0 ve ŠKODA AUTO.

Tabulka 7 SWOT analýza
Zdroj: Vlastní tvorba

Silné stránky
SAFe je metodika, která je tvořená pro velké společnosti s několika úrovněmi managementu.
Metodika není zaměřena primárně na vývoj softwaru v čistě softwarových společnostech, ale nabízí nástroje a principy pro vývoj komplexních softwarových produktů.
SAFe je již implementována na vybraných projektech VW a vzniká tak tlak na adaptaci metodiky SAFe i ve ŠKODA AUTO.
Slabé stránky
Metodika SAFe není zcela kompatibilní s vnitropodnikovým metodickým rámcem IT-PEP 2.0 (viz 4.6).
Vyšší náklady na školení SAFe, organizaci a přípravu událostí SAFe a certifikované externí konzultanty.
Stále převládající kultura výrobní společnosti, odlišný mindset vedoucích pracovníků, zažitý vodopádový přístup.
SAFe přináší očekávanou hodnotu, když je implementován do velké části organizace, a to může trvat ve velké společnosti typu ŠKODA AUTO dlouhou dobu a nemusí také vůbec nastat.
Příležitosti
Do budoucna lze škálovat na větší interní celky firmy.
SAFe nabízí nástroje a postupy pro tvorbu dnes neexistujícího uceleného metodického postupu při zavádění úrovně portfolio ve ŠKODA AUTO.
Hrozby
Pokud bude metodiku používat pouze zlomek týmů a projektů, může přinést pouze zvýšení byrokracie a režijních nákladů tzv. overhead costs.
Zhoršení konkurenceschopnosti v důsledku zavádění nové metodiky a snížené efektivity práce.
Krátkodobé až střednědobé prodloužení dodávek.
Nízká adaptovatelnost vývojových týmů ŠKODA AUTO na novou metodiku SAFe.

6.3 IT-PEP 2.0

Společnost ŠKODA AUTO využívá pro řízení vývoje produktů a vedení projektů celkem tři vlastní metodiky.

- PEP (zkratka pro Proces Vývoje Produktu)
- IT-PEP ('IT-PEP 2.0: Metodika řízení produktů a projektů', 2019)
- Digital handbook

PEP neboli Proces Vývoje Produktu popisuje proces vývoje vozu, milníky a jedná se čistě o vodopádový styl řízení typický pro automobilový průmysl. IT-PEP se poté týká pouze projektů a produktů, které se nějakým způsobem vážou na oblast informačních technologií. Zahrnuje jak vodopádový, tak agilní přístup. Například součástí realizace projektu je dodávka nového, nebo stávajícího IT řešení. Může se jednat o integrace různých IT řešení a digitalizaci obecně (cloud, robotizace, automatizace procesů, konektivita, online služby, digitální marketing atd.). Pokud se projekt žádným způsobem nedotýká IT, nebo se týká výhradně samostatného vývoje vozu, nebo jeho části, pak pro něj IT-PEP 2.0 není platná metodika řízení. Digital handbook se zabývá metodickým popisem vývoje digitálních projektů a vymezuje, kam jednotlivé projekty spadají, zdali do PEP, IT PEP nebo digitálních projektů.

IT-PEP ve své stávající verzi 2.0 je upravenou metodikou pro klasické a agilní řízení na základě zkušeností a zpětných vazeb realizace projektů za dobu platnosti původního metodického pokynu. Jedná se o komplexní metodickou příručku, která upravuje, detailně definuje a rozděluje pravomoci a odpovědnosti mezi jednotlivé role. Jádrem celé metodiky je životní cyklus produktu (obrázek 16). Metodika také odkazuje na běžně známé metodické standarty, ať už pro tradiční, nebo agilní řízení projektů. Nesporně velkou výhodou IT-PEP je to, jak upravuje některé procesy, aby lépe vyhovovaly realitě životních cyklů projektů a produktů ve ŠKODA AUTO.

Předmětem veškerého snažení vývojových týmů je dodávka IT produktů, služeb nebo řešení. IT-PEP 2.0 přináší soubor nástrojů a metod pro řízení dodávek a snižování rizik s tím spojených. Dodávky jsou rozlišeny z pohledu velikosti na změny

(changes, do 150 člověkodní) a projekty (nad 150 člověkodní). Řízení dodávek je dále rozlišeno na:

1. Projektové řízení (vodopádové nebo agilní) – dodává se projekt v přesně definovaných časových intervalech se začátkem a koncem.
2. Agilní produktový vývoj – dodává se produkt a ten je vyvíjen inkrementálním způsobem typickým v agilních metodikách, kdy se v čase zvyšuje jeho hodnota a často jsou měněny požadavky.

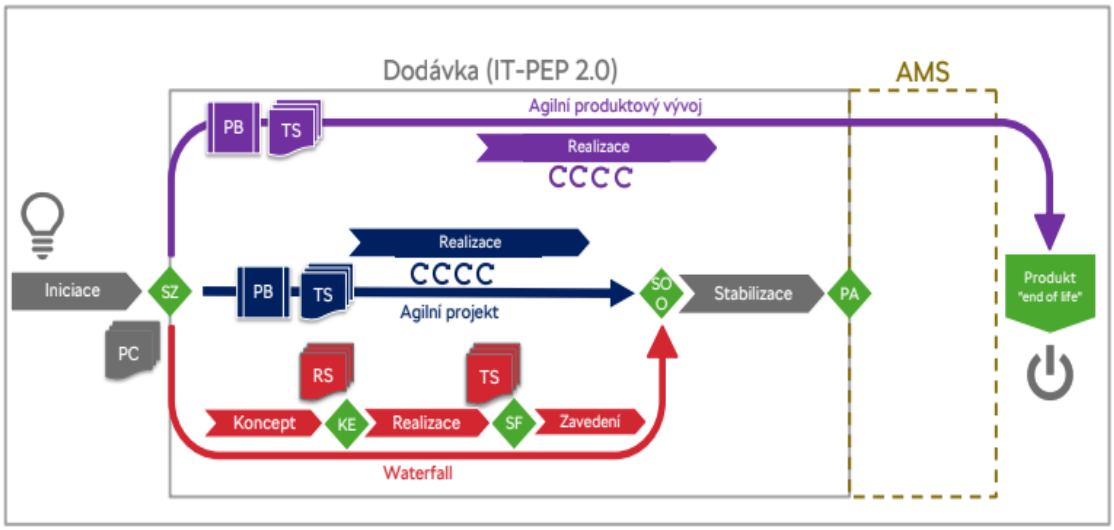
Obecný projekt je charakteristický tím, že má začátek, konec a jasně definované určité měřitelné parametry: harmonogram, rozpočet, kvalitu a výstup. Hlavním kritériem pro úspěšné dodání projektu je dodržení tzv. projektového trojimperativu: rozpočet, čas a rozsah. Tedy úplná realizace celkového rozsahu dodávky, ve stanoveném čase a při dodržení schváleného rozpočtu.

Naproti tomu při agilním produktovém vývoji je pro potřeby tohoto metodického rámce myšlen výsledek procesu vývoje, integrace a obecně práce vývojových týmů. Agilní produktový vývoj je pak adaptivní proces pokrývající celý životní cyklus vzniku, rozvoje a údržby komplexních produktů od iniciace až po ukončení provozu. Zaměřuje se na přínosy pro zákazníka a iterativní zvyšování užitné hodnoty produktu, spíše než na dodržení předem stanoveného plánu dodávky. Jde o kontinuální aktivitu rozvoje produktu udržitelným tempem. Životní cyklus produktu končí společně s jeho "posledním uživatelem", tedy v momentě, kdy produkt již není nadále aktivně využíván, nebo je nahrazen novým produktem.

Na obr. číslo 16 jsou graficky znázorněné tři možnosti doručení dodávky, podle IT PEP 2.0. Pro všechny tři metody je společný začátek. Tím je milník schválení projektu a konec fáze inicializace (SZ). V průběhu iniciační fáze proběhne volba modelu a typu dodávky na základě:

- tržního prostředí
- míry zapojení zákazníka
- inovativnosti řešení

- modularity práce
- dopadu chyb, které se objeví během vývoje



Obrázek 15 Životní cyklus dodávky podle IT-PEP 2.0[1]
Zdroj: ŠKODA AUTO a.s.

Z obrázku číslo 16 je tedy patrné, že pouze dvě ze tří metod dodávky podle IT-PEP 2.0 jsou spojeny s agilními principy. V následujícím textu tak budou uvažovány pouze tyto metody dodávky. Vodopádová metoda byla uvedena pouze pro obecné uvedení do problematiky vnitropodnikové metodiky. Jak pro agilní projekt, tak agilní produktový vývoj jsou v metodickém rámci ŠKODA AUTO IT-PEP 2.0 definovány stejné role a procesy. Agilní produktový vývoj má stejnou charakteristiku jako realizační fáze agilního projektu s tím rozdílem, že nemá dopředu plánovaný milník PA (milník schválení fáze stabilizace projektu) a jde o kontinuální aktivitu rozvoje produktu udržitelným tempem.

[1] PC – Project Charter, SZ – Milník Schválení projektu (Inicializace), PB – Product Backlog, TS - Technická specifikace, SOO – „Start of operation“, milník schválení fáze zavedení projektu, PA – Milník schválení fáze stabilizace projektu, AMS – „Application Managament Support“, RS – Requiremnt Specification, KE – Milník schválení fáze Koncept, SF – Milník schválení fáze Realizace

IT-PEP 2.0 se také velmi lehce dotýká škálování produktového vývoje a jsou pro tuto oblast vybrány dva metodické rámce. Pokud se jedná o lokální projekty bez návaznosti na koordinaci agilních programů napříč koncernem VW je doporučen metodický rámec LeSS (Large Scale Scrum). Pokud se naopak jedná o mezinárodní projekty, do kterých jsou zapojeny týmy z různých oblastí napříč koncernem VW je podle IT-PEP 2.0 doporučena metodika SAFe (Scaled Agile Framework).

Důležité je zmínit, že podle IT-PEP 2.0 se škálování agility netýká metody Agilní projekt, protože zavedení LeSS nebo SAFe jako organizačního a procesního rámce je časové náročnější a benefity se dostaví až po určité době a pro časově omezené dodávky tedy nedává smysl je zavádět.

6.4 GAP analýza

V této části je dopodrobna rozebrána vnitropodniková analýza rozdílů IT-PEP 2.0 a metodiky SAFe v konfiguraci Portfolio. Tato kapitola se netýká celého metodického rámce IT-PEP 2.0, ale pouze částí, které se věnují vedení Agilního projektu a Agilního produktového vývoje.

Porovnávání obou metodik a definování rozdílů je rozděleno do tří oblastí a těmi jsou: obecné rozdíly obou metodik, rozdíly v definovaných rolích a události, které provázejí vývoj obou metodik.

Pro přehlednost budou rozdíly zaznamenány formou jednoduché tabulky o stejném vzhledu a rozložení polí a budou seřazeny v jednotném pořadí napříč zkoumanými oblastmi. Jednotlivé rozdíly budou označeny jako GAP 1, GAP 2 a tak dále podle stejného vzoru.

Tato část práce navazuje, rozšiřuje a upravuje některé části již hotové GAP analýzy, kterou si společnost ŠKODA AUTO nechala vypracovat od nezávislého externího konzultanta.

6.4.1 Obecné rozdíly IT-PEP a SAFe

GAP 1	
Rozsah a účel metodik je rozdílný	
SAFe	IT-PEP 2.0 (Agile)
Jedná se o ucelenou metodiku, se kterou je možné řídit téměř veškeré funkce a aspekty organizace.	Je jakási nadstavba metodiky Scrum, případně LeSS nebo SAFe, která je optimalizována pro prostředí vzniku a vedení IT projektů uvnitř ŠKODA AUTO
Návrh řešení: Vzhledem k rozdílnosti obou metodik je doporučeno vydat nový metodický pokyn, který bude popisovat rozsah platnosti a způsob adaptace SAFe a bude odkazovat na již fungující mechanismy a metody v IT PEP 2.0, které nejsou v rozporu. Není cílem rozšiřovat IT PEP, protože popisuje řízení IT projektů a ty zahrnují pouze část problematiky, kterou SAFe pokrývá.	

GAP 2	
IT-PEP nedefinuje programovou a portfolio úroveň	
SAFe	IT-PEP 2.0 (Agile)
Programová i portfolio úroveň jsou základními principy metodiky SAFe a pomocí těchto úrovní může být v SAFe dosaženo cíleného škálování agilních principů na větší projekty, programy a portfolia produktů.	Ve ŠKODA AUTO neexistuje a IT-PEP nijak nedefinuje obecný přístup k řízení produktového programu nebo portfolia. Větší projekty a vývoje produktů na sobě závislé, jsou prioritizovány IT Steering Commitee.
Návrh řešení: Vytvoření Product Managementu tak, jak jej definuje SAFe z již existujících rolí v IT STEERING COMMITEE	

GAP 3	
Nejdůležitější konstrukt metodiky	
SAFe	IT-PEP 2.0 (Agile)
Nejdůležitějším prvkem – konstruktem, který SAFe přináší je Agile Release Train (ART). Viz kapitola 5. Ten je základním kamenem programu a portfolia v SAFe a je napájen agilními týmy.	Pro IT-PEP je základním konstruktem samotný agilní tým. Ekvivalent agilního vlaku je neexistující.
Návrh řešení: Zavést tzv. Essential SAFe pro menší týmy a zavést tím používání ART v metodice IT-PEP tak jak je definovaný v SAFe. Pro produktový vývoj, do kterého celkem spadá 50-125 lidí. Pro větší oddělení je pak možné zavést SAFe do úrovně Portfolio.	

6.4.2 Role – GAP analýza

Pro Agilní projekt a Agilní produktový vývoj jsou v IT-PEP 2.0 definovány stejné role. Jejich seznam a popis je uveden v tabulce číslo 8.

Tabulka 8 Role podle IT-PEP 2.0
Zdroj: ŠKODA AUTO a.s.

Sponsor	Vedoucí nebo zástupce organizační jednotky, který kryje finanční náklady na projekt ze svého rozpočtu. Sponsor je nejvyšší autorita v projektu, má také nejvyšší odpovědnost za realizaci projektu.
Business Owner	Vedoucí nebo zástupce organizační jednotky, který podává změnový požadavek směřující na nutnost zahájení implementace změny ve formě projektu. Business Owner zastupuje zájmy zákazníků, klíčových uživatelů produktu/služby/procesu a je zodpovědný za naplnění přínosů projektu.

IT Leader	Vedoucí nebo zástupce organizační jednotky v útvaru, který zodpovídá za dodržení realizace projektu, definovaných aktivit a výstupů z pohledu dodávky IT. Poskytuje zdroje na straně IT. Monitoruje dodávku řešení a zajišťuje adekvátní poměr kvality, nákladů a pracnosti dodavatele v souladu se stanovenými požadavky zákazníka. Garantuje technickou proveditelnost, kvalitu dodávky a soulad s technologickými standardy a strategií IT a digitalizace.
Steering Comitee	Rozhodovací instituce, která kontroluje konkrétní IT projekt. Skládá se ze Sponsora, Business Ownera, IT Leadera, a IT Consultanta projektu. Členové Steering Committee jsou povinni poskytovat maximální součinnost při realizaci projektu.
Product Owner	Odpovědný za agilní dodávku (produkt), nebo výstup agilního projektu, maximalizuje hodnotu produktu a práce Development Teamu; řídí rozpočet a časový plán dodávek. Reprezentuje všechny stakeholdery vůči týmu, určuje priority položek Product Backlogu. Eskaluje na úroveň Steering Committee problémy, které jsou nad rámec jeho kompetence.
IT consultant	Zajišťuje dodržení metodiky na projektu z pohledu IT, zjišťuje nedostatky, které eskaluje na úroveň Product Ownera, podílí se na výběru technologií a cílového řešení (vč. externího dodavatele), poskytuje podporu Product Ownerovi při koordinaci aktivit, konzultuje schválení a akceptaci projektových milníků / výstupů sprintů, zajišťuje požadavky na soulad se standardy ŠA. V případě nerealizace nápravných opatření ve zjištěných nedostatcích má pravomoc eskalovat i na úroveň Steering Committee.
Development team	Multidisciplinární (cross-funkční) tým složený z 3-9 jednotlivců, který je schopný samo organizace a odpovídá za

	dodávání potenciálně dodatelných inkrementů (dle Definition of Done) produktu na konci každého sprintu.
Scrum Master	Scrum Master je servant-leaderem pro Scrum Team. Koučuje tým k samo organizaci, cross-funkčnosti a pomáhá týmu odstraňovat překážky ve vývoji. Zároveň velice dobře ovládá Scrum, učí jej tým a okolí a je ambasadorem agilního přístupu v organizaci.
Delivery Manager	Koordinátor Development Teamu a v případě externího dodavatele je to hlavní kontaktní osoba externího dodavatele. Je odpovědný za svěřenou část projektu. Je povinen zajistit správný časový průběh, efektivní užití zdrojů a správné pořadí prací tak, aby byly řádně dodány požadované výstupy. Musí být dodržen správný časový a finanční rámec.

Následuje představení rolí, které jsou definovány pro konfiguraci Portfolio SAFe. Ta zahrnuje úrovně Team, Program a Portfolio. Jednotlivé úrovně jsou popsány v kapitole 5.

Tabulka 9 Přehled rolí v konfiguraci Portfolio SAFe
Zdroj: ŠKODA AUTOa.s.

Lean Portfolio Management	Představitelé nejvyšší úrovně rozhodování a finanční odpovědnosti za portfolio. Tato skupina je zodpovědná za tři primární oblasti: strategické a investiční financování, agilní portfoliové operace a štíhlé řízení. Skupina je složena z Business Owners, Enteprise Architect a zástupců agility v portfoliu Release Train Engineer.
Epic Owners	Jejich primární zodpovědnost je za koordinaci Epic, které jsou pro dané portfolio definované pomocí systému portfolio Kanban ve spolupráci s Lean Portfolio Managementem. Po akceptování Epicu pracuje Epic Owner napřímo s agilními týmy na zahájení vývojových aktivit nezbytných k realizaci tohoto epicu.

Enterprise Architect	Zajišťuje strategické technické směřování všech produktů uvnitř portfolia. Enteprise Architect může být často v roli Epic Ownera u tzv. Enabler Epiců, tedy Epiců, které nepředstavují určitou business funkcionalitu, ale např. integraci technického řešení napříč produkty.
Product Management	Interně zastupuje hlas zákazníka a spolupracuje se zákazníky a Product Ownery na porozumění a komunikaci jejich potřeb, definování systémových funkcí a jejich testování. Tato úroveň managementu je odpovědná za Program Backlog.
System Architect/Engineer	Je sdílená role, která se stará o definovaní a prosazování technických a architektonických vizí daného agilního vlaku.
Release Train Engineer	Je servant-leaderem a hlavní Scrum Master pro agilní vlak.
Business Owners	Je malá skupina stakeholderů, kteří mají obchodní zodpovědnost za řešení, ke kterému směřuje agilní vlak.
Agile Team	Multidisciplinární (cross-funkční) tým složený z 5-11 jednotlivců. Tato skupina má schopnost a oprávnění definovat, sestavit a otestovat User Story nebo Enabler pro danou iteraci.
Development Team	Hlavní část Agile Team, jde o vývojáře, testery a další specialisty, kteří pracují společně na doručení User Story a Enablerů.
Product owner	Product Owner tvoří obsah týmového Backlogu. Je zodpovědný za definování stories a prirotizaci. Je jediný z týmu, kdo hodnotí stories jako hotové.
Scrum Master	Scrum Master je servant-leaderem a coachem agilního týmu. Scrum Master pomáhá týmu odstraňovat překážky, usnadňuje týmové akce a podporuje prostředí pro vysokou efektivitu práce týmu.

Obě tabulky, číslo 8 a 9, jsou sepsány odshora dolů přesně tak, jak role kopírují hierarchii v dané metodice. Na první pohled je patrné, že pouze některé role jsou pro obě metodiky stejné, některé mají stejný popis a některé role jsou v druhé metodice neexistující. Protože jak SAFe, tak IT-PEP berou jako základní pracovní jednotku agilní tým, na nejnižší úrovni jsou role v týmu velmi podobné. Agilní týmy jsou složeny z vývojového týmu, Scrum Mastera a Product Ownera. Přesně tak, jak je to definováno v metodice SCRUM.

<table>
<tr><td colspan="2">GAP 4</td></tr>
<tr><td colspan="2">Odlišná definice a chápání role Product Owner</td></tr>
<tr><td>SAFe</td><td>IT-PEP 2.0 (Agile)</td></tr>
<tr><td>V SAFe je velká část zodpovědností Product Ownera převedena do vyšších pater produktového managementu a pravomoce Product Ownera a prostor, který je dán vývojovému týmu, je omezen hranicemi definovanými na Product Increment Planning ve spolupráci se zastoupením z úrovně Program. Jednotlivé produkty/komponenty jsou na programové úrovni zarovnávány na stejnou vizi a řízeny stejnou strategií.</td><td>V IT-PEP je role Product Ownera odvíjena čistě od jeho funkce ve Scrum. Má veškerou zodpovědnost za dodávku a naplnění obchodních požadavků na produkt.</td></tr>
<tr><td colspan="2">Návrh řešení – Zavést úroveň Program dle SAFe, kde jsou jednotliví Product Owners napřímo spojeni s Product Managementem, vztahem servant-leader a dodávky produktů tak lze z produktového pohledu lépe koordinovat.</td></tr>
</table>

GAP 5	
Role IT Consultant neexistuje v metodice SAFe	
SAFe	IT-PEP 2.0 (Agile)
Metodika SAFe nedefinuje roli IT Consultant. Částečně lze najít protnutí v náplni práce s rolí System Arcitect/Engineer, která řeší technické a architektonické náležitosti. V ostatních oblastech je však tato role rozdílná.	Pro vedení agilního produktového vývoje i pro agilní projekty je to však životně důležitá role a to proto, že externí softwarové dodávky a spolupráce s odděleními jako DevOps, databázová podpora nejsou možné řídit bez IT Consultanta na straně ŠKODA AUTO.
Návrh řešení: Pokud je součástí projektu/produktového vývoje převážně externí dodávka, je možné vytvořit v SAFe na úrovni Program speciální roli IT Consultant, která se bude převážně starat o tento externí tým. Druhá možnost je, že tuto práci bude zastávat Product Owner tak, jak je popsáno v GAP 7 a pozice IT Consultanta bude převedena do Systémového týmu, dle definice SAFe.	

GAP 6	
Steering Comitee pro projekt není definována v SAFe	
SAFe	IT-PEP 2.0 (Agile)
Ekvivalentem v SAFe je Lean Portfolio Management a z části také product management.	Ve ŠKODA AUTO IT-PEP je Steering Comitee hlavní Rozhodovací instituce, která kontroluje konkrétní IT projekt. Skládá se ze všech stake holderů projektu a pro fungování projektu je tak její existence zásadní.
Návrh řešení: Steering Comitee jako taková bude v rámci SAFe projektu rozdělena do rolí Lean Portfolio Mangement a Product Management.	

GAP 7	
Role Delivery Managera není definována v metodice SAFe	
SAFe	IT-PEP 2.0 (Agile)
Žádný ekvivalent Delivery Managera tak, jak je popisován v IT-PEP neexistuje.	Podle IT-PEP řídí Delivery Manager externí dodávku na straně dodavatele. Je součástí externího týmu. Z pravidla se nejedná o zaměstnance ŠKODA AUTO.
Návrh řešení: Není třeba obsazovat roli Delivery Managera na produktový vývoj/projekty řízené dle SAFe. Pokud je součástí projektu/produktového vývoje externí dodavatel, řešením je, že tento externí tým sám definuje, kdo bude komunikovat za tým a řídit dodávku. Na straně interního týmu ve ŠKODA AUTO bude s externím týmem komunikovat Product Owner případně IT Consultant. Na straně externího týmu může komunikaci a dodávku řídit externí Product Ownerem nebo Scrum Master.	

6.4.3 Události

IT-PEP i SAFe pro řízení nejnižší týmové úrovně používají techniky metodiky Scrum. Je proto pochopitelné, že v oblasti událostí (v prostředí ŠKODA AUTO se používá název ceremonie) jsou oba tyto metodické rámce velmi podobné. Následující seznam uvádí události, které jsou stejné pro obě metodiky:

- Sprint
- Sprint planning
- Daily Scrum
- Sprint review
- Sprint retrospective

I přesto, že SAFe nepoužívá ve svých materiálech označení „sprint“, ale „iteration“, lze podle účelu těchto ceremonií, události na týmové úrovni prohlásit za téměř identické.

IT-PEP logicky nedefinuje tolik událostí jako SAFe, protože, jak již bylo uvedeno, nedefinuje programovou ani portfolio úroveň. Na vyšších úrovních tak IT-PEP postrádá ucelená pravidla. Pro potřeby škálování a řízení větších produktů, portfolia a programů IT-PEP definuje velmi obecně pouze:

- Stakeholders Meeting
- Community of Practices

Oba tyto koncepty, i když opět pod jiným jménem, jsou zakomponovány i do metodiky SAFe. Ekvivalentem Stakeholders Meeting je v SAFe meeting Product Managementu, který je tvořen právě zástupci stakeholderů. Community of Practises je poté obecně užívaná metoda i v metodice SAFe.

Z pohledu událostí jednotlivých vývojových iterací se tak může zdát tailoring SAFe na IT-PEP jako jednoduchá záležitost. Důležité je také ale vzít v úvahu milníky, které IT-PEP definuje. Jak již bylo řečeno agilní principy využívají v rámci IT-PEP dva způsoby dodávky – agilní projekt a agilní produktový vývoj. Druhý zmíněný způsob dodávky nemá definované milníky a jedná se o čistý agilní vývoj postavený na přírůstcích produktu. Agilní projekt na druhou stranu je jakýsi mezistupeň mezi vodopádovým a agilním vývojem a má tak definované milníky, které jsou pro řízení agilních projektů ve ŠKODA AUTO důležité.

Milníky agilního projektu v IT-PEP jsou:

- SZ – milník schválení fáze realizace projektu
- SOO – milník schválení fáze zavedení projektu
- PA – milník schválení fáze stabilizace projektu

Milníky definované v SAFe:

- Product Increment (PI) milníky
- Milníky s pevným datem
- Milníky učení

Z pohledu milníků bude tedy nejtěžší uzpůsobit SAFe tak, aby dosahoval těchto milníků. To může být ale problematické, protože SAFe se milníky příliš nezabývá. I přesto, že jsou v SAFe obecné milníky definované, jedná se o jiné pojetí. Jednoduché propojení je zde mezi milníky SOO a PI milníky. IT-PEP 2.0. dělí SOO na menší SOOf – milník vždy po několika Sprintech (každý f-tý) ty odpovídají průběžným releasům, které se dějí u agilního vlaku.

Protože IT-PEP nedefinuje žádné události mimo úroveň týmů, nebudou v této části GAP analýzy uvedeny jednotlivé rozdíly. Nejdůležitějším doporučením této části je zavedení Product Increment Planningu do metodiky IT-PEP. Jde o základní událost, která umožnuje definovat vize a směr vývoje více vývojových týmů pohybujících se ve stejném programu. S tím se pojí také ceremonie Scrum of Scrum, kterou je při vytvoření programové úrovně začít dodržovat, aby byla komunikace a koordinace jednotlivých vývojových týmů možná.

6.4.4 Shrnutí GAP analýzy

Byla provedena analýza rozdílů metodiky SAFe a metodického rámce IT-PEP 2.0. Tato analýza ukázala, že rozdílů mezi metodikami je opravdu hodně. Bylo identifikováno sedm hlavních rozdílů. Kapitola 6.4 popisuje všechny rozdíly, které byly identifikovány a navrhuje řešení, jak lze k integraci u jednotlivých rozdílů přistoupit.

6.5 Integrace SAFe a IT PEP 2.0

V této části jsou uvedena doporučení pro integraci metodiky SAFe a IT PEP 2.0, které vychází z GAP analýzy (6.4) a z analýzy případové studie, která je uvedena v následující kapitole této práce. Jak vyplývá hned z prvního obecného rozdílu metodik, je doporučeno vytvořit nový metodický pokyn, speciálně vytvořený pro týmy a oddělení jichž se bude adopce metodiky SAFe týkat. Tento pokyn by popisoval rozsah platnosti, a hlavně způsobu implementace SAFe na základě best-practices. Tento nový pokyn bude plně kompatibilní s již zavedenými a fungujícími metodami v IT PEP 2.0.

Doporučení, která byla definována a ze kterých by nový metodický pokyn měl vycházet jsou:

- Pro každý vývojový tým vznikne na týmové úrovni role Scrum Mastera a ProductOwnera.
- Product Owner řídí externí tým a jeho dodávku. Pozice Delivery Managera je zrušena.
- Na týmové úrovni agilní týmy pořádají ceremonie tak, jak je definováno v metodice IT-PEP.
- ScrumMasteři jednotlivých týmů se účastní meetingu Scrum of Scrum, kde probíhá koordinace těchto týmů.
- Agilní týmy jsou vytvořeny tak, aby v rámci programu, do kterého náleží, mohli pracovat na jakémkoliv produktu/komponentě, který je v rámci daného programu vyvíjen.
- IT Consultanti společně tvoří Systémový tým dle definice SAFe.
- Na programové úrovni jsou vytvořeny role Release Train Engineer, System Architect a Product Manager.
- Na PI Planning je definována vize programu, jsou definovány agilní týmy, které budou napájet agilní vlak. Definovaná vize je budována pomocí Features, které jsou shromážděny v Program Backlog.
- Pro implementaci SAFe je vhodné užít SAFe Implemtation Roadmap

Předpoklady úspěšné implementace jsou:

- Správně nastavené školení a tréninky pro všechny členy agilní transformace
- Vytvoření Community of Practice, tak jak jej definuje SAFe
- Definice jasné vize a strategie na Portfolio i Program úrovni
- Podpora silného leadershipu zaměřeného na změny

Návrh nového rozdělení rolí v SAFe Portfolio v souladu s IT-PEP 2.0

V této části je uveden návrh nového rozdělení rolí na agilních projektech a agilním produktovém vývoji, které mají být řízeny pomocí metodiky SAFe a v souladu s vnitropodnikovým metodickým rámcem IT-PEP. Návrh je vytvořen tak, aby byl

přechod ze stávajícího rozdělení rolí dle IT-PEP co nejjednodušší a zároveň zachoval integritu metodiky SAFe a jejích principů.

Týmová úroveň

- Dev Team – Původní nezměněná role, podle SAFe
- Scrum Master – Původní nezměněná role, podle SAFe
- Product Owner – Původní nezměněná role, která nyní působí i v rámci Product Managementu
- IT Consultant – Nová role, podle definice v IT – PEP 2.0

Programová úroveň

- RTE – Nová role, podle SAFe
- Product Management – Nově definovaná úroveň managementu, složena z Product Owners jednotlivých agilních týmů
- System Architect/Engineer – Nová role, podle SAFe
- Business Owners – Původní nezměněná role, podle SAFe

Portfolio úroveň

- Lean management – Nově definovaná úroveň managementu, složena z původních rolí Sponzora a Steering Comitee
- Epic Owners – Nově vyšší úroveň role Business Owners, složena z vybraných Business Owners s rozšířenou zodpovědností
- Enterprise Architect - Přejmenování role IT Leadera

6.5.1 Integrace projektů LeSS do SAFe

Ve ŠKODA AUTO jsou také projekty a produktové skupiny vedené metodikou LeSS. Jak již bylo uvedeno jedná se o projekty převážně lokální bez návaznosti a provázanosti na koncernový vývoj. Pokud jsou tyto projekty obchodně nebo technologicky související se SAFe projekty, je doporučené co nejvíce je přiblížit

procesům metodiky SAFe. Pokud ale není jednoznačný důvod pro změnu těchto projektů, projekty LeSS by měly zůstat samostatné, bez integrace na metodiku SAFe.

Zde je uvedeno několik bodů, jak lze metodiku LeSS přiblížit a zarovnat s metodikou SAFe:

- Skupina LeSS produktů bude zahrnuta do Portfolio SAFe, jako samostatný agilní vlak (ART) s ohledem na rozdílné vnitřní fungování. Z pohledu portfolio managementu SAFe se však bude řídit stejnými procesy.
- Směřování produktové skupiny LeSS se bude rozhodovat na PI Planning, tak jako další ART, aby byla synchronizována vývojová činnost, vize a cíle jednotlivých iterací.
- LeSS Product Backlog se změní na SAFe Program Backlog
- Scrum Masteři jednotlivých LeSS týmů definují, jak vytvoří roli Release Train Engineera (RTE), neboli hlavního Scrum Mastera jednotlivého agilního vlaku (ART) tak jak je to v SAFe. Musí být definována jasná pravidla, kdo povede komunikaci za tuto roli.
- SAFe artefakt Product Increment může nahradit velký release pro LeSS týmy, pro které se tím změní jen nová orientace na tento časový rámec. Kromě toho budou LeSS týmy dále pracovat v rámci svých sprintů a budou provádět všechny události a procesy sprint od sprintu, tak jak LeSS definuje.

6.6 Závěr GAP analýzy

V kapitole číslo 6 je představeno prostředí společnosti ŠKODA AUTO a je prezentována analýza rozdílů mezi vnitropodnikovým metodickým rámcem IT-PEP 2.0 a metodikou Scaled Agile Framework do úrovně Portfolio, protože tato konfigurace byla v rámci konzultací identifikována jako nejvhodnější.

V celkem sedmi Gapech (rozdílech) jsou prezentovány hlavní rozdíly v definici a pojetí rolí. Samostatná kapitola je věnována událostem obou metodik. Z této analýzy vyplývá doporučení vytvoření vlastního metodického pokynu pro metodiku SAFe a ne její integrace do již stávajících procesů.

7 CASE STUDY zavedení SAFe na projekty ve ŠKODA AUTO

V této části práce je uveden rozbor probíhající agilní transformace vybraného vývojového oddělení společnosti ŠKODA AUTO. Nejprve je představena metoda, kterou byla získána data a okruhy otázek, na které se analýza zaměřovala (7.1.). Dále je představen útvar společnosti ŠKODA AUTO, kterého se transformace týká (7.2) a rozbor současné situace implementace škálovatelných agilních metodik (7.3). Na konec je uvedeno hodnocení dosavadního průběhu adopce škálovatelných agilních metodik (7.4) a z něho plynoucí doporučení pro ostatní útvary a vývojové týmy ŠKODA AUTO, kterých se zavádění SAFe nebo LeSS týká (7.5).

7.1 Metoda výzkumu

Bylo definováno pět hlavních výzkumných otázek, které byly položeny zástupům ŠKODA AUTO formou průběžných konzultací. Cílem celkem osmi konzultací, kdy každá trvala 45 až 60 minut, bylo analyzovat situaci v týmu, který se otázkou adopce large-scale metodiky zabývá, zhodnotit proces integrace a vytvořit tak analýzu zavádění large-scale agilních metodik v prostředí ŠKODA AUTO. Pro tento cíl bylo definováno těchto pět okruhů otázek:

- Jaká byla předchozí metoda řízení a proč jste se pro změnu rozhodli?
- Jaké úrovně, role, události a artefakty metodiky SAFe jste implementovali?
- Jaké byly potíže a výzvy samotné implementace?
- Jak hodnotíte tuto transformaci?
- Jaké změny a doporučení plynou pro implementace SAFe v dalších týmech ŠKODA AUTO?

Tyto otázky byly formou konzultací zodpovězeny pracovníky ŠKODA AUTO na pozicích Leader agilní transformace, Release Train Engineer, Scrum Master a Projektový koordinátor agilních projektů. Dalším zdrojem informací, ze kterých bylo čerpáno, byly vnitropodnikové dokumenty a prezentace týmů, které se touto problematikou zabývaly. Dále probíhala neustálá online komunikace prováděná za účelem další specifikace, vyjasnění a ověření platnosti informací.

7.2 Prostředí a kontext oddělení

Pro tuto případovou studii bylo vybráno konkrétní oddělení společnosti ŠKODA AUTO. Studie se týká oddělení VC, které se ve ŠKODA AUTO zabývá problematikou Customer Experience Managementu (dále jen CEM) a je zodpovědné za podporu prodejní a marketingové oblasti ŠKODA AUTO poskytováním různých předprodejních, prodejních a poprodejních řešení. Oddělení VC je součástí většího funkčního celku – skupiny V, která má na starost prodej. Filozofií oddělení VC je vytvořit bezproblémový a poutavý zákaznický zážitek, který povede k dalším obchodním příležitostem v celé organizaci. To zahrnuje plánování, budování a zavádění prvotřídních produktů a řešení na podporu současných i budoucích obchodních modelů ŠKODA AUTO. Oblast zájmů uvnitř oddělení VC se dá rozdělit do dvou hlavních větví.

1. B2C segment – jde o vývoj částí digitálního ekosystému a aplikací jako Car Configurator, jednotlivé weby spojené s CEM, digitální asistent LAURA a další
2. B2B segment – vývoj komplexního CRM řešení

Původně byl celý program řízen tradičními přístupy vývoje produktů. Přibližně v roce 2017 se začaly objevovat první snahy zavádění agilních metodik na týmové úrovni. Brzy však byly identifikovány potřeby škálování agilních principů. Mezi tyto aspekty patří:

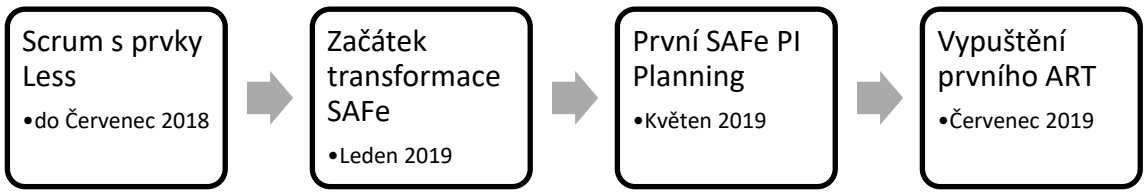
- Potřeba prioritizace vývoje Business Use Cases na vyšší úrovni než týmové (samotný Scrum nedokáže řešit prioritizaci na strategické úrovni)
- Zvýšit předvídatelnost dodávek a obecně vývoje produktů, které zavedení agilních principů na týmové úrovni nepřineslo
- Zlepšení společného směřování vývojových aktivit jednotlivých týmů, soustředit týmy na jednotnou vizi

Celý program CEM oddělení VC byl vybrán pro implementaci large-scale agilních principů a hlavní metodikou, ze které jsou tyto principy uplatňovány je Scaled Agile Framework (SAFe). Jedná se o jedno z prvních oddělení ŠKODA AUTO, které

přistoupilo k systematické implementaci agilních principů pro programové a později portfolio úrovně.

7.3 Implementace SAFe

Oddělení VC (7.2.) se v současnosti nachází v procesu implementování SAFe principů. Tato kapitola popisuje, v jaké fázi implementace se oddělení VC nachází. Jaké principy, artefakty, role a události metodiky SAFe jsou již používány a dále jsou popsány identifikované problémy a výzvy, se kterými se společnost setkala.



Obrázek 16 Časová osa large-scale agile transformace ve vybraném oddělení ŠKODA AUTO
Zdroj: Vlastní zpracování podle dat ze ŠKODA AUTO a.s.

Na obr. 17 lze vidět, jak se vyvíjela metodika řízení produktového vývoje a adopce large-scale agile metodik v útvaru VC. V současnosti je v tomto útvaru jasně daný záměr a tím je implementace SAFe. V současnosti je metodika implementována do úrovně Program.

Na základě konzultací a poskytnutých materiálů bylo identifikováno těchto šest large-scale agile praktik, které již byly integrovány do řízení produktového vývoje ve vybraném oddělení:

1. Plná alokace rolí na úrovni Program SAFe
2. Product Increment Planning
3. Scrum of Scrum
4. Program Backlog
5. Vypuštění prvního ART
6. Sestavení Systémového týmu

Plná alokace rolí na úrovni Program SAFe

Speciálně v prostředí ŠKODA AUTO, kde existují definice rolí pro vedení IT projektů, ale ne konkrétní definice rolí pro řízení programů / produktových portfolií, je těžké alokovat nové role, nebo role měnit podle nové metodiky. Proto je důležité při adaptaci nového frameworku nevytvářet nové nedefinované role a současně dodržovat jejich plnou alokaci. Na projektu současně působí a jsou zavedeny všechny základní role úrovně program tak, jak ji SAFe definuje. Jedná se o role Release Train Engineer, System Architect a Product Management. Na týmové úrovni se jedná o role Development Tým, Scrum Master a Product Owner.

Product Increment Planning

První PI Planning proběhl v květnu 2019 a předcházel tak vypuštění ART. Na tomto meetingu byly definovány a odhadnuty první Features pro Program Backlog jejichž plnění povede ke stejné programové vizi, které všichni stakeholdeři pro produkty skupiny VC mají. Definování těchto Features na prvním PI Planningu byl zásadní milník, který později umožnil vytvoření agilního vlaku. PI Planningu se účastní všichni stakeholdeři, komponentové týmů a zástupci jinak zapojených stran. Celkem se jedná o přibližně 150 lidí, z nichž větší část jsou externí týmy a dodavatelé. To klade na celou akci ještě větší koordinační nároky.

Scrum of Scrum

Scrum of Scrum je považován za jednu ze základních praktik škálovatelné agility. Týmy ve VC zavedli tuto ceremonii. Účastní se všichni Scrum Masteři jednotlivých týmů a koordinují dohromady vývojové aktivity všech týmů, které tvoří ART. Těchto schůzek se nahodile účastné také Program Owners. V současnosti probíhá intenzivní náborová kampaň zaměřená právě na Scrum Mastery. Nyní na jednoho Scrum Mastera připadá více než jeden agilní tým. Existuje však snaha o to, aby jeden Scrum Master působil pouze uvnitř jedné komponenty.

Program Backlog

Oddělení VC definovalo položky společného programového Backlogu. Navíc v současnosti vzniká předběžné mapování a definice Epics, které jsou vyšší úrovní Features a umožnují řídit portfolio produktů a směřovat ho ke stejným strategickým cílům a vizím. Features dnes VC využívá ve třech podobách. Jde o standartní businessové funkcionality, architektonické enablery a položky, které pomáhají aplikacím a systémům dorovnávat technologický dluh, protože část aplikací a systémů je starších.

Vypuštění prvního ART

Útvaru VC se podařilo první Agile Release Train vypustit v průběhu července 2019. Vlak je „poháněn" celkem patnácti agilními týmy a v součtu se jedná o přibližně 120 lidí. Tyto agilní týmy jsou navíc orientovány ještě podle komponenty / produktu, který v rámci programu existuje. Hlavní osobou, která řídila vypuštění prvního ART je zároveň leader celé této probíhající transformace. Release byl uskutečněn za podpory externích konzultantů ze společnosti, která s implementací SAFe pomáhala i v ostatních společnostech koncernu VW.

Sestavení Systémového týmu

Pro podporu naplňování delivery pipeline agilního vlaku existuje na týmové úrovni systémový tým složený z IT konzultantů. Tato role je v prostředí ŠKODA AUTO velmi dobře etablovaná, a proto byla integrována do systémového týmu. Jejich odpovědností je zefektivňování komunikace mezi agilními týmy a dalšími podpůrnými IT týmy (DevOps, security, síťová podpora atd.) ŠKODA AUTO a tím agilnímu vlaku pomáhat v naplňování cílů.

7.4 Klíčové faktory úspěchu a hlavní výzvy

V průběhu implementace SAFe a integrace této metodiky do prostředí společnosti ŠKODA AUTO se oddělení VC setkalo s několika výzvami, které transformace přinesla.

Na základě konzultací bylo vydestilováno těchto pět oblastí, které jsou zároveň klíčové pro úspěch celé transformace.

1. Nesjednocená strategie a vize skupiny
2. Nedostatečný leadership pro změnu
3. Mindset zástupců businessu
4. Vodopádové řízení DevOps
5. Trénink a podpora vzdělávání

Nesjednocená strategie a vize skupiny

Útvar VC, kterého se týká tato agilní transformace, je členem oblasti V ve ŠKODA AUTO. Tato oblast zaštiťuje prodej a marketing. Chybějící jednotná vize na úrovni celé V oblasti je výzva, se kterou se při large-scale implementaci SAFe útvar potýká. Příkladem může být existence dvou zákaznických portálů pro český trh. V dlouhodobém horizontu by právě SAFe a zavedení a sjednoceni úrovně Portfolio mělo přinést potřebné srovnání vize na skupinové úrovni. Nesjednocená strategie se týká i produktových business týmů a IT týmů, kdy v současnosti je stále manažerskou výzvou vytvořit prostředí, kde zástupci business a IT budou dohromady efektivně pracovat na doručení produktu.

Silný leadership nakloněný změně

Jako největší výzva transformace se v současnosti ukázalo vytvoření silného leadershipu, který je nakloněný large-scale agilním změnám a transformaci na portfoliové i programové úrovni. Pokud nebude managementem vytvořeno vhodné prostředí, selže dříve nebo později i sebelepší metodika.

Mindset zástupců businessu

Na programové úrovni, kde zástupci businessu mají největší kontakt se samotným vývojem produktu je výzvou, aby se změnilo, jak produktový management chápe inkrementální agilní vývoj. Je třeba dbát na to, aby business chápal inkrementální způsob vývoje, protože s tím je spojené nastavení, fungování, a hlavně očekávání všech členů produktového týmu. Jedná se více o kulturní věc než o praktické uchopitelné nástroje, nebo procesy a tato změna může trvat delší dobu.

Vodopádové řízení DevOps

Správě nastavený vztah mezi development a operations, tedy DevOps je v SAFe důležitý předpoklad pro správnou dlouhodobou podporu aplikací a systémů, které dodává ART. Současná situace ve ŠKODA AUTO je taková, že IT oddělení operations funguje převážně napříč společností ve vodopádovém stylu. Pro oddělení VC je tak výzvou zařadit ho bok po boku ke svým agilním, případně komponentovým týmům. Doposud má pouze jedna aplikace, která je součástí ART vlastního specialistu na DevOps. Je zde tedy do budoucna velký prostor pro zlepšení i přesto, že tato změna bude opět velmi rozsáhlá a komplexní, protože se dotýká fungování celé společnosti.

Trénink a podpora vzdělávání

V první fázi transformace proběhlo školení a odborné školení všech agilních týmů a ostatních členů agilního vlaku. Pro vedoucí pozice byly připraveny speciální přípravné large-scale školení a pro řadové členy ART šlo o jakýsi awarness trénink, tedy představení změny, která se chystá. Z tohoto pohledu lze říci, že tato fáze proběhla dostatečně dobře. Průběžné vzdělávání je však kritické pro úspěch transformace. V tomto ohledu je pro společnost velkou výzvou, jak zvládne školit management a vedoucí jednotlivých útvarů. Souvisí to s prvním a druhým bodem této části. Je třeba brát zřetel i na to, že na ART je napojeno spoustu externích vývojových týmů a ani zde se nesmí jejich trénink a školení potřebné k této transformaci podcenit. Nakonec lepší trénink a znalost metodiky, může přispět k rychlejší změně kultury a mindsetu celé společnosti.

7.5 Hodnocení implementace

Tato část se věnuje objektivnímu zhodnocení dosavadního průběhu implementace metodiky SAFe do vybraného oddělení VC ve společnosti ŠKODA AUTO. Prezentované výsledky byly získány opět formou konzultace. Dosavadní implementace metodiky SAFe přinesla:

- Maximální transparentnost celého programu a všech produktů vůči stakeholderům

- Zlepšení visibility závislostí produktových i technologických, díky tomu lze efektivněji plánovat a lépe definovat společné vize
- Naplnění závazku v prvním Product Increment cyklu z 84%[2]
- Obecně lepší odhad v predikci doručování agilních týmů
- Zlepšení komunikace napříč všemi týmy v ART a jejich koordinace

Dále předmětem hodnocení je definování oblastí, ve kterých je stále prostor pro zlepšení. Byly identifikovány tyto oblasti:

- Role IT konzultantů v System Team by měla více podporovat zlepšování delivery pipeline a tím podporovat fungovaní ART
- Lepší zajišťování zdrojů – v současnosti velká míra tzv. „bodyshopů“, kde externí dodavatel nenese žádnou zodpovědnost za dodávku
- Podpora nástrojů –Celý SAFe Backlog je veden v nástroji JIRA, který je sice podporován koncernově a je primární aplikací ve ŠKODA AUTO, nicméně kvůli velkému množství ostatních projektů a implementaci formou on-premise se potýká dlouhodobě s výkonnostními problémy. Zároveň v JIRA chybí modul, který by byl čistě zaměřen na podporu SAFe PI plannings (např. Agile Hive)
- Lepší náborový proces klíčových rolí jako Product Owner, Scrum Master a Release Train Engineer

7.6 Závěr případové studie

Tato část patří závěrečnému zhodnocení případové studie. Byl proveden výzkum v oddělení, které prochází procesem adopce large-scale agilních principů sjednocených pod metodikou SAFe. Byly sledovány konkrétní postupy škálování agility, které byly využity a následně byly zhodnoceny faktory úspěchu a výzvy, kterým společnost čelí.

[2] Tato hodnota je stanovena jako poměr dosažených cílů vůči plánovaným. Je stanovována na PI Planning kde se každému commitmentu přiřazuje odhadovaná hodnota na škále 1-10. Tato čísla jsou pak mezi sebou porovnána.

Sledovaná implementace je stále v prvotních fázích. Vypuštění prvního ART je pouze začátkem dlouhé cesty implementace SAFe na Portfolio nebo dokonce Large Solution úrovni. I proto není možné v tuto chvíli dělat komplexní zhodnocení nebo přesně vyčíslit benefity této změny. Budoucí analyzování a podrobení detailnímu zkoumání je více než doporučené. V každém případě již nyní bylo možné identifikovat některé pozitivní dopady této změny, stejně tak jako výzvy, které ještě budou potřeba překonat. Některé klíčové faktory úspěchu, které byly identifikovány u dalších velkých firem v práci (Diekert, Paasivaara a Lassenius, 2016) se ukázaly jako klíčové i v prostředí ŠKODA AUTO. Jde zejména o kulturu a myšlenkové nastavení středního a vyššího managementu, vodopádové řízení některých částí společnosti a obecnou chuť ke změně. Na druhou stranu se časté faktory neúspěchu, jako nedostatek tréninku a špatná komunikace týmů, které jsou v práci (Diekert, Paasivaara a Lassenius, 2016) zmiňovány jako časté, v prostředí ŠKODA AUTO nepotvrdily.

Na vědomí je potřeba brát fakt, že vývojové prostředí ve společnosti ŠKODA AUTO je velmi komplexní. Z toho důvodu, že jde historicky převážně o výrobní firmu, existuje zde stále probíhající přerod společnosti na digitální produktově orientovanou společnost. Tento fakt sebou nese spoustu rizik a výzev, které se například v čistě softwarových společnostech nemusí vůbec objevovat a celý proces implementace SAFe mohou zpomalovat. Dalším faktorem, který je potřeba brát v potaz, je to, že vývojové týmy ve zmíněném agilním vlaku jsou velmi často externí. Tento fakt, stejně jako velké množství stakeholderů ve ŠKODA AUTO, může také přispívat k pomalejšímu procesu adopce large-scale agile principů a je zároveň velkým rizikem ztráty know-how. V momentě, kdy by bylo potřeba změnit vývojový tým, by vznikly velké náklady na přeškolení nového týmu, uvedení do problematiky SAFe a opět by to celý postup transformace zpomalovalo.

Tato případová studie ukázala, že integrace metodiky SAFe a IT-PEP není zdaleka tak problémová. Obecně lze říci, že až na drobné výjimky by ostatní oddělení společnosti neměly mít problém s přechodem na SAFe na týmové a programové úrovni. Bohužel se však tato transformace doposud týkala pouze jednoho oddělení

a jejího produktového programu. V momentě, kdy přijde první pokus expandovat mimo oddělení VC na ostatní oddělení skupiny V a budovat tím úroveň Portfolio, mohou se začít objevovat nové překážky. Může jít o jiné přístupy skupinových Steering Comitee, které se doposud na implementaci nijak nepodílely. Proto by tato a podobná expandovaní měla být zastřešována například oddělením Agile Center of Excellence, které by pomáhalo se zaváděním portfolio úrovní.

I z tohoto důvodu je nutné pozitivní nebo negativní dopady transformace analyzovat v několika časových horizontech a brát na vědomí, že i přes slibně vypadající rozjezd se celý projekt může stát neúspěšným.

8 Shrnutí výsledků

Tato diplomová práce byla vypracovávána ve spolupráci se společností ŠKODA AUTO a cílem praktické části byla analýza integrace metodiky SAFe do firemního prostředí. Společně s teoretickou částí bylo dosaženo těchto cílů práce:

- Uvést popis základních vlastností tradičních a agilních metodik vývoje SW.
- Popsat oblast tzv. large-scale agile metodik a podrobněji vysvětlit principy metodiky SAFe.
- Vytvořit analýzu rozdílů mezi metodikou SAFe a vnitropodnikovým metodickým rámcem společnosti ŠKODA AUTO.
- Analyzovat implementaci metodiky SAFe ve vybraných týmech společnosti ŠKODA AUTO.

Jednotlivé cíle byly rozděleny do samostatných kapitol práce. Popisu základních vlastností a porovnání tradičních a agilních metodik se věnují kapitoly 3 a 4. Vznik oblasti large-scale agile metodik popisuje kapitola 4.5. Samostatná kapitola 5 je věnována metodice Scaled Agile Framework, jakožto zástupci large-scale agile metodik a zároveň metodika, která je následně využita v praktické části. Následuje praktická část práce. Analýza rozdílů je uvedena v kapitole 6 a případová studie probíhající agilní transformace v kapitole 7.

V analýze rozdílů jsou porovnány hlavní principy obou metodických rámců, role, které se v metodikách používají a také události, které obě metodiky definují. Pro každou oblast je sestaven seznam rozdílů. Z těchto rozdílů dále vyplývají doporučení, jak lze k budoucí integraci těchto metodik přistoupit. Tato doporučení byla definována i na základě provedené případové studie.

Hlavním přínosem této diplomové práce je vytvoření uceleného přehledu integrace metodiky SAFe do vnitropodnikové metodiky IT-PEP pro zaměstnance a vývojové týmy společnosti ŠKODA AUTO. Tato diplomová práce může sloužit jako pomocný materiál při adopci metodiky SAFe v jiných útvarech/odděleních/týmech společnosti ŠKODA AUTO. Identifikuje totiž riziková místa, poukazuje na mezery a

rozdíly v metodice (GAP analýza) a definuje návrhy na sjednocení a harmonizaci s metodikou IT PEP.

Tato práce byla do jisté míry ovlivněna omezeným počtem studií a odborné literatury věnující se tématu large-scale agile. Tento fakt je dan tím, že je tato problematika stále v rané fázi rozvoje. I přesto, že se jedná o trend, který hýbe celým odvětvím, neexistuje dostatečné množství vědeckých studií. Proto je práce v teoretické části zaměřena na menší počet zdrojů a část praktická není porovnávána a srovnávána s podobným typem analýz. Výzvy spojené se zaváděním large-scale agile metodik vyžadují podrobnější vědecká šetření.

9 Závěry a doporučení

Tato práce identifikovala hlavní rozdíly mezi metodikou SAFe a vnitropodnikovým metodickým rámcem IT-PEP. Jsou definovány konkrétní rozdíly a navrhnuty způsoby, jak lze tyto rozdíly řešit. Dále je zpracována případová studie, která se zaměřuje na oddělení, které v současnosti prochází transformací a cílem této transformace je integrovat do svých procesů prvky metodiky SAFe. Na základě konzultací byly identifikovány praktiky, procesy, role a události metodiky SAFe, které již byly oddělením adoptovány a týmy je používají ke svému řízení. V této části práce je popsán také milník, kterým bylo vypuštění agilního vlaku, který je základním předmětem zájmu metodiky SAFe a veškeré další procesy se ho týkají.

Na základě této analýzy byla definována doporučení pro ostatní oddělení společnosti ŠKODA AUTO. Práce vycházela z úvahy rozšířit IT PEP o principy SAFe. V průběhu zpracovávání této práce a na základě rozdílové analýzy je však doporučeno vytvořit nový samostatný metodický pokyn, který bude metodice SAFe určovat vymezené hranice v prostředí společnosti ŠKODA AUTO a nebude ovlivňovat efektivitu a výkonost této metodiky a měnit její oficiální postupy a principy.

Bylo zjištěno, že výzvy a problémy implementace se kterými se oddělení společnosti ŠKODA AUTO potýká, se do velké míry shodují s těmi, které jsou popisovány v odborné literatuře. Jedná se zejména o nesjednocené strategie a vize společnosti (v tomto případě části společnosti ŠKODA AUTO). Odpor ke změně na manažerských postech a slabý leadership, který by tuto agilní změnu uvnitř společnosti podporoval.

Na základě zjištěných dat lze konstatovat, že dosavadní průběh adopce metodiky SAFe je ve zkoumaném oddělení úspěšný. Vypuštění agilního vlaku podle standardů metodiky SAFe je ukázkou toho, že i ve velmi komplexním prostředí společnosti ŠKODA AUTO je možné úspěšné zavést a ukotvit principy nového metodického rámce. Je však potřeba podrobit celou transformaci dalšímu zkoumání v delším

časovém intervalu. Praktická část i případová studie byly omezeny rozsahem komunikace se zástupci společnosti ŠKODA AUTO a tím, že se implementace nacházela v rané fázi. Jistě by bylo možné získat podrobnější informace k celé agilní transformaci. Zde se zcela určitě nabízí prostor pro další výzkum. Tím může být například detailní dotazníkové šetření podané všem členům vybraného oddělení, které transformací prochází. Je doporučeno toto šetření provést s několika časovými rozestupy. Výsledky dotazníkového šetření mohou doplnit celkový obraz úspěšnosti transformace, protože data od pracovníků na nižších úrovních nebyla součástí zkoumání této práce.

Dosažené výsledky, které byly představené v této práci byly kladně ohodnoceny konzultantem společnosti ŠKODA AUTO a poznatky uvedené v této práci budou nadále sloužit společnosti v procesu implementace metodiky Scaled Agile Framework. Úplné hodnocení spolupráce od konzultanta ze společnosti ŠKODA AUTO je uvedené v přílohách této práce.

10 Seznam použité literatury

1. ALQUDAH, Mashal a Rozilawati RAZALI. A Review of Scaling Agile Methods in Large Software Development. *International Journal on Advanced Science Engineering Information Technology* [online]. 2016, 6(6) [cit. 2019-10-25]. ISSN 2088-5334.

2. BECK, Kent. *Extrémní programování*. Praha: Grada, 2002. Moderní programování. ISBN 80-247-0300-9.

3. BOEHM, B. a R. TURNER. Management Challenges to Implementing Agile Processes in Traditional Development Organizations. *IEEE Software* [online]. 2005, 22(5), 30-39 [cit. 2019-11-27]. DOI: 10.1109/MS.2005.129. ISSN 0740-7459. Dostupné z: http://ieeexplore.ieee.org/document/1504661/

4. BOEHM, Barry. A spiral model of software development and enhancement. *Computer*.IEEE. 1988, 61-72. ISSN 1558-0814.

5. COTTMEYER, Mike. *12 Key Reasons Companies Are Adopting Agile* [online]. 30.1.2011 [cit. 2019-09-26]. Dostupné z: https://www.leadingagile.com/2011/01/the-12-key-reasons-companies-adopt-agile/

6. DIKERT, Kim, PAASIVAARA, Maria a LASSENIUS Casper. Challenges and success factors for large-scale agile transformations: A systematic literature review. *Journal of Systems and Software.* New York: Elsevier North Holland, 2016, 87-108. ISSN 0164-1212

7. ICT odborníci v České republice [online]. Český statistický úřad, 15.08.2019 [cit. 2019-09-26]. Dostupné z: https://www.czso.cz/csu/czso/ict-odbornici

8. KADLEC, Václav. *Agilní programování: metodiky efektivního vývoje softwaru.* Brno: Computer Press, 2004. ISBN 8025103420.

9. KALENDA, Martin, HYNA, Petr a ROSSI, Bruno. (2018). *Scaling agile in large organizations: Practices, challenges, and success factors*. Journal of Software: Evolution and Process. 10.1002/smr.1954.

10. KNASTER, Richard. *SAFe 4.0 distilled: applying the Scaled Agile Framework for Lean software and systems engineering*. Boston, MA: Addison-Wesley, [2017]. ISBN 978-0-13-420942-5.

11. LEFFINGWELL, Dean. *SAFe reference guide: scaled agile framework for lean software and systems engineering*. Boston, MA: Addison-Wesley, [2016]. ISBN 978-0-13-451054-5.

12. LEFFINGWELL, Dean. *Scaling software agility: best practices for large enterprises.* Upper Saddle River: Addison-Wesley, 2007. The Agile software development series. ISBN 0-321-45819-2

13. Manifesto for Agile Software Development. *Manifesto for Agile Software Development* [online]. 2001 [cit. 2019-08-28]. Dostupné z: https://agilemanifesto.org/

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

15. NICOLETTE, David. *Software development metrics*. Shelter Island: Manning, [2015]. ISBN 9781617291357.

16. PAPADOPOULOS, Georgios. Moving from Traditional to Agile Software Development Methodologies Also on Large, Distributed Projects. *Procedia - Social and Behavioral Sciences* [online]. 2015, 175, 455-463 [cit. 2019-11-27]. DOI: 10.1016/j.sbspro.2015.01.1223. ISSN 18770428. Dostupné z: https://linkinghub.elsevier.com/retrieve/pii/S1877042815012835

17. PHAM, Andrew a Phuong Van PHAM. *Scrum in action: agile software project management and development*. Boston: Course Technology PTR, 2012. ISBN 978-1-4354-5913-7.

18. PRESSMAN, Roger S. *Software engineering: a practitioner's approach*. 7th ed. New York: McGraw-Hill Higher Education, c2010. ISBN 978–0–07–337597–7.

19. PROCHÁZKA, Jaroslav a Cyril KLIMEŠ. *Provozujte IT jinak: agilní a štíhlý provoz, podpora a údržba informačních systémů a IT služeb*. Praha: Grada, 2011. Průvodce (Grada). ISBN 978-80-247-4137-6.

20. Scaled Agile Framework – SAFe for Lean Enterprises. Scaled Agile Framework – SAFe for Lean Enterprises [online]. Copyright © Scaled Agile, Inc. [cit. 04.10.2019]. Dostupné z: https://www.scaledagileframework.com/

21. Scrum.org: Home of Scrum [online]. 2019 [cit. 2019-11-27]. Dostupné z: https://www.scrum.org/

22. SCHWABER, Ken. *Agile project management with scrum: Developer Best Practices*. 2. vydání. Redmond, WA: Microsoft Press, 2015. ISBN 978-073-5696-938.

23. SOMMERVILLE, Ian. Softwarové inženýrství. Brno: Computer Press, 2013. ISBN 978-80-251-3826-7.

24. STELLMAN, Andrew a Jennifer GREENE. *Learning agile*. Sebastopol, CA: O´Reilly, 2015. ISBN 978-1-449-33192-4.

25. SUTHERLAND, Jeffrey Victor. *Scrum: the art of doing twice the work in half the time*. New York: Crown Business, [2014]. ISBN 978-038-5346-450.

26. SUTHERLAND, Jeffrey. Agile Can Scale: Inventing and Reinventing SCRUM in Five Companies. IT Journal [online]. Cutter Information, 12/2001, 12(14), 5-11 [cit. 2019-11-23]. Dostupné z: http://www.controlchaos.com/storage/scrum/articles/Sutherland%20200111%20proof.pdf

27. ŠKODA AUTO a.s. IT-PEP 2.0: *Metodika řízení produktů a projektů*. 2019.

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

29. ŠOCHOVÁ, Zuzana. *Skvělý Scrum Master*. Přeložil Milan DANĚK. Brno: Computer Press, 2018. ISBN 978-80-251-4927-0.

30. VersionOne. (2007). *The State of Agile Developmen*t [online]. Dostupné z: https://www.versionone.com/pdf/2006-state-of-agile-survey.pdf

31. VersionOne. (2008). *2nd Annual Survey 'The State of Agile Development'*[online]. Dostupné z: https://www.versionone.com/pdf/2007-state-of-agile-survey.pdf

32. VersionOne. (2009). *3rd Annual Survey: 2008 The State of Agile Development* [online]. Dostupné z: https://www.versionone.com/pdf/2008-state-of-agile-survey.pdf

33. VersionOne. (2010). *4th Annual State of Agile Survey*: 2009[online]. Dostupné z: https://www.versionone.com/pdf/2009-state-of-agile-survey.pdf

34. VersionOne. (2011). *5th Annual State of Agile Survey*: 2010 [online]. Dostupné z: https://www.versionone.com/pdf/2010-state-of-agile-survey.pdf

35. VersionOne. (2012). *6th Annual State of Agile Survey*: 2011 [online]. Dostupné z: https://www.versionone.com/pdf/2011-state-of-agile-survey.pdf

36. VersionOne. (2013). *7th Annual State of Agile Development Survey* [online]. Dostupné z: https://www.versionone.com/pdf/2012-state-of-agile-survey.pdf

37. VersionOne. (2014). *8th Annual State of Agile Survey* [online]. Dostupné z: https://www.versionone.com/pdf/2013-state-of-agile-survey.pdf

38. VersionOne. (2015). *9th Annual State of Agile Survey* [online]. Dostupné z: https://www.versionone.com/pdf/2014-state-of-agile-survey.pdf

39. VersionOne. (2016). *10th Annual State of Agile Report* [online]. Dostupné z: https://www.versionone.com/pdf/2015-state-of-agile-survey.pdf

40. VersionOne. (2017). *11th Annual State of Agile Report* [online]. Dostupné z: https://www.versionone.com/pdf/2016-state-of-agile-survey.pdf

41. VersionOne. (2018). *12th Annual State of Agile Report* [online]. Dostupné z: https://www.versionone.com/pdf/2017-state-of-agile-survey.pdf

42. VersionOne. (2019). *13th Annual State of Agile Report* [online]. Dostupné z: https://www.stateofagile.com/#ufh-i-521251909-13th-annual-state-of-agile-report

11 Seznam příloh

1) Full Scaled Agile Framework – grafická reprezentace metodického rámce SAFe
2) Komentář odborného konzultanta

Příloha č. 1

Příloha č. 2

Podklad pro zadání diplomové práce