VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA PODNIKATELSKÁ

FACULTY OF BUSINESS AND MANAGEMENT

ÚSTAV INFORMATIKY

INSTITUTE OF INFORMATICS

ZAVEDENÍ VYBRANÉ AGILNÍ METODIKY PŘI ŘÍZENÍ PROJEKTU VÝVOJE SOFTWARU

IMPLEMENTATION OF SELECTED AGILE METHODOLOGY OF SOFTWARE DEVELOPMENT PROJECT MANAGEMENT

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

Sabina Křížová

VEDOUCÍ PRÁCE
SUPERVISOR

doc. Ing. Radek Doskočil, Ph.D., MSc

BRNO 2021

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA PODNIKATELSKÁ

Zadání bakalářské práce

Ústav:	Ústav informatiky
Studentka:	**Sabina Křížová**
Studijní program:	Systémové inženýrství a informatika
Studijní obor:	Manažerská informatika
Vedoucí práce:	**doc. Ing. Radek Doskočil, Ph.D., MSc**
Akademický rok:	2020/21

Ředitel ústavu Vám v souladu se zákonem č. 111/1998 Sb., o vysokých školách ve znění pozdějších předpisů a se Studijním a zkušebním řádem VUT v Brně zadává bakalářskou práci s názvem:

Zavedení vybrané agilní metodiky při řízení projektu vývoje softwaru

Charakteristika problematiky úkolu:

Úvod
Cíle práce, metody a postupy zpracování
Teoretická východiska práce
Analýza současného stavu
Vlastní návrhy řešení, přínos návrhů řešení
Závěr
Seznam použité literatury
Přílohy

Cíle, kterých má být dosaženo:

Hlavním cílem práce je zavedení agilní metodiky projektového řízení do vybrané firmy, která se zabývá vývojem softwaru pro simulátory zbraní.

Základní literární prameny:

DOLEŽAL, J. a kol. Projektový management podle IPMA. 2. aktualiz. a dopl. vyd. Praha: Grada, 2012. ISBN 978-80-247-4275-5.

KORECKÝ, M. a V. TRKOVSKÝ. Management rizik projektů: se zaměřením na projekty v průmyslových podnicích. Praha: Grada, 2011. ISBN 978-80-247-3221-3.

LESTER, A. Project Management, Planning and Control: Managing Engineering, Construction and Manufacturing Projects to PMI, APM and BSI Standards. 6th Edition. Oxford: Butterworth-Heinemann, 2013. ISBN 978-0080983240.

SCHWALBE, K. Řízení projektů v IT. 1. vyd. Brno: Computer Press, 2011. ISBN 978-80-251-2882-4.

YADAV, S. R. a A. K. MALIK. Operations Research. India: Oxford University Press, 2014. ISBN 978--19-809618-4.

Termín odevzdání bakalářské práce je stanoven časovým plánem akademického roku 2020/21

V Brně dne 28.2.2021

L. S.

Mgr. Veronika Novotná, Ph.D. ředitel	doc. Ing. Vojtěch Bartoš, Ph.D. děkan

Abstrakt

Bakalářská práce se zabývá zavedením vybrané agilní metodiky při řízení projektu, přesněji vývojem softwaru pro simulátory ručních střelných a protitankových zbraní.

Cílem této práce je zavedení vybrané agilní metodiky při řízení projektu vývoje software a jeho následný návrh na fiktivním projektu společnosti Saab Czech, s.r.o.

Teoretická část je věnovaná popisu použitých metod a jejich porovnání. Následně budou podrobněji popsány samostatné metody. Další část se bude zabývat analýzou současného stavu společnosti, která využívá tradiční přístup. Východiskem k návrhu samotného vzorového projektu je analýza současného stavu vývoje softwaru a odvětví. Cílem řešení je vypracovaný návrh projektu, který bude mít zavedené vybrané agilní řízení, a tedy bude úspěšné dosažení stanovených cílů bakalářské práce.

Abstract

The Bachelor thesis deals with the introduction of a selected agile methodology in project management, more specifically the development of software for small arms and antitank weapons simulators. The aim of this work is the introduction of selected agile methodology in the management of the software development project and its subsequent design on a fictitious project of Saab Czech, s.r.o.

The theoretical part is devoted to describing the methods used and comparing them. Separate methods will be described in more detail. The next part will be an analysis of the current state of society using the traditional approach. The starting point for the design of the project itself is an analysis of the current state of development of software and industry development. The aim of the solution is to develop a project proposal that will have a selected agile management in place and thus will be successfully achieve the stated objectives of the Bachalor thesis.

Klíčová slova projektové řízení, vývoj softwaru, tradiční přístup, agilní metodiky, SCRUM

Key words project management, software development, traditional project management, agile methodologies, SCRUM

Bibliografická citace

KŘÍŽOVÁ, Sabina. *Zavedení vybrané agilní metodiky při řízení projektu vývoje softwaru* [online]. Brno, 2021 [cit. 2021-05-11].

Dostupné z: https://www.vutbr.cz/studenti/zav-prace/detail/135099. Bakalářská práce. Vysoké učení technické v Brně, Fakulta podnikatelská, Ústav informatiky. Vedoucí práce Radek Doskočil.

Čestné prohlášení

Prohlašuji, že předložená bakalářská práce je původní a zpracovala jsem ji samostatně. Prohlašuji, že citace použitých pramenů je úplná, že jsem ve své práci neporušila autorská práva (ve smyslu Zákona č. 121/2000 Sb., o právu autorském a o právech souvisejících s právem autorským).

V Brně dne 16. května 2021

.....

podpis studenta

Poděkování

Chtěla bych poděkovat panu doc. Ing. Radku Doskočilovi, Ph.D., MSc, za vedení mé bakalářské práce, za jeho cenné připomínky a rady. Také bych chtěla poděkovat mým rodičům za podporu během celého mého studia.

OBSAH

ÚVOD

Metodiky a management jsou součástí každého projektu. V posledních letech se ve firmách z tradičního přístupu řízení přechází na používání agilních metod projektového řízení, jelikož se nejen vývoj technologii posouvá vpřed, ale i metody řízení projektů.

Vývoj technologií na trhu je v posledních letech velmi rychlý, právě proto je třeba neustálý vývoj zlepšovat. To platí i pro poptávku po virtuálních simulátorech, která roste v posledních letech o desítky procent. Tyto simulátory významně pomáhají i armádě s výcvikem.

Bakalářská páce se zabývá zavedením vybrané metodiky agilního řízení v projektech, které zpracovává společnost Saab Czech, s.r.o. Společnost v současné době vyvíjí nejen software pro simulátory ručních střelných a protitankových zbraní, ale i bezpečnostní technologie a systémy.

Práce poskytne teoretická východiska, která popisují problematiku a metody využité pro zavedení agilního řízení u projektu. Dále také analýzu současného stavu vybrané společnosti a jejích požadavků, tvorby projektů a okolí. Výsledky analýz pomohou definovat vhodné realizace projektu v návrhové části.

Výsledkem práce bude nastínění vzorového projektu, který poskytne náhled na řízení a tvorbu projektu s metodikou, která bude vybrána pomocí analýzy současného stavu a porovnávacích tabulek s metodikami, které poskytne teoretická část práce.

1 CÍLE PRÁCE, METODY A POSTUPY ZPRACOVÁNÍ

V této části bakalářské práce budou popsány cíle práce.

Definují se zde metody a techniky, které budou použity pro dosažený daných cílů.

1.1 Cíle práce

Hlavním cílem bakalářské práce je zavedení vybrané agilní metodiky projektového řízení do zvolené firmy, která se zabývá vývojem softwaru pro simulátory ručních střelných a protitankových zbraní. Pro dosažení hlavního cíle práce byly definovány dílčí cíle následovně:

- Zpracování literární rešerše v oblasti současných teoretických poznatků agilních metodik
- Zpracování kritických analýz současného stavu řízení projektů ve vybrané firmě
- Zpracování návrhu zavedení vybrané agilní metodiky řízení projektů ve zvolené firmě

Výstupem z analýz současného stavu budou poznatky, které poslouží jako podklady pro návrhovou část řešení bakalářské práce.

1.2. Metody a postupy zpracování

S ohledem na vytýčený hlavní, resp. dílčí cíle bakalářské práce, budou při zpracování práce využity jak vybrané metody z oblasti obecných metod, tak vybrané metody speciální.

Z oblasti obecných metod, bude využita především metoda analýzy a syntézy, dále metoda indukce a dedukce. Ve fázi sběru relativních dat a informací souvisejících s řešenou problematikou, bude aplikována především metoda řízených rozhovorů, dotazování a pozorování.

Dále, zejména ve fázi návrhové části, bude využito vybraných speciálních metod a technik z oblasti agilních metodik projektového řízení.

2 TEORETICKÁ VÝCHODISKA PRÁCE

V této části bakalářské práce bude popsána teoretická část práce. Bude zde teoreticky popsáno, co je to tradiční přístup řízení a jeho metodiky v porovnání s využíváním metodik agilních.

2.1. Projekt

Je chápán jako soubor činností, úkolů, metodik a přístupů, které mají za úkol dojít k naplnění jedinečného cíle. Bývá vymezen časem, financemi, požadavky na materiál, na tým lidí, kteří jej budou zpracovávat [1]. Měl by být realizován v rámci určité strategie a svou vlastní strategií dosáhnout i cíle projektu. [2].

Projekt pomáhají definovat následující atributy:

- **Jedinečnost**
  - vztahuje se především k cíli projektu, který nám říká, jak originální problém budeme řešit a jak jedinečný přístup bude na konci projektu dodán.
- **Vymezenost**
  - čas, finance, lidské a materiální zdroje vymezují projekt a na základě jejich dostupnosti je stanoven jeho rozsah.
- **Komplexnost**
  - je reprezentována různorodostí metod, které jsou využívány dle potřeb úměrně k životnímu cyklu projektu.
- **Nejistota**
  - provází projekt především při zahájení projektu, mohou z ní plynout rizika nebo příležitosti.
- **Projektový tým**
  - který vzniká v době zahájením projektu a v momentě ukončení je rozpuštěn [1;4].

2.2 Životní cyklus projektu a jeho fáze

Většina, ne-li všechny projekty prochází životním cyklem, který se liší velikostí a složitostí projektu [7]. Životní cyklus projektu lze popsat jako přirozený rámec pro zkoumání vazeb a procesů pro oblast projektového managementu. Je uváděn jako prostředek pro definování začátku a konce projektu a jeho fází. Forma definic životního cyklu projektu se liší podle odvětví, ale i ve stejném odvětví mohou bývat definice různé pro jiné organizace a podniky [6]. Životní cyklus projektu zahrnuje modely jako je vodopádový model, spirálový model, přírůstkový model, prototypování a model RAD (z angl. Rapid Application Development). Tyto typy modelů jsou příkladem životního cyklu, který dokáže přesně rozčlenit rozsah projektu a předem přesně určit jeho harmonogram a náklady [4].

Životní cyklus IT projektu prochází následujícími fázemi:

1. **Proveditelnost:** definice nákladovosti, přínosy, kritéria přijatelnosti, čas, odhady nákladů
2. **Hodnocení:** definice požadavků, výkonnostních kritérií, procesů
3. **Funkčnost:** funkční a provozní požadavky, rozhraní, návrh systému
4. **Autorizace:** schválení povolení, zpevnění postupů
5. **Návrh a realizace detailního návrhu:** systémová integrace, vizualizace, dokumentace
6. **Implementace:** provádění integrační a akceptační zkoušky, instalace, školení
7. **Provoz:** načítání dat, nastavení podpory, předání [7]

Obdobím životního cyklu procházejí kontrolní systémy a fáze rozhodování, ve kterých je pozice projektu přezkoumávána. Rozhraní fází životního cyklu tvoří příhodné milníky, které mohou sloužit jako oznamování pokroků vedení, které se dle toho pak rozhoduje, zda zrušit, či poskytnout další potřebné financování projektu [7].

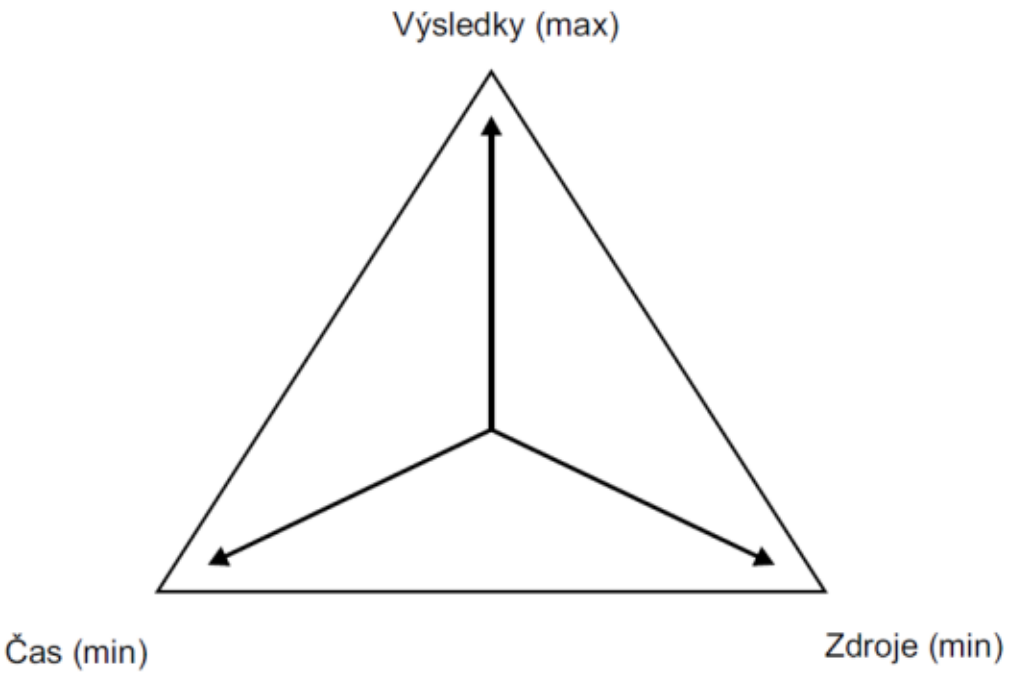
2.3 Trojimperativ

Řešením trojimperativu je nalezení optimálního vztahu mezi specifikací cíle, časovou lhůtou a náklady na konkrétní projekt. Ve spojení s projekty a projektovými cíli pracujeme vždy se třemi základními otázkami, kterými jsou:

- CO?
- KDY?
- ZA KOLIK?

Odpovědí na tyto otázky jsou pomocí tří základních pojmů – cíl, čas a náklady, což souhrnně pojmenováváme trojimperativem projektového řízení. Základním poznatkem je provázanost těchto tří veličin. Pokud dojde ke změně jedné z nich a druhá zůstane stálá, znamená to, že se musí odpovídajícím způsobem změnit i třetí [2].

Pro lepší představu je trojimperativ znázorněn jako trojúhelník (viz obrázek 1) níže.



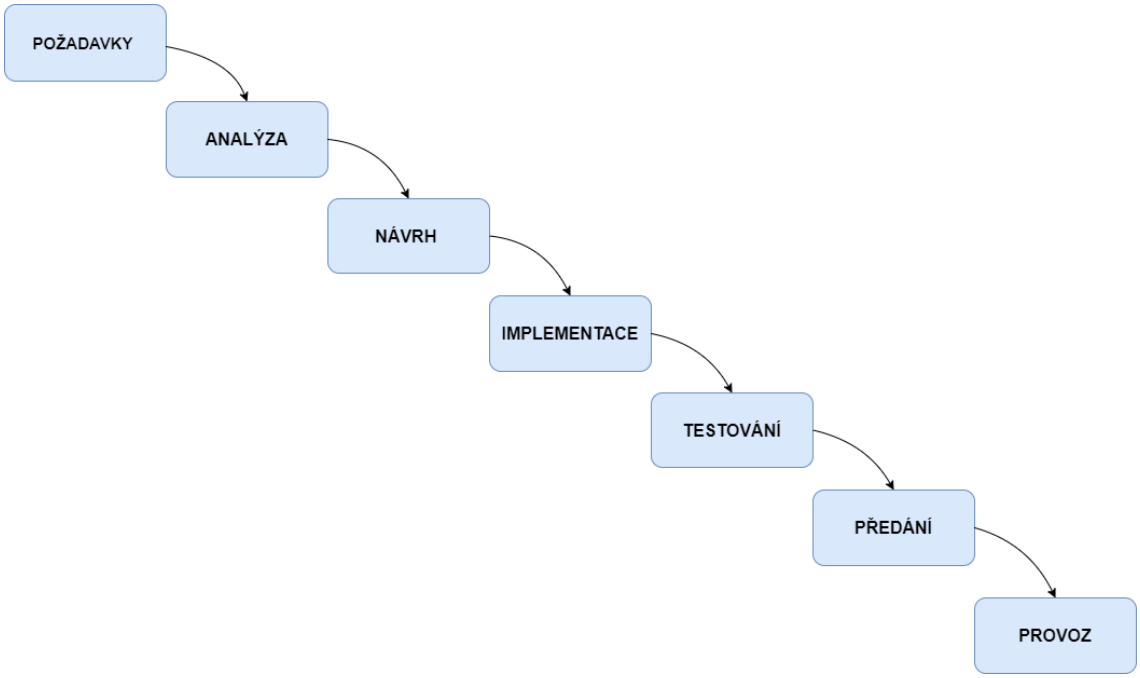
Obrázek 1 - Trojimperativ [2]

2.4 Tradiční metodiky

Tradiční metodiky označujeme jako tradiční, abychom je odlišili od dnes modernějších agilních metodik. Tradiční tedy znamená, že ty to metodiky vznikly dříve. To však neplatí vždy. Existují i moderní, nově vzniklé tradiční metodiky. Tyto metodiky uplatňují především tradiční přístupy k vývoji softwaru. [3]

2.4.1 Vodopádový model

Vodopádový model představuje model životního cyklu vývoje softwarového produktu. Patří mezi nejstarší metodiky, která v dnešní době není již tak moc využívána, jelikož je pro většinu moderních projektů nedostatečný. Znázorňuje určitý stav, postupně na sebe navazujících fází, které nemají zpětnou cyklickou návratnost. Veškeré podstatné fáze, jako jsou definice problémů, specifikace problémů a požadavků, návrh, architektura a nakonec testování, vše se uskutečňuje v předem stanoveném pořadí, bez téměř žádných opakování. [3]



Obrázek 2 - Vodopádový model (vlastní zpracování)

2.4.2 Časová analýza projektu

Časová analýzy projektu blíže popíše druhy metod, které se na tuto analýzu nejčastěji využívají a k čemu přesněji každá slouží.

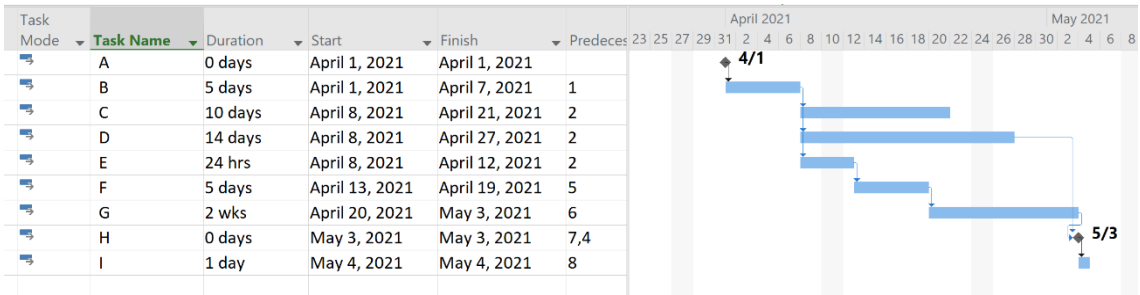
Na tvorbu časové analýzy projektu lze využít následující metody:

- CPM metody
- PERT metody
- Ganttův diagram

Ganttův diagram představuje sled úkolů, včetně vyobrazení jejich začátků a konců. Jednotlivé úkoly jsou nejčastěji organizovány v posloupnosti shora dolů, zatímco časová osa je rozvinuta v linii vodorovné. [1]

Jedná se o velmi účinný nástroj, který má sloužit nejen k plánování v kalendáři, ale také k evidenci jednotlivých aktivit. K těmto účelům se v něm mohou objevit dva i více řádků, které budou představovat plán a skutečnost. Hodnoty v něm mohou být v časových úsečkách, či v počtu jednotek. Při kontrole plnění úkolů, se z plánu harmonogramu zjišťují odchylky. Pokud se u odchylek vyskytne záporná hodnota, rozhoduje se o opatřeních vedoucích k jejich odstranění [3].

Ganttův diagram lze vytvořit i ze síťového grafu a to tak, že do jeho formuláře nejdříve vyneseme ty aktivity, které se nachází na kritické cestě a následně poté ostatní spolu s vyznačením jejich návazností i časových rezerv. Vyobrazení Ganttova diagramu lze vidět na fiktivním projektu X (viz Obrázek 3).



Obrázek 3 - Ganttův diagram projektu X [23]

Metody CPM a PERT jsou využívány ke zlepšení efektivity fungování u větších projektů, které mají předem určené časové a nákladové limity. Tyto metody jsou velmi užitečné zejména proto, že pomáhají při rozhodování o tom, za jakou dobu bude projekt dokončen. Také zároveň poskytuje představu, jak by mohlo vypadat plánování na počátku a konci jednotlivých fází projektu. Dále také vyobrazují kritické fáze, které se v projektu mohou objevit a mohou tak vyžádat pozornost, aby se zabránilo zpoždění při dokončení projektu dle naplánovaného času [8].

2.5 Agilní metodiky a jejich druhy

V této podkapitole si vyzdvihneme nejpoužívanější metodiky, které se využívají při řízení vývoje softwaru. Metody, které vycházejí z agilních metodik je mnoho a neustále se rozšiřují. Mezi nejpoužívanější u řízení vývoje softwaru patří:

- SCRUM
- Extrémní programování XP
- Vývoj řízený vlastnostmi FDD
- Vývoj řízený testy TDD [3]

2.5.1 SCRUM

Tento druh metodiky přiblíží, co SCRUM je, a proč je často využívanou metodikou při vývoji softwaru.

SCRUM používá nejčastěji tým zabývající se vývojem softwaru.

Můžeme jej definovat jako rámec, v němž mohou lidé řešit složité problémy a zároveň produktivně a kreativně dodávat produkty nejvyšší možné kvality. Je to jednoduchý rámec pro efektivní týmovou spolupráci na komplexních produktech [3;19].

2.5.1.1 SCRUM artefakty

Pokud máme vyvinout jakýkoliv software, primární, co potřebujeme znát, jsou požadavky zákazníka. Správa požadavků je samostatná disciplína, kde se setkávají dvě protichůdné potřeby, kterými jsou:

- Pochopitelnost a jednoznačnost pro vývojáře
- Pochopitelnost a jednoznačnost pro zákazníka

Protichůdné jsou především z důvodu, jelikož zákazník rozumí svému oboru, ve kterém podniká, oproti tomu vývojář rozumí svému IT prostředí. Tím pádem se jejich znalosti opačného oboru nebudou potkávat ve stejné kvalitě. Z toho tedy vyplývá, že zásadní věcí bude komunikace, ale i tak bude obtížné požadavky zapsat tak, aby vyhovovaly oběma stranám [3].

User story je tedy doslova uživatelský příběh toho, co by systém měl dělat. Tyto uživatelské příběhy mají tři základní části, kterými jsou:

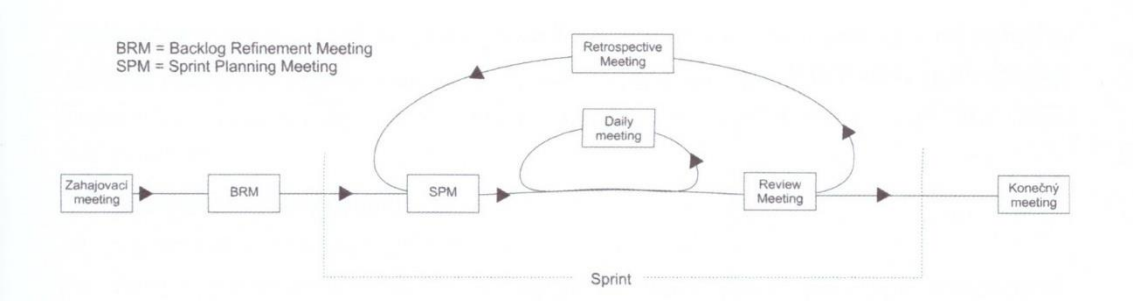
- Definice role
- Definice cíle
- Definice užitku [3]

Uživatelské příběhy mají několik velmi důležitých pozitiv, která mohou sloužit jako shrnutí tohoto nástroje. Jsou to:

- Extrémní stručnost
- Umožňují zákazníkům i vývojářům jednodušeji řešit požadavky
- Potřebují velmi malou údržbu
- Umožňují rozdělení projektů do malých částí
- Usnadňují odhad úsilí potřebného na splnění vývojářských úkolů
- Umožňují udržovat úzký kontakt se zákazníkem [3]

Sprint

Jedná se o iteraci, tedy jeden z mnoha opakujících se cyklů při vývoji software. Cílem každého sprintu, obecně jako u každé iterace, je vytvořit aplikaci, která bude ověřitelná a testovatelná. Každý sprint musí být naplánovaný tak, aby na jeho konci byl spustitelný software. Optimální délka sprintu je cca mezi dvěma až čtyřmi týdny, akceptovatelná pak cca od týdne do šesti týdnů [3]. Vyobrazení sprintu (viz Obrázek 4).



Obrázek 4 - Vyobrazení průběhu sprintu [3]

Backlog

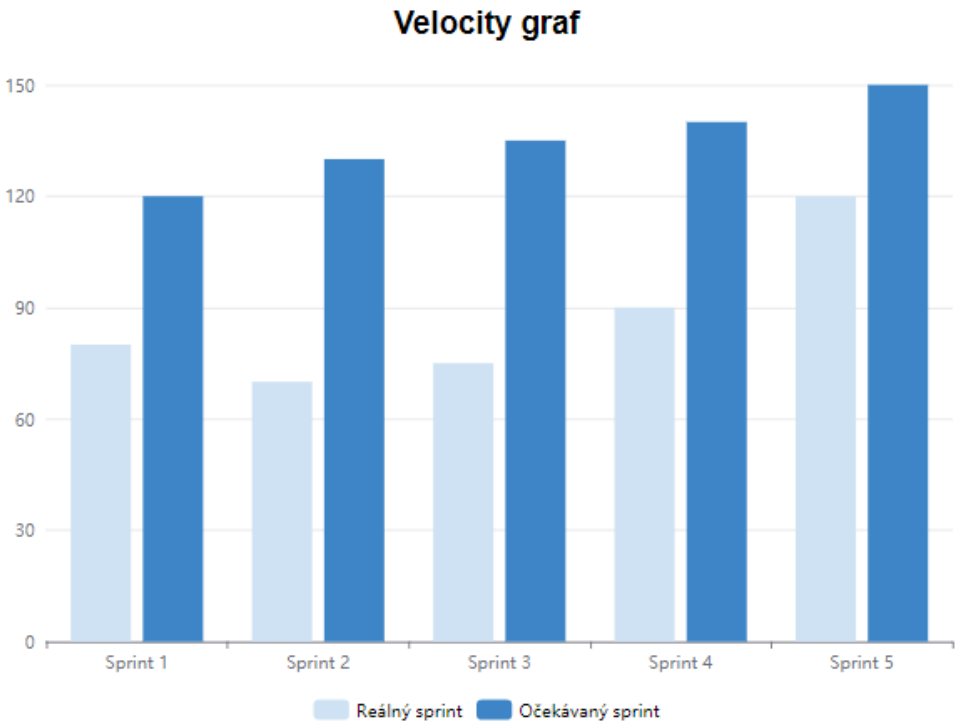
Lze popsat jako seznam nevyřízených úkolů, přesněji řečeno user stories, tedy úkolů, které je nutno splnit a implementovat do systému. Existují přímo dva základny druhy:

Product Backlog – jedná se o kompletní seznam uživatelských příběhů, které je nutno v rámci vývoje softwaru aplikovat.

Sprint Backlog – jedná se o seznam úkolů, které je nutno implementovat v rámci aktuálního sprintu [3].

Velocity

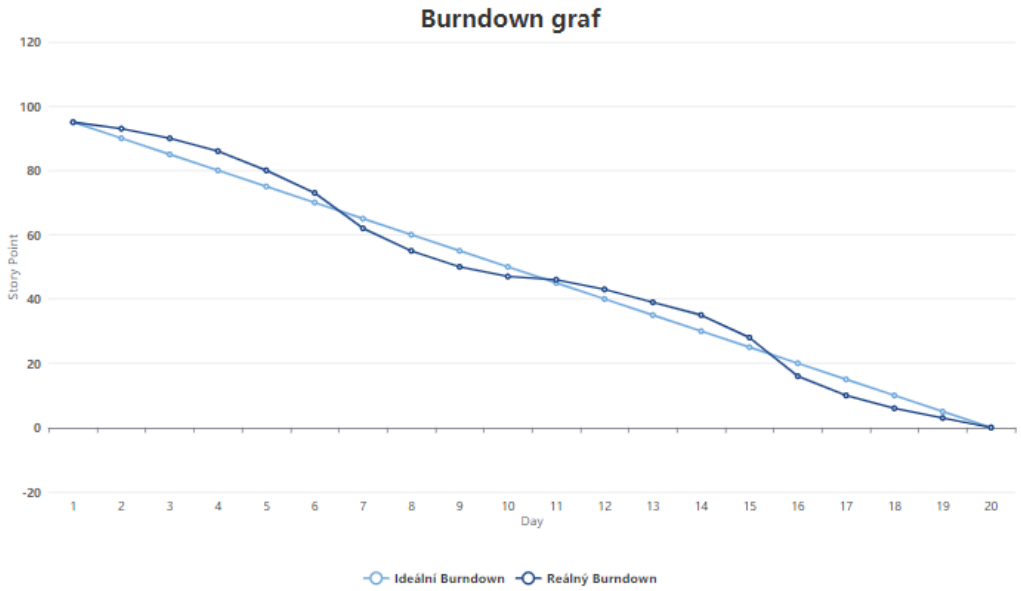
Je měřítkem množství práce, kterou může tým zvládnout během jednoho sprintu a je klíčovou metrikou ve SCRUMu. Rychlost je vypočtena na konci sprintu součtem bodů pro všechny plně dokončené uživatelské příběhy. Měla by být sledována během celého sprintu na sprint burndown grafu a být viditelná pro všechny členy týmu. Velocity je klíčovým mechanismem zpětné vazby pro tým. Pomáhá jim měřit, zda změny procesů, které provádějí, zlepšují jejich produktivitu, nebo jí škodí [17]. Vzor grafu (viz Obrázek 5).



Obrázek 5 - Velocity graf (vlastní zpracování)

Burndown graf

Zobrazuje grafické znázornění toho, jak rychle tým pracuje prostřednictvím zákazníkových user stories, agilního nástroje, který slouží k zachycení popisu funkce z pohledu koncového uživatele. Burndown graf vyobrazuje celkové úsilí oproti množství práce pro každou iteraci. Množství zbývající práce se zobrazuje na svislé ose, zatímco čas, který uplynul od zahájení projektu, je umístěn horizontálně na grafu, který ukazuje minulost a budoucnost. Graf je vyobrazen tak, aby ho každý v týmu viděl a byl pravidelně aktualizován, aby byl přesný [16]. Vzor grafu (viz Obrázek 6).



Obrázek 6 - Burndown graf (vlastní zpracování)

2.5.1.2 SCRUM role

SCRUM Master

SCRUM master není klasický Team Leader, ale pracuje jako mezičlánek mezi týmem a jakýmkoliv rušivým elementem přicházející zvenku. Jeho prioritním cílem je vytvořit samostatný, spolehlivý a efektivní samoorganizovaný tým. Detailnější cíle a povinnosti by se daly shrnout následovně:

- Pomáhá týmu dosáhnout jeho cílů
- Odstraňuje vyskytlé problémy
- Motivuje tým k dosažení lepších výsledků
- Chrání tým před vnějšími vlivy, které by mohly tým odvádět od práce na definovaném cíli

Dále se také stará o to, aby byl SCRUM efektivnější a fungoval. Má na starost jeho dodržování, ale zároveň také i možnost iniciovat změnu, pokud je potřeba. SCRUM Master by měl upřednostňovat koučovací principy, podporovat jednotlivce a tým v jejich rozvoji. Měl by být komunikativní, vnímavý a předcházet případným konfliktům v rámci týmu [5].

Product Owner

Je doslova vlastní produktu. Má na starost definování vize projektu a její transparentní komunikaci týmu, zákazníkům a firmě. Product Owner určuje priority, rozhoduje, na které funkcionalitě se bude pracovat dříve, na které později a na které vůbec. Je zodpovědný za celý Produckt Backlog čili za seznam všeho, o čem se ví, že bude u produktu třeba. Role Product Ownera by neměla být kombinovaná s rolí SCRUM Mastera. Product Owner musí vhodně vyvážit funkce své role, tedy znát dobře zákazníka a být s ním v kontaktu, zároveň být týmu kdykoliv k dispozici. [5]

Vývojový tým

Je samoorganizovaný tým, který se skládá z profesionálů, kteří na konci každého sprintu doručují potencionální hotový produkt. Ten je vždy vytvořen pouze členy vývojářského týmu. Vývojové týmy jsou strukturovány a mají oprávnění se samy organizovat a řídit si vlastní práci. Výsledná součinnost pomáhá optimalizovat celkovou efektivitu práce vývojového týmu. Velikost vývojového týmu by měla být ideálně někde mezi 3–9 členy [11].

2.5.1.3 Druhy schůzek

Zahajovací meeting

Představuje začátek projektu. Zahájení projektu je velmi důležitou etapou projekt, jelikož špatně zahájený projet může znamenat problémy po celou dobu existence projektu.

Úkoly zahajovacího meetingu jsou následovné:

- Vytvoření týmu a vzájemné seznámení
- Seznámení týmu s projektem, předání vizí
- Transformace vizí do předběžné verze Product Backlogu
- Stanovení základních atributů projektu [3]

Standup Meeting

Jedná se o jednu z nejznámějších praktik, která se v agilních metodikách používá. Vývojový tým se setkává na pravidelných schůzkách každý den. Jednotlivci týmu předkládají informace o tom, kdo na čem pracoval předešlý den a jaká bude náplň práce dnes a zdali je někde problém v nějaké dříve prováděné úloze. Jak již napovídá samotné ang. slovo Standup meeting, schůzka se odehrává ve stoje. Samotná schůzka by neměla trvat déle než 15 minut a měla by se odehrávat v místnosti s tabulí, kde by bylo zřetelné, jaké user stories se nachází ještě stále ve sprintu backlogu, na kterých se již začalo pracovat a které jsou již hotové [5].

Sprint Planning Meeting

Jedná se o velmi důležité setkání, které značným způsobem rozhoduje o tom, jaká bude úspěšnost nadcházejícího sprintu. Cílem tohoto setkání je připravit plán sprintu. Product Owner navrhuje a poté schvaluje konečné cíle sprintu. SCRUM Master vede setkání a technicky schvaluje konečné návrhy [3;5].

Sprint Review Meeting

Jedná se o předváděcí setkání, kde se prezentuje výsledek sprintu. Výsledkem sprintu má být funkční a spustitelný program, který si koncový zákazník může řádně vyzkoušet. Sprint Review Meeting by měl být veden v neformální atmosféře a jeho cílem by mělo být úsilí o získání zpětné vazby pomocí odpovědí na dotazy [5].

Sprint Retrospective Meeting

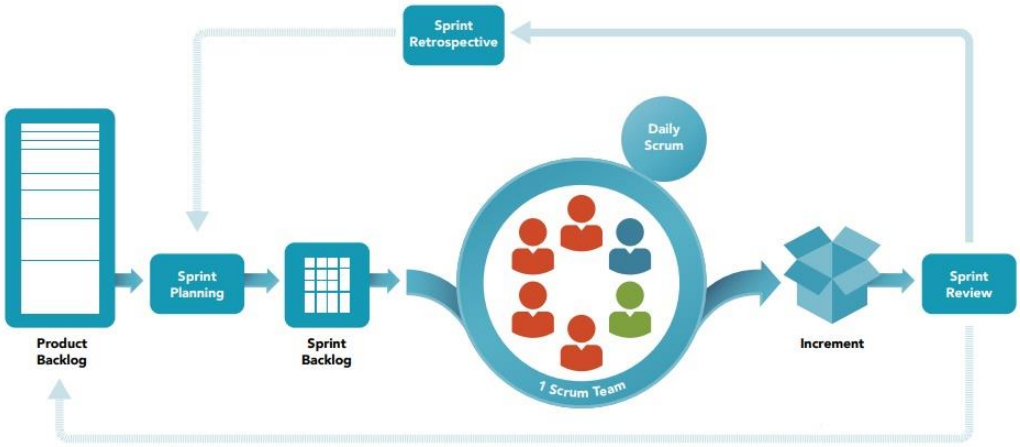
Ve srovnání s review meetingem, kde je cílem prezentování úspěchů zákazníkovi, tento meeting je spíše zaměřen na hledání problémů ohlédnutím se zpět na uplynulé sprinty a definovat si, co je třeba zlepšit. Zlepšením je zde myšleno zejména zlepšování samotných procesů [3].

Konečný Meeting

Jedná se o poslední setkání, které v projektu realizujeme. Toto setkání završuje celý projekt vývoje softwaru. Je tím posledním, co by se v projektu mělo udělat [3].

2.5.1.4 Proces SCRUM

Samotný proces začíná u Product Ownera, který poskytuje vizi pro výrobek a tím vytvoří backlog product. Dalším krokem je setkání s týmem na Sprint Planning Meetingu, kde tým vývojářů vybere user stories, na kterých budou pracovat po dobu sprintu. V tu chvíli vzniká Sprint Backlog. Následně začíná tým pracovat po dobu jednoho až čtyř týdnů. Každých 24 hodin probíhají malé SCRUM meetingy. Na konci každého sprintu prezentuje tým svůj výsledek během Sprint Review. V závěru zhodnotí pomocí retrospektivy, co fungovalo, co nefungovalo a co je třeba do příště změnit. Retrospektiva v závěru má za úkol zlepšit proces pro další sprinty [12].



Obrázek 7 - SCRUM proces [19]

2.6.2 Extrémní programování XP

Extreme programming, neboli extrémní programování vysvětlí, co tato metodika znamená, jaké jsou její výhody oproti jiným metodikám a na čem je založena.

Tato metodika odvozuje svůj název od toho, že dovádí do extrému mnohé principy známé z jiných metodik. Jedná se o metodiku osvědčenou v praxi, která pokud se na správný projekt nasadí správně, může výrazně pomoci k dosažení konečného cíle, jehož cílem není nic jiného, než funkční software [3].

V tradičních metodách pro srovnání, testujeme vždy na konci iterace, v extrémním programování se testuje neustále.

Do extrému je také dovedena spolupráce, ať už se zákazníkem, nebo v rámci vývojového týmu [3].

Změny, ty jsou úplně běžné. Mění se několikrát denně, což znamená i častokrát zahození části kódu, který byl dříve vytvořen [3].

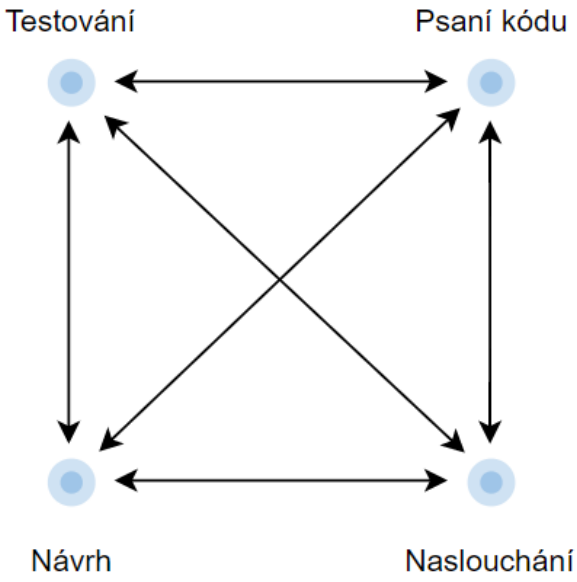
Extrémní programování je založeno na základě čtyř hodnot:

- **Komunikace –** komunikovat musí všichni manažeři, vývojáři a zákazník. Komunikace v extrémním programování je velice důležitá, jelikož zde v podstatě neexistují meetingy.
  V tradičních metodách často existují situace, kdy analytik napíše analýzu a programátor převezme písemný text, aniž by se reálně komunikovalo z očí do očí. To v extrémním programování často není možné, protože žádné velké dokumenty neexistují [3].
- **Jednoduchost –** co není nezbytně nutné, se nedělá. Díky tomu mizí nepotřebná dokumentace, formuláře, ale mizí také i potřeba vytvářet kód, který by mohl výt užitečný [3].
- **Zpětná vazba –** v projektu probíhá neustálé testování a na základě testů jsou upravovány další postupy. Pokud test dopadne špatně, okamžitě je zahájena oprava. I zákazník je součástí testování, i on testuje, zda mu funkcionalita vyhovuje [3].
- **Odvaha –** bez odvahy se v extrémním programování neobejdeme. Musí zde být odvaha a připravenost i na to, že může nastat okamžik, kdy se vyhodí i týdenní práce [3].

Extrémní programování má dále i čtyři základní činnosti, kterými jsou:

1. **Testování** – průběh testování je průběžné, nelze integrovat žádný modul, aniž by uspěl v automatizovaném testu. Test modulu je programován ještě dříve před samotným modulem – po následné implementaci modulu dojde k nasazení testu. Je spuštěn opakovaně v rámci integrace. Testy jsou oddělené (neovlivňují jiné testy) a automatické (poskytují nezávislý, neovlivněný výsledek).

2. **Psaní zdrojového kódu** – Psaní zdrojového kódu začne ve chvíli, kdy jsou napsány a připraveny všechny testy.
3. **Naslouchání** – vývojáři vyslechnout zákazníky i své kolegy, aby věděli, co mají implementovat. Vyslechnutí i komunikace musí být strukturované, v některých případech i lehce řízené.
4. **Navrhování (design)** – návrh představuje způsob, jak se vyhnout cestě do „slepé uličky“. Navrhování organizuje logiku v systému tak, aby změna v jedné části systému nevyžadovala pokaždé změny v ostatních částech [13].



Obrázek 8 - Vzájemný vztah činností XP (vlastní zpracování)

2.6.3 Vývoj řízený testy TDD

Vývoj řízený testy přiblíží, jak tento vývoj probíhá.

Vývoj řízený testy, anglicky také zvaný Test Driven Development, navrhuje psaní testů před samotným kódem a následně je nutné naprogramovat samotný kód. Implementuje se přesně takové množství kódu, jaké dokáže projít testem. Programuje tedy tak, aby byly splněny předem definované testy [3].

2.6.4 Vývoj řízený vlastnostmi FDD

Vývoj řízený vlastnostmi nastíní vlastnosti této metodiky a srovnání s jinými metodikami agilního řízení.

Anglicky také zvaný Feature Driven Development, začíná vytvořením doménového modelu popisujícího celý systém. Tento vývoj má celkově pět fází. První tři jsou sekvenční, další dvě iterativní. Sekvenční fáze mají za cíl zejména vytvořit doménový model. Iterační fáze trvají dva týdny. Během každé iterace se implementují konkrétně užité vlastnosti systému.

Zákazník v průběhu dostává mezivýsledky a nové verze výrobku, je to velmi podobné jako u metody SCRUM.

Na rozdíl od extrémního programování nebo SCRUM, je jednotlivým programátorům práce přidělena. Nevybírají si ji sami [3].

2.7 Agilní vývoj

V této podkapitole bude přiblížen agilní vývoj a manifest agilního vývoje, který pochází z roku 2001 a zahrnuje v bodech, co znamená být agilním [13;15].

Agilní vývoj je o spolupráci a komunikaci s připraveností na možné změny. Slovo agilní, lze říci také jinými slovy jako dynamický, rychlý, přizpůsobivý, iterativní, rychle reagující na změnu. Jedná se o jiný způsob vývoje, který upřednostňuje jiné hodnoty. Jde o výsledky, které nemají striktní procesy. Mohou být kdykoliv změněny a nejsou předem pevně naplánovány [5].

2.7.1 Manifest agilního vývoje

Manifest vychází ze dvou základních tézí, kterými jsou:

1. Akceptovat a poskytnout změnu, je mnohem efektivnější, než se snažit jí zabránit
2. Připravenost reagovat na nepredikovatelné události [13]

Manifest agilního vývoje stojí na 12 principech, kterými jsou:

1. Prioritně vyhovět zákazníkovi včasným a průběžným dodáním hodnotného softwaru
2. Flexibilita ve změnách požadavků, a to i v pozdějších fázích vývoje v projektu
3. Spolupráce pracovníků z oddělení obchodu a vývoje po celou dobu projektu
4. Budování projektů pro motivované jednotlivce vytvářením vhodného prostřední a podpory jejich potřeb
5. Osobní komunikace o informacích z vnějšku i uvnitř vývojového týmu
6. Fungující software jako hlavní měřítko pokroku
7. Sponzoři, uživatelé i členové vývojového týmu by měli být schopni udržet trvalé, stálé tempo práce
8. Pozornost věnovaná technické odlišnosti a dobrému designu neustále zvyšuje agilitu
9. Jednoduchost – umět maximalizovat množství nevykonané práce, je zde klíčová
10. Nejlepší požadavky, návrhy a následné architektury, vznikají ze samo organizujících se týmů
11. Pravidelné zamýšlení se v týmu nad tím, jak být efektivnější, následně poté upravovat přizpůsobovat své chování a návyky [15]

.

2.8 Porovnání metodik Tradiční vs. Agilní

V podkapitole porovnání metod, bude popsáno pár zásadních rozdílů, čím se od sebe metody liší a jaká je jejich efektivnost Zásadní rozdílnost je viditelná v následujících bodech:

- **Flexibilita**

  – Tradiční projektové řízení neposkytuje téměř žádný prostor pro změny u produktu. Je to ustálený proces, který se řídí pouze přístupem shora dolů.

  Jakmile je plán dokončen, manažeři ho sdělí svým týmu a zajistí, aby se ho všichni drželi co nejpřesněji. Existuje zde velký odpor vůči jakékoliv změně, která je navržena, protože může způsobit narušení harmonogramu projektu.

  – Agilní metodika je oproti tomu přizpůsobivější a nabízí velkou flexibilitu, pokud jde o provádění změn ve výrobku. Tato flexibilita umožňuje členům týmu experimentovat a najít tak některé z lepších alternativ. Mohou svobodně sdělit jakoukoliv myšlenku, o které se domnívají, že pomůže produkt dále vylepšit.

- **Vlastnictví a transparentnost**

  – V tradičním řízení patří vlastnictví vedoucímu projektu. Je na manažerovi, aby naplánoval a zdokumentoval celou cestu produktu. Kromě manažerů jsou do fáze plánování zapojeni pouze zákazníci, ale jakmile začne realizace, jejich zapojení je nulové. Vzhledem k tomu, že manažeři drží všechny otěže projektu, členové týmu obvykle nemají možnost mluvit do výsledku svého úsilí nebo do toho, jak projekt pokračuje.

  – Oproti tomu v agilní metodice se členové týmu dělí o vlastnictví projektu. Všichni dají hlavy dohromady, aby přišli s plánem, jak dokončit práci v odhadovaném čase a nákladech. Jsou schopni sledovat svůj vývoj produktu od jeho začátku až do konce. Tato transparentnost hraje důležitou roli při udržování produktivního a vysoce zainteresovaného pracovního prostředí.

- **Řešení problémů**

  – V případě neočekávaných překážek musí jednotlivci v tradičním řízení problém předat svým manažerům k řešení. Oslovovat však svého manažera pokaždé, co problém nastane, není proveditelná možnost. Může způsobit nepřiměřená zpoždění a překročit tak odhadovanou lhůtu a výši celkových nákladů.

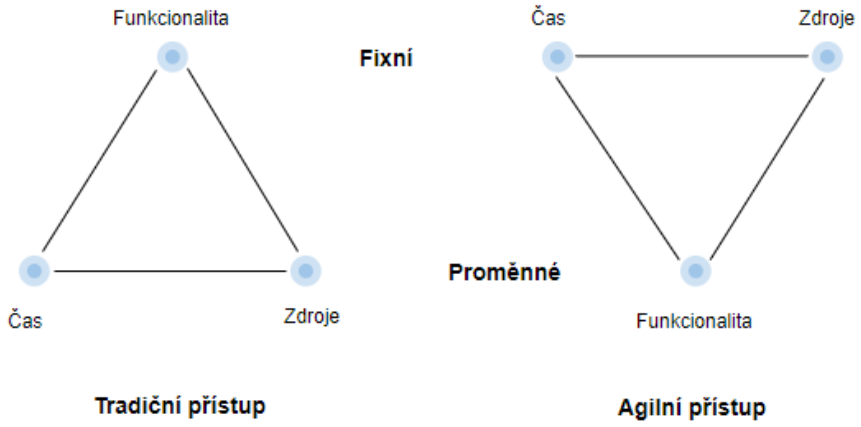
– V agilním týmu mají oproti tradičním pravomoc rozhodovat samy. Snaží se vnitřně vyřešit všechny problémy, aby neztráceli čas. Díky úzkému zapojení do tohoto procesu jim jejich znalosti pomáhají řešit většinou problémů, které brání jejich pokroku.

- **Kontrola a sledování vývoje**

  – Tradiční řízení obhajuje náročné plánován ve fázi analýzy a návrhu projektu. Zaměřují se spíše na zefektivnění procesů než na samotný produkt. Po dokončení procesu se očekává, že tým bude postupovat krok za krokem s minimálním vedením. Průběh je určen po dokončení projektu.

  – Agilní metodika vzhledem k jejím kratším a rychlejším iteracím nabádá členy týmu, aby měli kontrolní stanoviště v pravidelných intervalech. Snadněji tak určí pokrok, stejně jako pomáhá jednotlivcům udržet odpovědnost ve své práci.

  Grafické znázornění rozdílnosti přístupů (viz Obrázek 9).



Obrázek 9 - Rozdílnost mezi přístupy (vlastní zpracování)

3 ANALÝZA SOUČASNÉHO STAVU

Tato část bakalářské práce se zabývá analýzou současného stavu společnosti, která byla pro tuhle práci vybrána, a to společnost Saab Czech, s.r.o. Analýza popíše aktuální stav fungování firmy při tvorbě projektů vývoje softwaru a poskytne tak náhled na nedostatky pro následné zpracování návrhů.

3.1 Charakteristika společnosti

V podkapitole charakteristiky společnosti bude popsána historie a vývoj společnosti až do současného stavu.

Společnost SAAB AB, vznikla v roce 1937 ve švédském Trollhättanu. Společnost se původně zabývala výrobou letounů, kterou poté rozšířila i výroba automobilů a následně i nákladních automobilů.

Společnost Saab Czech, s.r.o. byla založena roku 2007, jakož to obchodní zastoupení společnosti SAAB AB v České republice. Česká pobočka se mimo jiné zaměřuje na projekty pro Ministerstvo obrany ČR, zahrnující mobilní hi-tech radiolokátory a protiletadlové systémy.

Společnost SAAB AB v roce 2011 završila akvizici společnosti E-COM s.r.o. ve Slavkově u Brna, která byla specializovaná především na virtuální simulátory a zřídila zde jedno z klíčových vývojových a výzkumných center své společnosti.

3.2 Organizační struktura společnosti

Podkapitola organizační struktury společnosti popíše, z jakých rolí je společnost složená a jaké mají na sebe vazby. Hierarchie společnosti poskytne také grafické znázornění pomocí obrázku, (viz Obrázek 10).

Společnost má jako nejvyšší řídící orgán ředitele pobočky, jelikož se nejedná o hlavní firmu, ale o pobočku, která má jedno ze zastoupení v ČR.

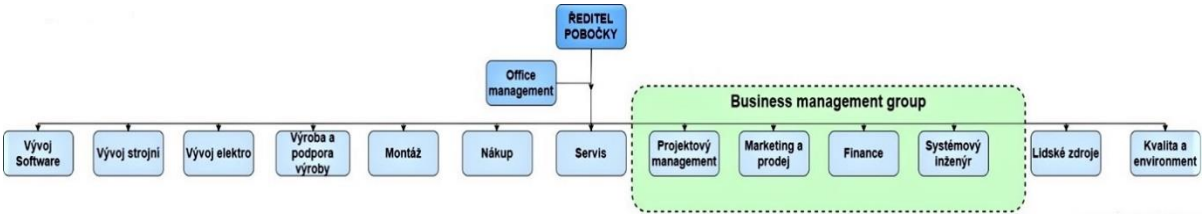
Pod ředitelem pobočky se hned nachází jako samostatný úsek office management, který je přímá spojka mezi ředitelem a ostatními články firmy.

Ostatní články firmy se dělí do dvou větších okruhů a dvou menších. Do každé oblasti spadají procesy sobě blízké. Jedna oblast je vývojová, kde najdeme vývoje softwaru/hardwaru, vývoje strojní, vývoje v elektro sféře, různé výroby a podpory výroby, pak samotné montáže zbraní.

Jeden z menší okruh mezi dvěma hlavními je okruh nákupu a servisu.

Druhý větší okruh je tzv. Business management group, čili v překladu skupina procesů věnujících se obchodním činnostem se zákazníky. Sem spadají zástupci oddělení, jako jsou projektový management, marketing a prodej, finance a systémové inženýrství.

Posledním a druhým menším okruhem je okruh zaměstnanců oddělení lidských zdrojů a působní práce na kvalitu výsledného výrobku a jeho dopad na životní prostředí.



Obrázek 10 - Organizační struktura společnosti (vlastní zpracování)

3.3 Předmět podnikání

Tato podkapitola přiblíží předmět podnikání firmy, a tedy lepší orientaci v tom, na co se zaměřuje.

Společnost se zabývá kromě vývoje, výroby, opravy elektrických přístrojů a technologických zařízení, také skladováním a znehodnocováním bezpečnostního materiálu.

Aktuálně se zaměřují především na vývoj a výrobu výcvikových simulátoru pro ruční střelné a protitankové zbraně.

Zaměřují se také na provádění zahraničního obchodu s vojenským materiálem v rozsahu povolení zákona č. 38/1994 Sb.

3.4 Nástroje

V podkapitole nástroje budou zmíněny aplikace, které společnost využívá jak při tvorbě projektů, tak na samotnou komunikaci ve firmě.

- **Microsoft Project**, **IFS ERP systém** – tyto aplikace slouží na práci projektovým manažerům, kteří v nich tvoří projektové návrhy

- **JIRA** – software project management, slouží k vyhotovení projektu, sledování jednotlivých pracovníků při výkonu plnění úkolů. Eviduje chyby a problémy při vývoji softwaru nebo při řízení projektů. Zajišťuje funkci, kdo na čem pracoval, jakou časovou jednotku a jaké prováděl operace. Pro projektového managera slouží tato aplikace jako evidence, kde se nachází v čase a jakou výši tvoří náklad na projekt.

- **PDM** – neboli řízení produktu. Uchovávají informace o daném produktu, postupech jeho výroby a technologických zdrojích. Ukládají se zde výstupy z milníků. Je určen pro technology, výrobní oddělení a oddělení kvality.

- **Gitlab –** webová aplikace a správce repozitářů. Jedná se o jednoduchý, moderní server. Jeho nástrojem je správa, verzování a vedení projektů a jeho změn. Využívá se především u vývoje softwaru.

Interní komunikace:

- **Mattermost –** slouží firmě na posílání vnitropodnikových zpráv
- **Skype for business –** aplikace pro online porady
- **Web outlook –** webová mailová komunikace
- **Outlook kalendář**

Nástroje, které firma využívá jsou pro ni vyhovující, tedy zde není žádný nedostatek, který by v řízení projektů byl problémem.

3.5 Technologie

Podkapitola technologii uvede základní programy a technologie, které firma využívá pro vývoj softwaru.

Závěrem této podkapitoly bude následovat shrnutí, jestli je zde nějaký program či proces ve vývoji, který je třeba zefektivnit, a tedy jej navrhnout jako problém k řešení v návrhové části.

Společnost využívá následující programovací jazyky a vývojové prostředí:

- Unity 3D
- Visual Studio C++, C#
- HTML5
- JavaScript

Pro vývoj používají následující programy:

- Průběžná integrace
- Automatické testování
- Kódové kontroly
- Systém správy revizí GIT

Veškeré výše zmíněné programy, které vývojové oddělení využívá, jsou pro práci vývojáře dostačující a práce v nich zajišťuje kvalitní stránku vývoje.

3.6. Softwarový tým

Podkapitola softwarového týmu specifikuje odvětví, jaký softwarový tým má a jak se dělí.

Odvětví, které softwarový tým má:

- **Vývoj framework**
  - Aplikace na vizualizaci realtime
  - Aplikace na výpočet fyziky a simulovaného prostředí
  - Aplikace pro počítání a vyhodnocení
  - Learning management systém – systém, který se stará o studenty
- **Integrování s hardwarem a auty**

V závěru softwarových týmů lze říci, že dosavadní využívání vodopádového modelu nevyhovuje již po všech stránkách, a tak by bylo vhodné zvolit efektivnější metodu práce v každém odvětví softwarových teamů.

3.6.1 Velikost softwarového týmu

Velikost softwarového týmu blíže popíše, jak velký tým aktuálně vývojové oddělení má, jaké jsou v něm role a rozdělení specializací.
Velikost jednoho týmu se aktuálně nachází v počtu 10-12 členů, tvořící jeden tým. Počet je odlišný podle specializace týmu. Tvoří dvě rozdělení, a to specializované role a vývojáři.

Specializované role:

- Softwarový Architekt / Teamleader / Vývojář
- Tester 1
- Integrator
- 3D artist

Vývojáři:

- Vývoj backend
- Vývoj frontend

U velikosti softwarového týmu v závěru lze říci, že hlavním problémem je zde velikost jednoho jediného týmu. Má mnoho členů a někteří z nich zastávají i více funkcí najednou, takže zde není taková výkonnost či člověk, který je zaměřený a soustředěný jen na svou práci. Efektivita práce by zde byla třeba rozdělit mezi více pracovníků do více týmů. Obsahovaly by sice méně osob, za to každá by měla svou roli a mohla ji plnit naplno.

3.6.2 Softwarové podpory

Softwarových podpor přiblíží druhy podpor, které společnost využívá pro software a technickou podporu.

Softwarové podpory, které společnost používá, jsou rozděleny do tří částí. Na přímo softwarové a poté na ty, které podporují nástroje pro vývoj softwaru a v poslední řadě ty, které slouží jako podpora v oblastech.

Rozdělení podpor a jejich vlastností, níže:

- **Technická podpora COTS modulů**
  - řeší vnitřní problém u produktu se softwarem, kde technická podpora pomáhá řešit problém např. při vývoji softwaru
  - podpora probíhá zasláním fragmentu kódu -> druhá strana nalezne chybu, tu poté opraví a zašle zpět vývojářům => výrobce software opraví a sdílí tak novou verzi
- **Podpora nástrojů pro vývoj**
  - tato podpora je především využívána pro programy, jako je např.: JIRA nebo Visual Studio
  - komunikace probíhá se stranou podpory pomocí online komunikace, kde se během hovoru pomocí navigace problém řeší nebo jej zástupce podpory vyřeší ze vzdáleného připojení do serveru
- **Podpora v oblastech (outcoursing)**
  - tato podpora se nachází na hranici podpory a školení
  - podpora v oblastech zde znamená pozvání externí firmy, která zaudituje procesy, které jsou ve firmě zavedeny a dají doporučení, co zlepšit a změnit ve směrnici

3.6.3 Spolupráce

Spolupráce mezi oddělením mechaniků a elektrotechniky probíhá v návaznosti na sebe následovným způsobem, kdy zástupce mechanické konstrukce nakreslí konstrukci, z čeho bude konstrukce vyrobena a jak bude vypadat. Zástupce elektrotechniky dodá poté kontrolky a elektroniku do těla repliky střelné zbraně.

Softwarové oddělení je až posledním článkem, které supluje systémové oddělení, kdy ve chvíli, kdy je vyvinuto, vezme se software, který se zaintegruje s hardwarem a začne se testovat.

Projektové oddělení spolupracuje se softwarový oddělením neustále a na vše tak dohlíží. Řeší problémy, které nastanou, a reportuje je top managementu, který je za celý projekt zodpovědný.

Spoluprací je však mnohem více a každá navazuje na každou dle provázaností v projektu. Zde byl vyzdvihnut jen určitý náhled z pohledu vývoje softwaru.

U spolupráce se jeví problémem komunikace podle vodopádového modelu. Vyplývá to z toho, že podle vodopádového modelu je vše jasně dáno, bez možnosti zpětného vrácení se k dané věci, a tedy více diskusí na daném problému. Není tolik flexibilní, jak by spolupráce a komunikace s ní spojená měla být, tudíž potlačuje vývojovou složku.

Návrh zlepšení komunikace a spolupráce bude nastíněno v návrhu řešení, jehož cílem bude tento nedostatek minimalizovat.

3.6.4 Vývojový proces

Vývojový proces probíhá plynule a není nijak časově omezený. Zahrnuje jednotlivé vývojové podprocesy, které již mají dané termíny, do kdy mají být splněny. V těchto podprocesích se také zkouší zbraně a opakovaně testují podle parametrů od zákazníka, aby byly co nejvíce splněny. U testování se jedná především o teplotní rozsahy, co má zbraň vydržet, odolnost pádů z různých výšek a jiné.

Výrobní proces se spouští v momentě, kdy je vývojový proces prototypů a jejich parametrů dokončen a otestován dostatečnými testy. Pokud je vše v pořádku začne tzv. první série výroby.

Po dokončení vývoje a výroby zbraní se vyvinutý software nahraje do hardwaru a simulátory se připravují již na reálný provoz.

Vývojový proces je procesem, který má jasně dané postupy, ať jsou využívány jakékoliv metody řízení projektů. Zde nebyl shledán tím pádem žádný problém, který by změna metodiky v řízení měla nějak ovlivnit k efektivnějšímu výsledku.

3.6.5 Zpřesňování požadavků

Zpřesnění požadavků přiblíží, jak ve firmě funguje komunikace se zákazníkem při zpřesňování požadavků.

Zpřesňování požadavků zákazníka s jeho zpětnou vazbou spočívá v tom, co zákazník říká, co je od něj požadováno a tým na to reaguje nabídkou, jakým způsobem bude výroba probíhat. Předvede, jak by návrhy mohly vypadat a jak by vypadal cílený výrobek. Tým má nastavené nějaké role, avšak často více rolí zastupuje jeden člověk. Z tohoto důvodu by bylo dobré rozdělit role mezi více členů týmu a tím zefektivnit jejich náplň práce. Zde by se jednalo o roli zástupce týmu a zástupce zákazníka pro lepší komunikaci mezi oběma stranami.

3.6.6 Dodávání zakázek

Dodávání zakázek popíše podrobněji průběh od samotného setkání zákazníka se zástupcem firmy až k finálnímu výstupu a dodání zákazníkovi.

Spolupráce se zákazníkem probíhá v následujících krocích:

- Jednání se specialisty zákazníků
- Ověření požadavků SRR
- Ověření řešení se zákazníkem PDR
- Schválení finálního designu CDR
- Otestování finálního designu
- Ověřovací série
- Výroba
- Předání a školení zákazníka

Prvním krokem spolupráce se zákazníkem je jednání se specialisty ze strany zákazníka, aby byly co nejlépe vydefinované požadavky na dodávaný produkt. Druhý krok představuje ověření požadavků procesem SRR. Třetí krok zahrnuje ověření řešení se zákazníkem, jestli je vše tak, jak si zákazník přál, pomocí procesu PDR. Následný krok již představuje schválení a otestování finálního designu. Následuje testovací série výrobku, zdali vše splňuje, jak po stránce parametrů, tak po stránce funkčnosti.

Pokud je vše v pořádku, i se souhlasem ze strany zákazníka, jde produkt do sériové výroby. Posledním krokem u dodávky zákazníkovi, je předání produktu a školení k jeho použití.

3.6.7 Sepsání požadavků

Prvotní komunikaci se zákazníkem provádí obchodní zástupce a poté bid manager, tzv. nabídkář. Sepíší se požadavky od zákazníka, které jsou následně součástí kontraktu. Požadavky se poté analyzují a vytváří se z nich nové požadavky, které naplňují přesnější požadavky zákazníka.

3.6.8 Přehled o vývoji

Přehled o vývoji přiblíží především práci s pomocí aplikace Gitlab, kterou společnost při vývoji využívá.

Softwarový tým využívá aplikaci Gitlab, která jim slouží především pro sledování a správu během celého vývoje softwaru. Dává dohromady programy, které poté dají jeden velký program a následně se tak spustí série testů, které program otestují. Pokud testy projdou, stroj zahlásí, že je otestováno a zašle upozornění správci modulu a ten jen odsouhlasí, že je vše v pořádku. Souhrnně jsou tyto dvě akce nazývány jako automatické testování a průběžná integrace. Aplikace mimo testování také slouží jako kontrola rozdělené práce u vývoje softwaru. Tato aplikace byla zavedena jako náhrada za pracovníka, který veškeré informace měl zpracovávat, a často mohlo dojít k chybám. Díky využívání programu je možné mít lepší přehled a je méně nákladný, proto je to pro firmu výhodnější.

3.7 Úspěšnost projektů

V této podkapitole bude nastíněno, jaká je úspěšnost historických projektů pomocí aktuálních využívaných tradičních metodik.

Pro tuto podkapitolu bylo vybráno 5 projektů, které firma v minulosti realizovala. Jedná se převážně o vetší projekty, které trvaly déle než jeden rok. Zaznamenávají v čase přesnou ukázku toho, zda během doby trvání docházelo k razantním změnám v projektu či nikoliv. Informace o projektech budou vyobrazeny v tabulce (viz Tabulka 1) níže.

Tabulka 1 - Tabulka historických projektů (vlastní zpracování)

	NÁZEV PROJEKTU	DOBA TRVÁNÍ PROJEKTU	ROZPOČET
PROJEKT 1	Projekt FL	36 měsíců (2018-2021)	126 Mio CZK
PROJEKT 2	Projekt ČK	15 měsíců (2018-2020)	17 Mio CZK
PROJEKT 3	Projekt ŠO	22 měsíců (2016-2018)	70 Mio CZK
PROJEKT 4	Projekt JY	20 měsíců (2015-2017)	40 Mio CZK
PROJEKT 5	Projekt SX	36 měsíců (2013-2016)	81 Mio CZK

Pro zlepšení budoucích projektů, které se obecně tvoří déle než 1 rok, je dobré mít zpětné vazby a tzv. Lessons Learned, které mají poukázat na to, co je třeba do příště zlepšit. Tyto zpětné vazby a problémy během průběhu projektu k výše zobrazeným historickým projektům lépe popíše rozsáhlá tabulka (viz Příloha 1).

Dle popisu zpětných vazeb a problémů, s kterými se projektový tým během realizace setkal, je očividné, že se zde objevují problémy v oblastech komunikace se zákazníkem,

či nedostatečné či špatné rozdělení pracovníků v týmu. Tyto nedostatky mohou značně ovlivňovat průběh projektu. Poznatky z těchto historických projektů značí, že je změna v řízení potřebná, aby se nedostatky odstranily a zpětné vazby od zákazníka tak byly více pozitivní s tendencí se k této firmě vracet pro další produkty.

3.9 Shrnutí

Firma agilní metody a prvky zatím nevyužívá, ale plánuje do roka vybrané metody aplikovat především v oblasti softwarového vývoje. Aktuálně společnost používá především metodiky tradiční a samy v některých částech vývoje v projektu vidí tyto postupy jako zbytečně zdlouhavé či neefektivní, které způsobují často časový tlak nebo i zbytečné náklady navíc.

Nedostatky, které byly analýzou zjištěny, poslouží jako cenné body do implementační části práce. Jedná se o oblasti, kde by bylo vhodné je návrhem agilního řízení minimalizovat nebo zefektivnit proces na tolik, aby byly nedostatky zanedbatelné.

Oblasti, kde by změna měla být realizována dle analýzy současného stavu pomocí nastavení vhodné metodiky, jsou:

- Nastavení efektivnějšího počtu členů a jejich funkcí v softwarovém týmu
- Nastavení vhodných meetingů pro týmy a jejich lepší komunikaci o problematice
- Zlepšení sepisování požadavků v návaznosti na zlepšení komunikace

Výše zmíněné oblasti poslouží jako podklad pro následnou implementaci řešení.

4 NÁVRH ŘEŠENÍ A PŘÍNOS ŘEŠENÍ

V této kapitole budou nastíněny návrhy řešení a jejich přínosy, návrhy se budou opírat o teoretická východiska a analýzu současného stavu. Závěrem této kapitoly bude nástin vzorového projektu, který nastíní, jak by práce s vybranou metodikou mohla vypadat v rámci řízení projektu.

4.1 Výběr metodiky

Výběr metodiky vychází z teoretické části, kde jsou vybrané metodiky podrobně popsány. Vybrané agilní metodiky sdílejí dosti podobné hodnoty, avšak každá z odlišného úhlu pohledu na vývoj a danou situaci k posouzení.

Zvolená kritéria pro výběr metodiky vycházejí z analýzy současného stavu a požadavků, které z této analýzy vyplývají.

Součástí výběru metodiky jsou také porovnávací tabulky. Ty díky získaným informacím z teoretické části v návaznosti na analýzu současného stavu porovnají pomocí zvolených kritérií, jaká agilní metoda bude pro návrhovou část nejvhodnější do projektové řízení. To je zde především zaměřeno na část projektu zabývajícího se řízení vývoje softwaru. Veškeré hodnoty a kritéria byla konzultována s vedoucím softwarového týmu společnosti, aby odpovídaly co nejvíce realitě.

Tabulka (viz Tabulka 2) vyobrazuje srovnání metodik, které byly rozebrány v teoretické části a byly vybrány jako ukazatele, které poslouží ke srovnání a následnému vyhodnocení pro návrhovou část.

Tabulka 2 - Porovnání agilních metodik (vlastní zpracování)

	SCRUM	**XP**	**FDD**
NÁROKY NA VLASTNOSTI VÝVOJOVÉHO TÝMU	Zkušený, schopný vývojář a tester v týmu	Alespoň půlka vývojářů by měla být zkušená	Nemusí být zkušený každý vývojář v týmu
POČET ČLENŮ VÝVOJOVÉHO TÝMU	5-9 členů	2 členové	Liší se podle velikosti projektu
DÉLKA ITERACE	1-4 týdny	Dny, není přesně definováno	2 týdny
ZÁKLADNÍ POJMY	Sprinty, Backlogy, Standup meeting	Párové testování, testování kódu	Třídy, vlastník třídy

V tabulce je možné vidět informace o potřebách vývojového týmu, počtu členů týmu, ideální délka iterace a v poslední řadě základní pojmy, které se k daným metodikám váží. Pokud podle srovnání těchto metodik má být vybrána ta, která by pro vývoj softwaru v této společnosti byla vhodná, jedná se zde o SCRUM. Důvod, proč bude vybrána právě tato metodika, vyobrazuje tabulka (viz Tabulka 3).

Tabulka 3 - Vhodnost agilních metodik (vlastní zpracování)

POPIS	SCRUM	XP	FDD
SNADNOST PŘECHODU Z TRADIČNÍ METODIKY NA AGILNÍ METODIKU	1	3	2
ZVLÁDNUTELNOST ROZSÁHLÝCH PROJEKTŮ	2	4	2
ZVLÁDNUTELNOST RIZIKOVÝCH PROJEKTŮ	1	3	3
REAKCE NA ZMĚNY V POŽADAVCÍCH	1	2	3

Tato tabulka byla hodnocena na základě nastavených hodnocení, které bude vyobrazeno níže.

Hodnocení bylo nastaveno pomocí následujícího hodnot:

1 – výborná vhodnost

2 – velmi dobrá vhodnost

3 – dobrá vhodnost

4 – dostačující vhodnost

5 – nedostačující vhodnost.

Tabulka vhodnost metodik byla vytvořena na základě vybraných kritérií, které mají přiblížit a objasnit, proč je SCRUM pro vývoj v této společnosti nejvhodnější.

4.2 Návrh metodiky

Návrh metodiky nastíní návrhy změn pomocí metodiky SCRUM, která pomocí porovnávacích kritérií v tabulkách výše vyšla jako nejvhodnější pro práci v oblasti vývoje softwaru. Bude zde kladen důraz na zavedení kvalitnějšího a lepšího sestavení týmu pomocí SCRUM principů, což povede k zavedení rolí podle SCRUM a tím k lepší komunikaci se zákazníkem. Dále zde bude navrženo zavedení SCRUM Standup porad, které jsou efektivní svou krátkou dobou trvání a jasným sdělením obsahu. Tyto krátké porady tak mohou pomoci při komunikaci v týmu, a také k efektivitě a minimalizování sporů v názoru.

Jak již bylo možné vidět v kapitole analýzy současného stavu u podkapitoly historických projektů, každý projekt je jiný, má svou unikátnost, a tak nelze říci, že postup bude vždy totožný. Navržená metodika proto bude představena na vzorovém projektu, kde při reálném vývoji mohou být jejich dílčí části změněny, zadány jinak nebo vůbec nevytvořeny. Proto bude návrh metodiky představovat pouze jakousi šablonu, ze které poté může firma čerpat i jen určité části, i když byla tvořena co nejreálněji pro prožití pod dohledem vedoucího přímo z oddělení vývoje.

4.2.1 Softwarový tým SCRUMu

Softwarový tým a jeho efektivita práce a výstupy z ní jsou zásadou členů týmu, od kterých se výsledky týmové práce odráží na kvalitě. Návrhem řešení v tomhle případě je stanovení více týmových jednotek, přesněji 3 týmy, které by měly obsahovat 6-8 členů. Každý bude mít nastavenou svou roli, a tak se zabrání tomu, aby jeden člověk vykonával více jak dvě aktivity najednou. Předejde se tak poklesu efektivity a kvality výsledku.

Návrhem rozdělení rolí pro týmy jsou specializované role, kde každý pracovník danou roli zastupuje dle odvětví firmy a je seznámen s její náplní.

Navrhované specializované role:

- Team Leader
- SCRUM Master
- SCRUM Product Owner
- Softwarový architekt

- Vývojář
- Tester
- Integrátor
- 3D artist

Návrh rozdělení rolí oproti analýze současného stavu má především za cíl rozdělit členy týmu tak, aby jejich zaměření bylo soustředěno jen na jejich roli. Měli školení a veškeré vzdělání v rámci pracovní pozice pouze pro svou náplň práce, aby výstupy týmu měly co nejlepší výsledky a zákazník byl spokojený.

Největší změna specializace rolí nastane v rolích Team Leadera a softwarového architekta, který do teď byla jedna osoba v jednom obrovském týmu, proto zde nebyla taková výkonnost pracovníka. Po rozdělení bude Team Leader soustředit lépe svou aktivitu na kvalitnější vedení týmu a softwarový architekt do návrhů softwaru pro simulátory, aby jejich kvalita byla co nejvíce přiblížená realitě. Budou se tak moci věnovat jen práci jim určené. V této změně vidím velký potenciál zlepšení fungování v týmu, kde není mnoho lidí. Je zde daný počet členů podle druhu projektu a každý soustředí svůj potenciál na zadanou práci.

4.2.2 Zavedení SCRUM rolí

Zavedení rolí zde významně přispěje k lepší komunikaci mezi zákazníkem a týmem, který bude mít daný projekt na starost a zamezí se tak nedorozuměním v požadavcích či návrzích ze stran vývojářů a projektantů a nespokojené zpětné vazby ze strany zákazníka.

Team Leader

V nastavení rolí v týmu, který bude mít po návrhu menší množství členů a každý bude mít danou roli, je team leader právě jeden z těch důležitějších rolí. Stará se o chod týmu a je jeho reprezentantem. Zastánce této role má za úkol tým zastupovat při jednání se zákazníkem a také za něj prezentovat návrhy a výsledky práce. Tato role bude především využívána v oblasti zpřesňování požadavků.

Product Owner

Jedná se o velmi důležitou osobu. Role produktového vlastníka spočívá v zastupování zákazníka ve věci zpřesňování požadavků a zadání požadavků firmě. Jako zástupce zákazníka v projektovém týmu zastupuje jeho zájmy. Určuje, co a jak se bude realizovat. Tuhle roli by měl vykonávat pracovník, který zná chod firmy a jednotlivé procesy.

Náplň jeho práce by měla být následovná:

- Získávání a předávání informací a požadavků ze strany zákazníka
- Tvorba backlogů neboli definování úkolů
- Účast na pravidelném Standup meetingu
- Stanovení priority jednotlivých funkcionalit
- Testování vyvinutých funkcionalit
- Diskutování navrhovaných řešení s týmem

SCRUM Master

SCRUM Master není Team Leadrem v klasickém slova smyslu, pracuje jako mezičlánek mezi týmem a vnějším okolím. Aktuálně má k této roli ve firmě nejblíže vedoucí oddělení vývoje. Ve firmě se aktuálně nachází jako Head of Development, který má na starost veškeré agendy spojené s vedením týmu, strategické porady, organizace a zadávání úkolů a jejich vyhodnocení. Kromě vedení týmu je také ještě i softwarovým architektem. Cílem této role je, aby i nadále vedl tým, podporoval ho i jednotlivce v jejich rozvoji a také určoval požadavky, které jsou potom součástí implementace v projektu. Změna role také bude znamenat, že se pracovník na tomto postu bude věnovat pouze náplni SCRUM Mastera a nebude mít na starost 3 role, které při jednom člověku nefungují, a změna je žádaná.

Náplní jeho práce by mělo být:

- Pomáhat týmu dosáhnout jeho cílů
- Odstraňovat problémy
- Motivovat tým k lepším výsledkům
- Chránit tým před vnějšími vlivy, které by mohly odvádět od soustředěné práce na definované cíle

Product Owner a SCRUM Master jsou role, které mají vzájemné propojení při spolupráci tak je jejich spolupráce důležitá a měla by mít jasně nastavená pravidla.

4.2.3 Zavedení Standup meetingu

Spolupráce a s ní spojená komunikace je v týmu velmi důležitá. Zavedení Standup meetingů má právě danou spolupráci zkvalitnit a zlepšit. Do teď byla realizována pouze strategická porada, která byla jasně daná a dopředu „nalinkovaná“ bez možnosti změn za pochodu. Zavedení SCRUM meetingů má přinést největší změnu ve větší flexibilitě schůzek, které mohou být denní, v dané časy. Další změnu, které zavedení Standup meetingů má přinést je otevřenost pro více názorů ostatních členů týmu, a tak se lépe a více zapojit do řešené problematiky.

4.2.3.1 Pravidla pro zavedený Standup meeting

Po zavedení Standup meetingů je třeba si určit i pravidla, která by tento druh schůzek udělala lépe realizovatelný a kvalitnější.

Po zavedení změn budou schůzky realizovány denně v určitý čas. Začátek schůzky bude nastaven na 11:45. Tento čas bude nastaven právě z důvodu efektivnosti. Jelikož setkání bude na 15 minut, tým poté bude moci chodit rovnou na oběd, který mají ve 12 hodin. Čas před obědem je stanoven proto, jelikož každá porada vyžaduje vymezený čas na setkání, které znamená přestávku od dané práce a ta díky tomuto postupu nebude přerušována nějak zásadně.

Setkání před 12 hodinou tak bude znamenat hladký průběh schůze, poté obědovou pauzu a po obědové pauze plynulé pokračování pracovních povinností. Výhodou před

obědového času je také to, že se pracovníkům blíží pauza, tak bude projednávání problematiky stručné a výstižné.

Schůze by se měla odehrávat v místnosti tomu určené pro změnu prostředí a naladění se na diskutování. Není dobré schůzky provádět i z hlediska efektivnosti v místě kanceláře, kde pracovník celý den pracuje. Meetingová místnost by tak neměla být příliš daleko, aby přesun trval minimum času a pracovník neztratil motivaci do jednání zdlouhavým přesunem z místa na místo.

Nastavení připomínek na pravidelná setkání bude formou informační online tabule interního systému, kde je možná změna nebo úprava za chodu.

4.2 Vzorový projekt

V této podkapitole bude nastíněn vzorový projekt pomocí vybrané metodiky a náležitosti s projektem spojené.

Použití vybraných metodik, které byly posuzovány v tabulkách (viz Tabulka 2 a 3) a rozepsány v teoretické části práce ukázaly, která technika by vyšla jako vhodný kandidát na zavedení agilního řízení. Jelikož se zde jedná o práci především s vývojem softwaru a s ním spojeným vývojovým týmem, nejvhodnější dle více kritérii vyšla metoda SCRUM. Tato metoda bude níže ukázána na vzorovém projektu jako ukázka, jak by projekt pomocí této metodiky mohl vypadat.

Jelikož se projekt neobejde bez zadavatele a firmy, která zadání bude řešit, je třeba si jako první určit základní role zadavatele, firmu, která bude zadání realizovat čili realizátora a současný stav.

Zadavatel – zadavatele zde tvoří zákazník, kterým je zástupce armádních výcviků v zemích po celém světě

Realizátor – realizátorem je zde firma Saab Czech, s.r.o., která se zaměřuje na vývoj softwaru do simulátorů ručních střelných a protitankových zbraní. Tato firma je jako realizátor vybrána z důvodu dlouholeté zkušenosti s touhle výrobou.

Současný stav – v současném stavu má firma projektové řízení realizováno tradičním přístupem, který v určitých částech realizace již zcela nevyhovují tak jak by si firma představovala. Nechtějí však zatím přejít na stoprocentní agilní přístup, jelikož v určitých oblastech firmě tradiční především vodopádový model vyhovuje.

4.2.1 Řešení projektu podle SCRUMu

Jelikož dle tabulek porovnání metod vyšel nejvíce vhodný SCRUM, bude návrh řešení projektu ukázán na této metodě.

Tato metodika je založena především na kladení důrazu na řízení procesů za pochodu vývoje. Aby toho dosahovala má stanovená určitá pravidla než nějaké konkrétní přístupy. Jelikož metodika začíná svou činnost sběrem požadavků od zákazníka a její činnost končí u vytvořeného spustitelného a funkčního softwaru, bude na vzorovém projektu SCRUM ukázán na této části vývoje.

4.2.2 Vývojový tým

Složení vývojového SCRUMového týmu by mělo mít přítomnost pracovníka, který by zde zastával roli zákazníka a zastupoval jeho názory a připomínky. Pracovník této role by měl mít dobrý přehled o aktuálních procesech ve firmě a také zároveň zná a ví, co zadavatel od vývoje očekává. Následující roli, kterou zde bude pracovník ze SCRUMu zastávat je vlastník produktu, jinak také zvaný Product Owner. Jeho náplní je zastupovat zájmy zákazníka. Měl by mít schopnosti jako je především dobrá komunikace se zákazníkem a porozumění, dále také analytické myšlení a chápání zákazníka a jeho potřeb vůči tržní a konkurenční situaci. Tuto roli by v týmu měl zastupovat pracovník, který má zkušenosti z předešlých projektů i v roli samotného analytika. Následnou SCRUM rolí, která by se v týmu měla objevit je SCRUM Master. Je to velice specifická role, jejíž náplní je především odstraňovat překážky, které mohou z vnějšku přijít. Zajišťovat týmu efektivnější práci a také týmu jako role pomáhat. Jelikož se občas u překážek jedná i o mezilidské spory, tuto roli by měl zastávat člověk, který někdy dříve tým vedl jako vedoucí. Poslední rolí je zde tým jako takový. Tým by měl být schopný

sebeorganizace a efektivnosti. Tým by se po rozdělení rolí měl nacházet kompletní, a tak může přejít k samotnému vývoji.

4.2.3 Vývojový proces

Vše začíná setkáním zákazníka s Product Ownerem, neboli vlastníkem produktu. Jejich první setkání je zaměřeno na vytvoření požadavků, které říkají, co zákazník od systému očekává a vše se sepíše. Celý tento proces je evidován do systému podniku, aby bylo možné se k tomu kdykoliv vrátit a cokoliv doplnit. Sepsání požadavků by mělo mít formu, která je srozumitelná i pro zákazníka a umí se v nich zorientovat. Poté, co je vše sepsáno a odsouhlaseno oběma stranami, Product Owner vytvoří z tohoto seznamu produkt backlog. Požadavky v backlogu by neměly být již měněny a měla by se jim přiřadit požadovaná priorita podle toho, o co v nich jde. Zde je prioritou vytvoření funkčního softwaru s co nejvyšší kvalitou a včasnou dodávkou. Backlogy, jejich priority a následné sprinty, které z nich budou tvořeny, budou představeny na vzorovém projektu v programu JIRA.

4.2.4 Příprava před zahájením tvorby projektu

Než bude vytvořen samotný projekt a jeho postup ve vzorovém programu, je třeba si určit pár zásadních fází, které budou předcházet samotné tvorbě v programu. V programu se jedná poté už jen o vizuální představu průběhu projektu. Dále, jak se vkládají informace a údaje pro tvorbu úkolů, pro možnost upozornění či grafické znázornění, kdy, v jakém časovém období a jaká práce má být vykonána, aby se výsledek přibližoval k nastavenému cíli.

Fáze by měly být rozděleny následovně:

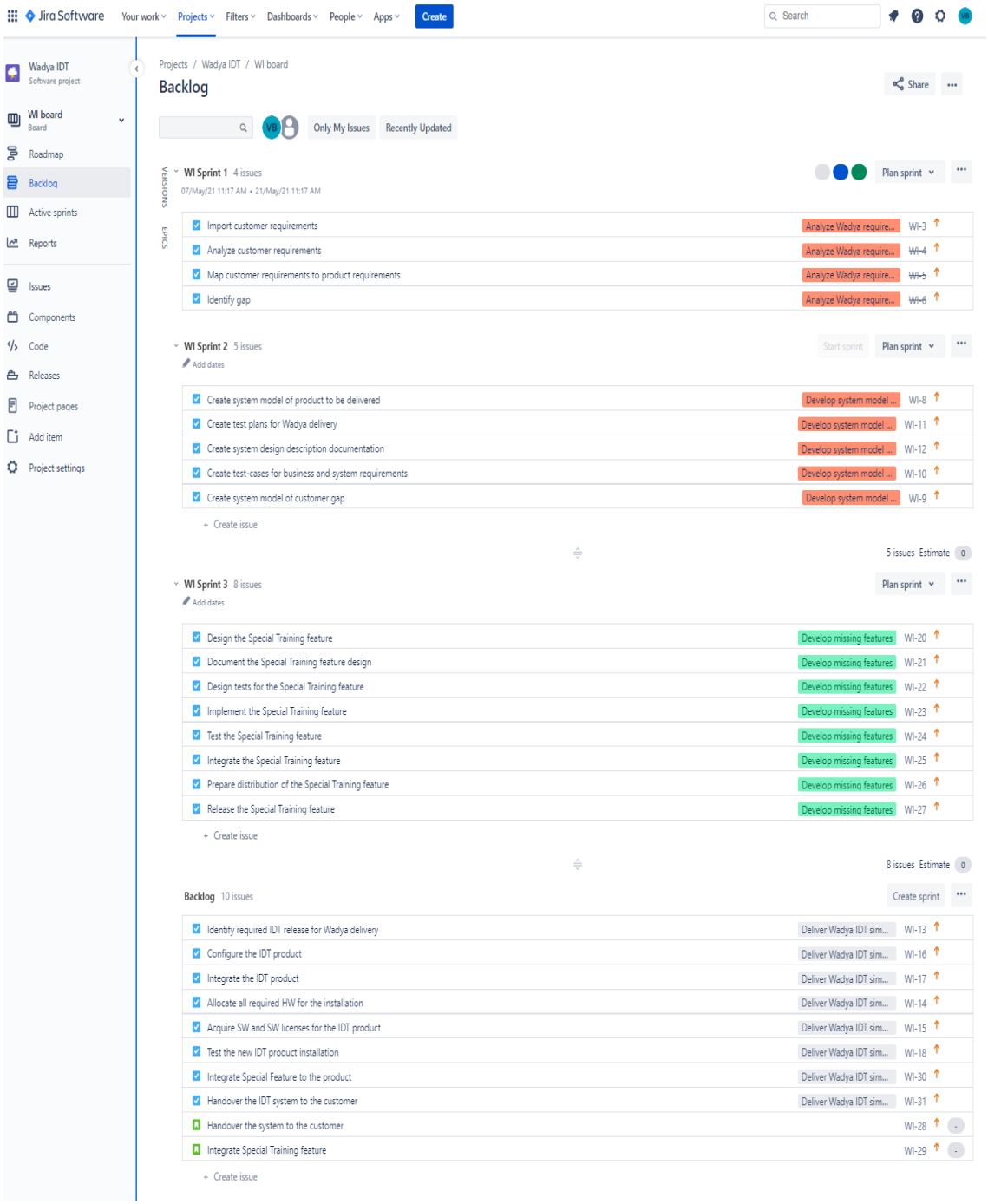
- Návrhová fáze – kde se tým sejde na společné poradě a projedná předložené návrhy a s nimi spojené požadavky, které byly získány od zákazníka. Tým vytvoří možnosti, ze kterých poté vyjde jedna alternativní a odhlasují si tak směr, jakým projekt bude veden.
- Implementační fáze – zde se již vývojáři zaměří na návrhy a design prototypů produktu. Tyto návrhy konzultují poté s celým týmem. Pokud dojde ke schválení

návrhu, mohou být poté naprogramovány příslušné procedury, které dávají požadované výstupy, jaké jsou zadány v návrhovém prototypu. Poté zdali je vše v pořádku, ověří se jejich integrita pomocí řady testů.

- Testovací fáze – kde je produkt testován pomocí různých testů, které slouží k detekci nedostatků či chyb v kódu a jeho návaznosti na funkčnost.
- Ověřovací fáze – zde se vývojový tým setká se zákazníkem, kterému představí výrobek z ověřovací série výroby s nahraným požadovaným softwarem. Zákazník je s produktem seznámen, jak produkt použít a poté si jeho funkčnost sám vyzkouší. Pokud se zde objeví nějaké požadavky na změny nebo přídavky, tým tyto požadavky sepíše a poté vrátí zpět implementační fázi na úpravu.
- Předávací a školící fáze – nastává, pokud je zákazník spokojen i po splněných dodatkových úpravách. Je mu předána konečná série výrobků a dáno školení, jak jej používat.

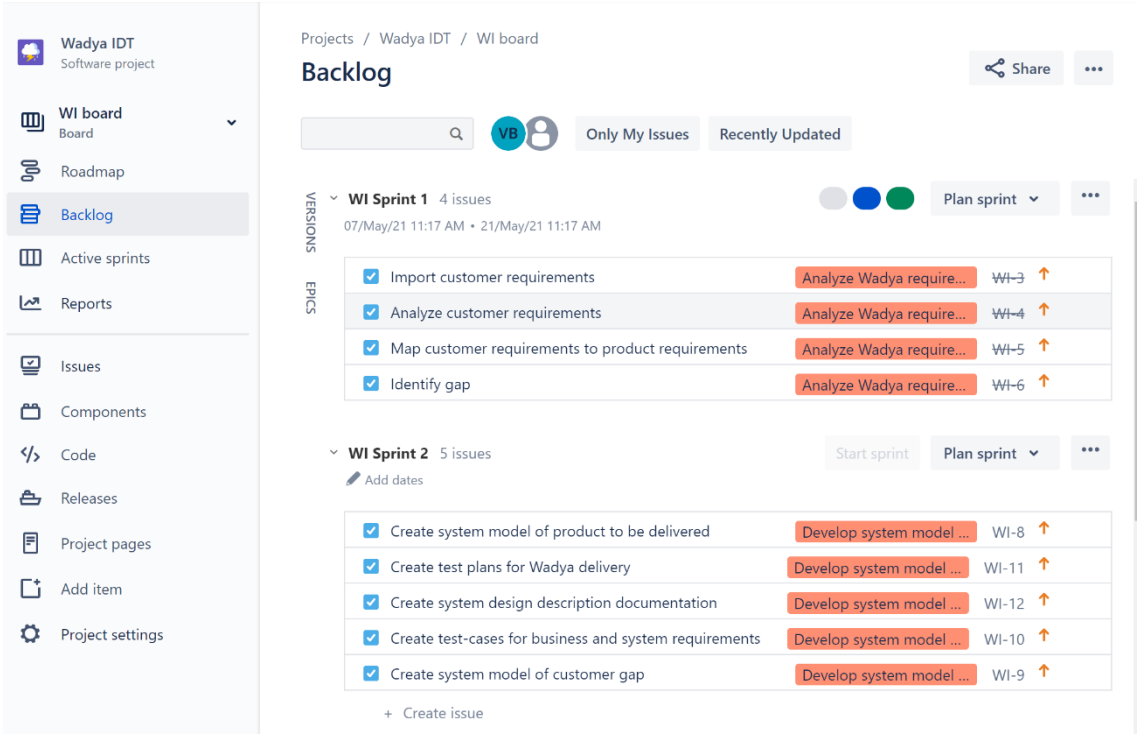
4.2.5 Tvorba vzorového projektu v programu JIRA

Na začátku tvorby projektu v programu JIRA, je třeba nejprve vytvořit tzv. issues neboli úkoly či problémy k řešení. U každého issue se vyplní jeho název a popis, co má daný úkol nebo problém obnášet. Každé vytvořené issue dostane přidělený vygenerovaný identifikátor v podobě písmene a čísla a vytvoří se tak seznam, kterým se naplní backlog. Poté co je backlog naplněný potřebným množstvím issues, je možné tyto úkoly rozdělit do sprintů. Zde se bude jednat o čtyři po sobě jdoucí sprinty. Úkoly z backlogu se vybírají dle priorit, které mají být v prvním, a které například až ve čtvrtém spritnu a jsou do nich vloženy ze seznamu v backlogu. Každý přidělený úkol z backlogu má ve svém vytvořeném issue zadaný i předpokládaný čas, jak dlouho by práce měla trvat. Tyto hodnoty jsou v řádech 2 týdnů. Veškeré podrobnější informace pro lepší představu (viz Obrázek 11).

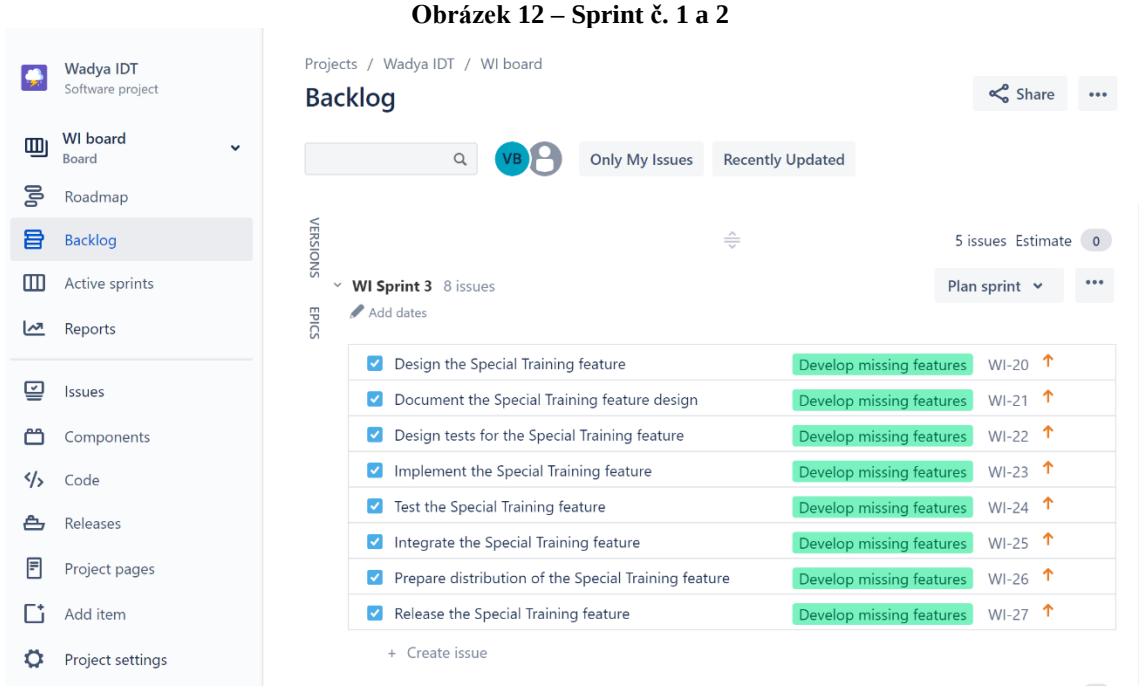


Obrázek 11 - Backlog s naplánovanými sprinty

Obrázky níže (viz Obrázek 12 a 13) vyobrazují naplněné sprinty, které se již realizují. Splnění každého sprintu se vyobrazí jako přeškrtnutý identifikátor přiřazeného úkolu sprintu, aby bylo lépe rozeznatelné, který je již hotový a který ne. Identifikační čísla sprintů jsou také zpětně dohledatelná, kdyby někde během průběhu projektu nastala chyba.

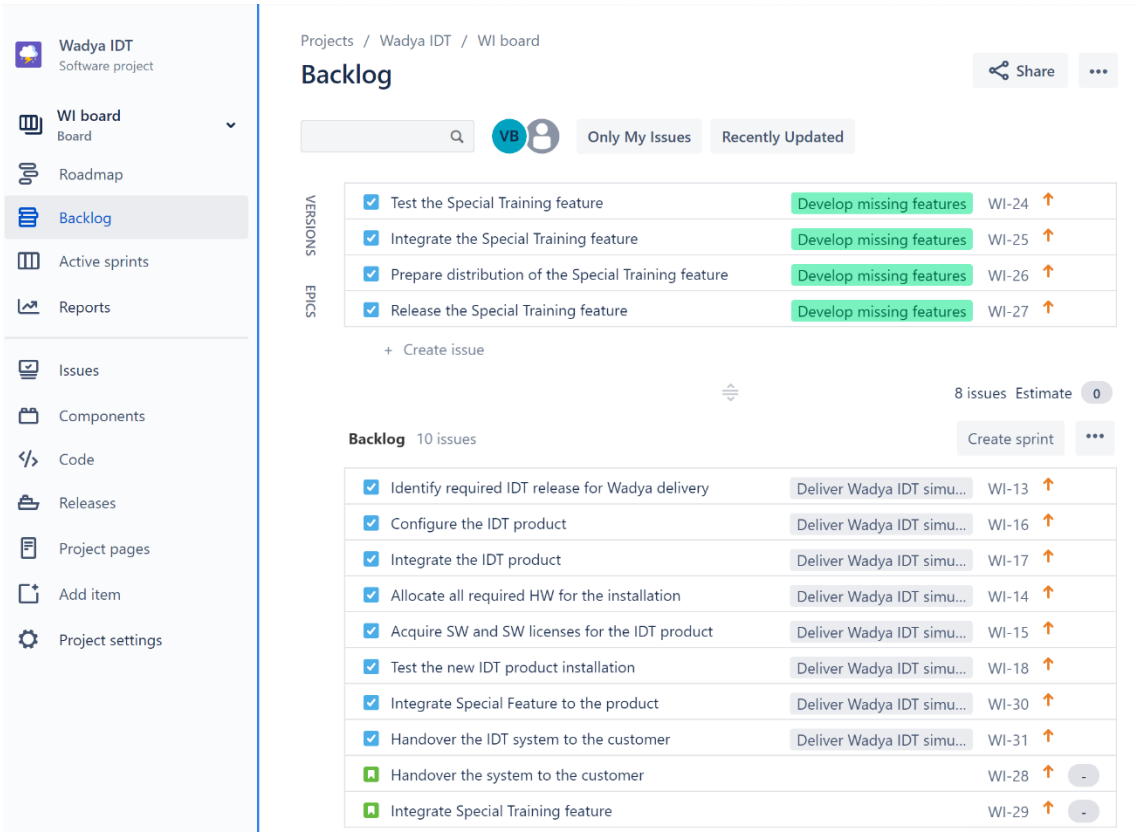


Obrázek 12 – Sprint č. 1 a 2



Obrázek 13 - Sprint 3

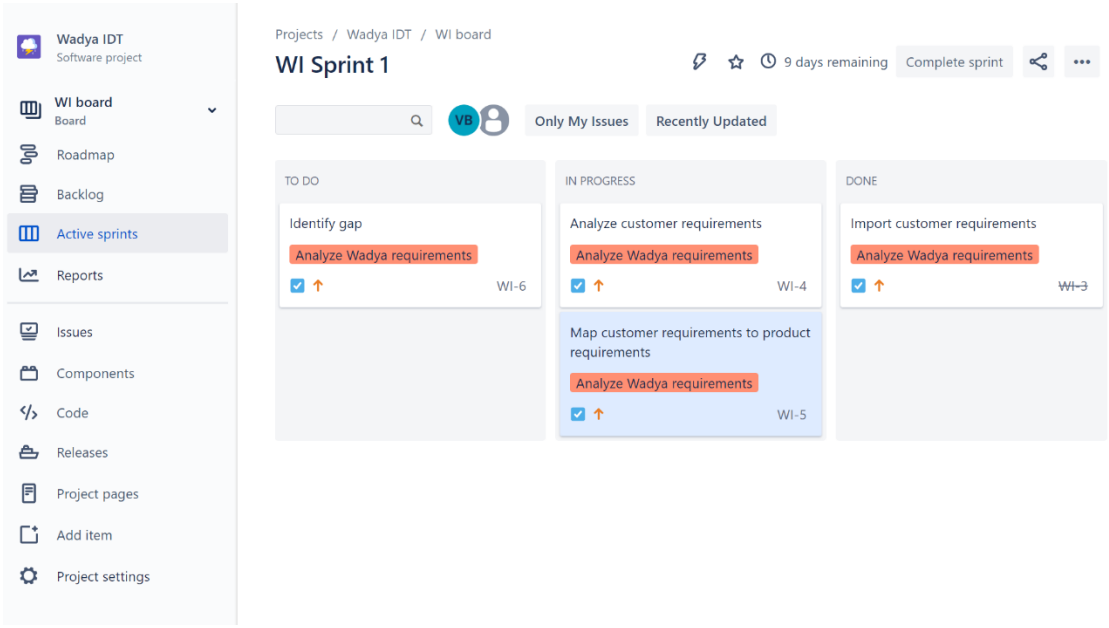
Issues, které po naplnění sprintů zůstanou nepřiřazené, setrvávají poté stále v backlogu, jelikož byly prioritně naplněny zatím pouze tři sprinty. Čtvrtý sprint bude doplněn jako poslední, jelikož zde se již řeší testování a integrace systémů čili poslední fáze vývoje. Pro představu, jak takový nachystaný backlog seznam vypadá, přiblíží návrh (viz Obrázek 14) níže.



Obrázek 14 - Nepřiřazené issues

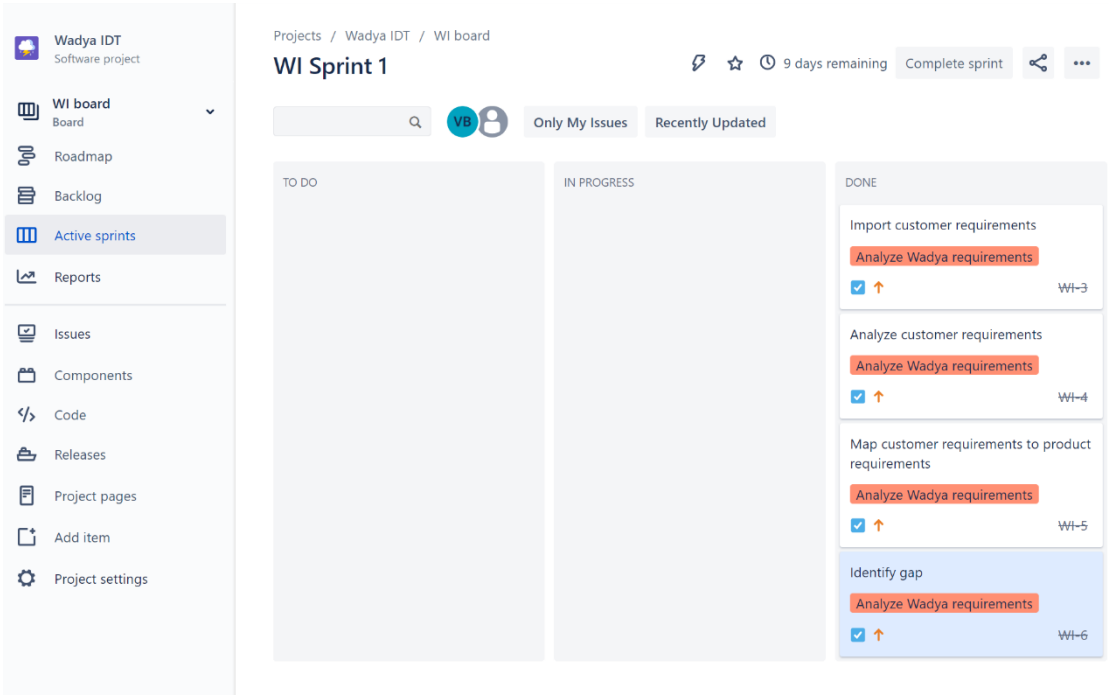
Průběh sprintu je u všech téměř totožný, proto bude blíže popsán průběh sprintu prvního, kde bude vyobrazen i jak vypadá sprint dokončený.

Vše začíná naplněním kolonek, které představují fázi TO DO neboli seznam úkolů, které je třeba řešit. Poté fázi IN PROGRESS, což značí, že tento úkol je již v realizaci. Poslední fáze je seznam DONE neboli seznam hotových úkolů, kde vznikne již výše zmíněné přeškrtnutí identifikátoru na znamení, že je tato úloha již hotová. Pro lepší vizuální představu poslouží obrázek níže (viz Obrázek 15), který tento průběh srozumitelně zobrazuje.

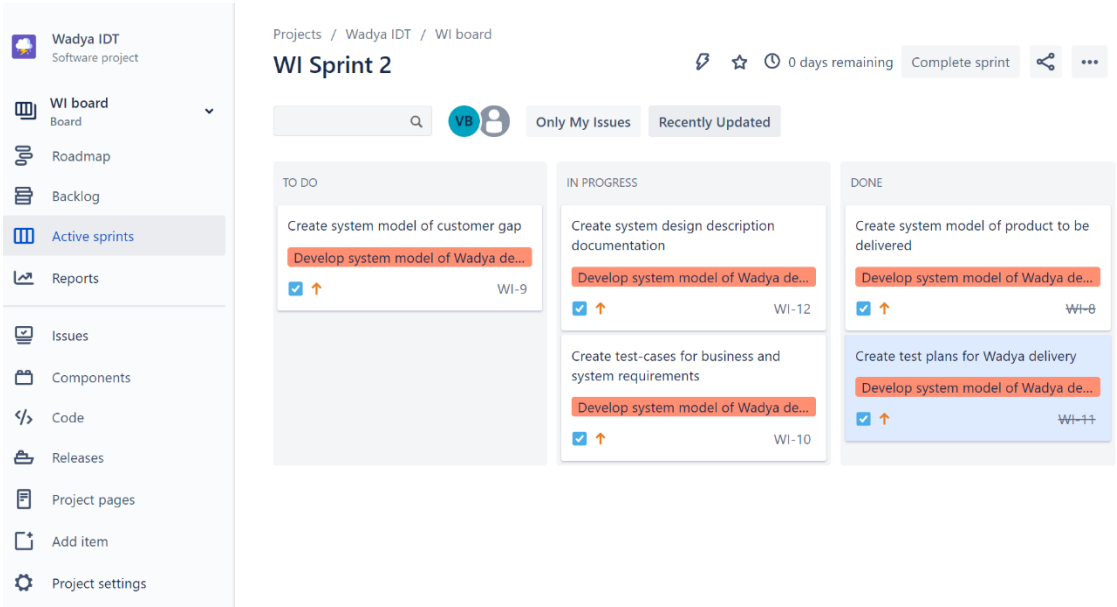


Obrázek 15 - Průběh sprintu č. 1

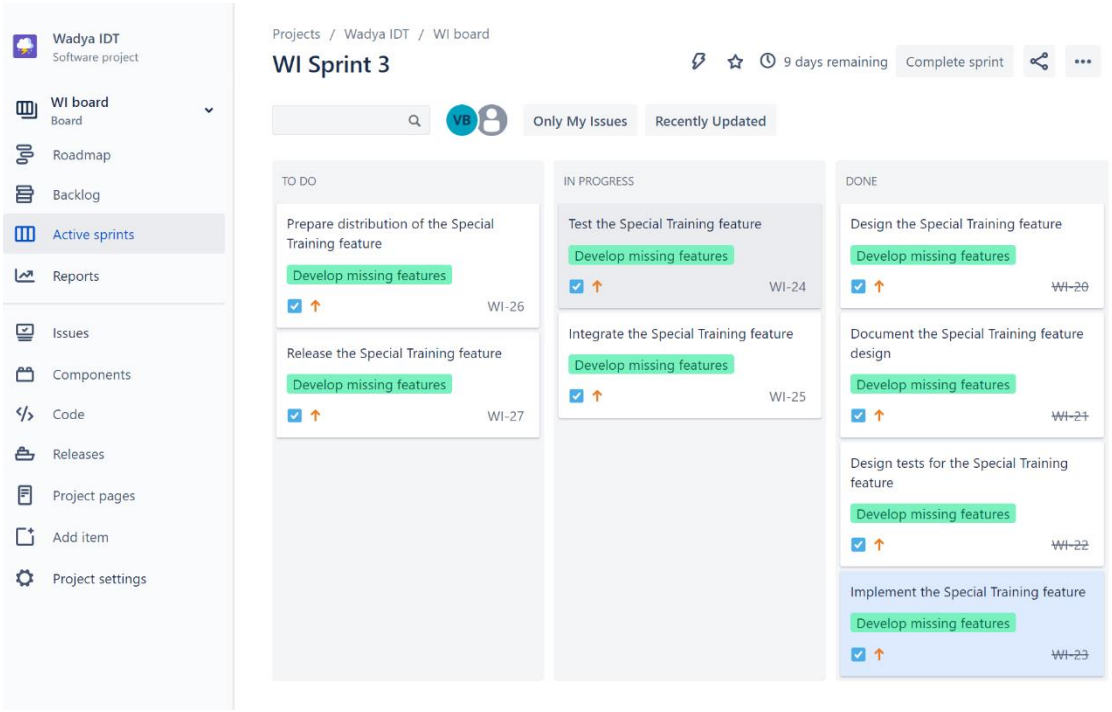
Dokončený sprint se vyobrazuje jako seznam hotových úkolů v kolonce DONE s přeškrtnutými identifikátory na znamení, že jsou již hotové. Díky těmto identifikátorům je poté možné je dohledat v hotových sprintech pod danou značkou, která již není pro práci s ním aktivní. (viz Obrázek 16).



Obrázek 16 - Dokončení sprintu č. 1

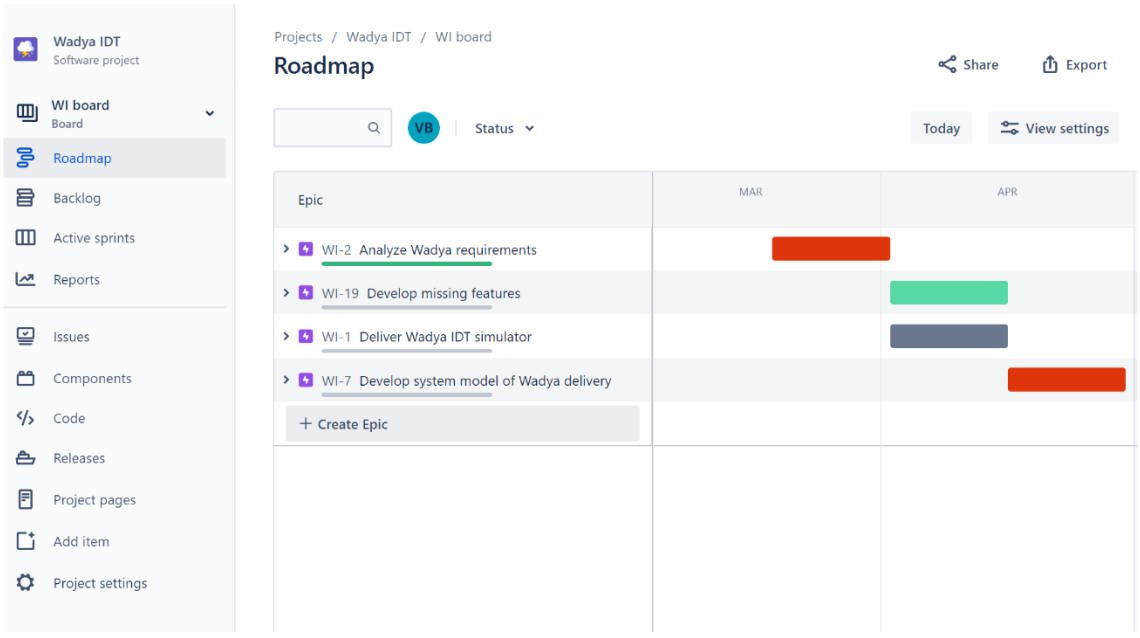


Obrázek 17 - Průběh sprintu č. 2



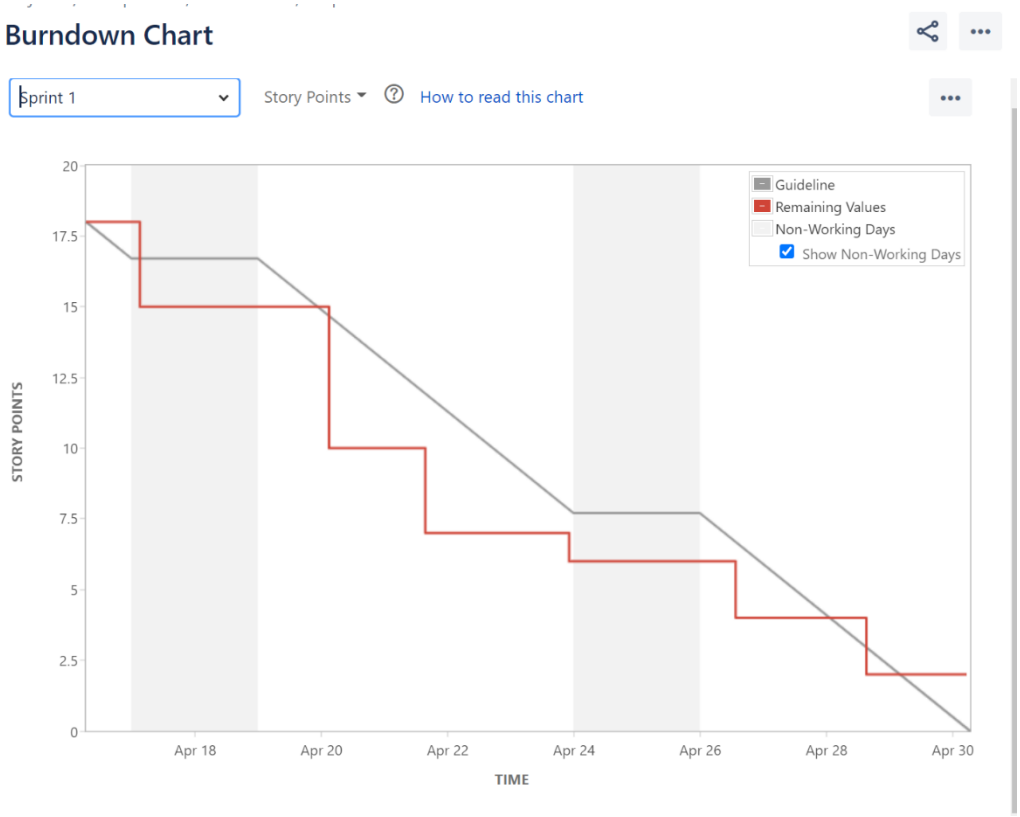
Obrázek 18 - Průběh sprintu č. 3

Následující roadmapa (viz Obrázek 19) vyobrazuje vizualizaci sprintů v čase, která poslouží týmu pro lepší plánování a řízení společné práce.



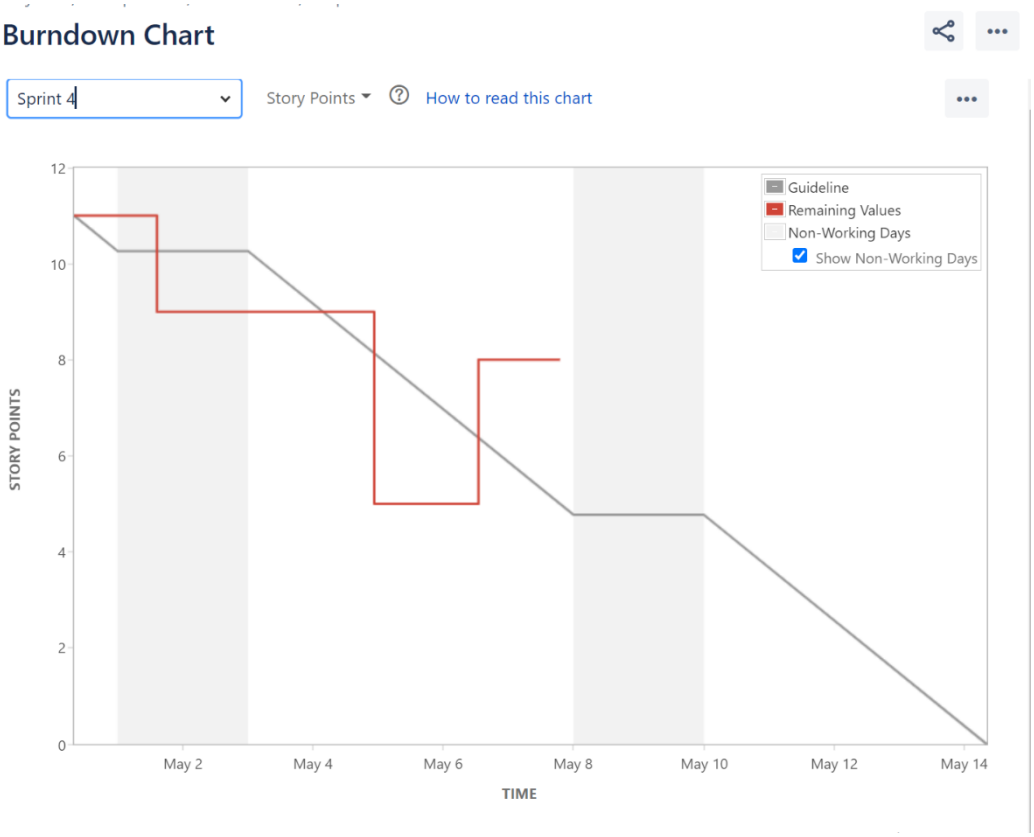
Obrázek 19 - Roadmapa

Níže vyobrazené Burndown Charty graficky znázorňují průběh sprintů, přesněji sprintu 1 a 4. Sprint 1 (viz Obrázek 20) ukazuje již hotový sprint. Vyobrazené křivky značí předpokládaný průběh řešení úkolů ve sprintu, což značí v grafu šedá křivka a reálné provedení v čase, které značí křivka červená.



Obrázek 20 - Burndown chart sprint č. 1

Druhý Burndown Chart vyobrazuje sprint 4, který má již také danou předpokládanou křivku, jak by sprint měl v čase probíhat. Také se zde jedná o šedou křivku. Rozdíl oproti předchozímu grafu je takový, že zde je vyobrazený sprint, který nebyl ještě dokončen pro představu, že si jej lze nechat vygenerovat i procesu, aby tým věděl, jak moc se blíží nebo naopak neblíží ideálu křivky sprintu. (viz Obrázek 21)



Obrázek 21 - Burndown chart sprint č. 2

4.2.6 Přínosy navrhovaného řešení

Přínosy, které z navrhovaného řešení pro firmu mohou plynout, jsou viditelné především v podrobnějším rozpracování postupu ve zvolené webové aplikaci. Ta byla zvolena po konzultaci se softwarovým specialistou, tak aby byl návrh pro firmu co nejlépe realizovatelný. Vyzdvihla bych zde především velmi přívětivé a silně procesně orientované uživatelské rozhraní, které tím že je provozované ve vnitrofiremním cloudu, umožňuje vývojářům pohodlnou práci bez ohledu na to, kde se právě nacházejí – není tedy rozdíl, zda člověk pracuje z kanceláře či z homeoffice.

Co se týče samotné práce v tomto programu, je i pro laika srozumitelná a pod dozorem pracovníka z praxe pochopitelná. Nespornou výhodou je také fakt, že firma již podle analýzy současného stavu aplikaci využívá na aktuální tradiční řízení, tudíž nasazení mnou navrhnované metodiky nepovede ke zvýšení provozních nákladů spojených s používáním aplikace. Firmě tak mé návrhy poslouží jako šablony a příklady pro doškolení pracovníků v aplikaci, kterou využívají na svou práci denně.

Přínosy pro firmu, které z návrhové části plynou:

- Procesní podporu pro nové řízení projektů ve zvoleném programu
- Podrobný popis implementace nového přístupu, který časově i efektivně zlepší práci týmu na projektu
- Praktická ukázka vybrané metodiky k zavedení, včetně vyorazených vzorů a šablon pro tvorbu grafického zobrazení a podrobných popisů úkolů

Do budoucna zde vidím velký potenciál ve srovnání s aktuálním řízení především v lepší komunikaci v softwarovém týmu, a také i při komunikaci se zákazníkem, který díky novému přístupu bude moci více ovlivňovat vývojovou fázi průběhu projektu. Zákazníkova účast na vývoji, je zde právě díky sprintům, které mu umožňují práci na konci každého sprintu zhodnotit a předat tak vývojářům tolik potřebnou zpětnou vazbu. Tato vyzdvihnutá možnost tak ukazuje přesný opak toho, co nastávalo při zpětných vazbách v aktuálním přístupu řízení, které podrobně popisuje podkapitola historických projektů a jejich zpětných vazeb.

4.3 Shrnutí

Návrh řešení a výběr vhodné metodiky byl konzultován po celou dobu tvorby práce se softwarovým specialistou a projektovým manažerem v oblasti vývoje čili zpětná vazba na realističnost zavedení byla poskytnuta přímo od pracovníků z praxe.

Než byl samotný návrh řešení prezentován, předcházela tomu analýza vícera metodik spadajících do agilního řízení, ze kterých dle porovnávacích kritérii vyšel vhodný SCRUM. Tato metodika vyšla vhodná z více důvodů, a tak lze konstatovat, že vývojovému týmu přinese vyšší efektivitu a usnadní tak jeho práci. Hlavním důvodem, proč je pro tuto společnost tato metodika vhodná je to, že dodává software do simulátorů zbraní produktovým způsobem, a tak je schopna na konci každého sprintu demonstrovat zákazníkovi aktuální stav produktu. Zákazník tak může snadno poskytnout zpětnou vazbu.

Velkou výhodu pro firmu je také to, že program, ve kterém byl vzorový projekt navržen, již využívají. Jednalo by se tedy jen o doškolení pracovníků v práci s podporou agilních metod v programu JIRA, který aktuálně využívají na tradiční řízení.

Firma již nějakou dobu zvažovala zefektivnění procesu vývoje produktu. Navrhovaná metodika byla jednou ze zvažovaných a preferovaných možností. Tato práce poskytla cenná vstupní data pro tzv. „Financial and operational feasibility study“, kterou firma následně použije k vyhodnocení nasazení touto prací navrhované metodiky.

ZÁVĚR

Bakalářská páce se zabývá zavedením vybrané metodiky agilního řízení v projektech, které zpracovává společnost Saab Czech, s.r.o. Společnost v současné době vyvíjí nejen software pro simulátory ručních střelných a protitankových zbraní, ale i bezpečnostní technologie a systémy.

Práce poskytuje teoretická východiska, která popisují problematiku a metody využité pro zavedení agilního řízení u projektů. Dále také analýzu současného stavu vybrané společnosti a jejích požadavků, tvorby projektů a okolí. Výsledky analýz pomohly definovat vhodné realizace projektu v návrhové části.

Výsledkem práce je nastínění vzorového projektu, který poskytne náhled na řízení a tvorbu projektu s metodikou, která byla vybrána pomocí analýzy současného stavu a porovnávacích tabulek s metodikami, které poskytuje teoretická část práce.

Na závěr lze konstatovat, že tato práce úspěšně ověřila možné nasazení metodiky agilního řízení v projektech ve společnosti Saab Czech, s.r.o. Shrnutí možných metodik vhodných pro agilní produktový vývoj umožnilo společnosti vytvořit si představu, jaké klady a zápory mohou ze zavedení této metodiky plynout.

Do budoucna je zde prostor práci obohatit o nové teoretické poznatky i rozšíření návrhů metodik a nástrojů, které by společnost mohla využívat a rozšířit tak agilní řízení.

SEZNAM POUŽITÉ LITERATURY A ZDROJŮ

1. SMOLÍKOVÁ, Lenka. *Projektové řízení: studijní text pro prezenční a kombinovanou formu studia*. Brno: Akademické nakladatelství CERM, 2018. ISBN 978-80-214-5695-2.
2. DOLEŽAL, J., LACKO, B., MÁCHAL, P. *Projektový management podle IPMA.* 2. vyd. Praha: Grada, 2012. ISBN 978-80-247-4275-5.
3. MYSLÍN, Josef. SCRUM: průvodce agilním vývojem softwaru. Brno: Computer Press, 2016. ISBN 978-80-251-4650-7.
4. SCHWALBE, Kathy. *Řízení projektů v IT*: *kompletní průvodce*. Brno: Computer Press, 2011. ISBN 978-80-251-2882-4.
5. ŠOCHOVÁ, Zuzana a Eduard KUNCE. Agilní metody řízení projektů. Brno: Computer Press, 2014. ISBN 978-80-251-4194-6.
6. KORECKÝ, Michal a Václav TRKOVSKÝ. Management rizik projektů: se zaměřením na projekty v průmyslových podnicích. Praha: Grada, 2011. Expert (Grada). ISBN 978-80-247-3221-3
7. LESTER, A. Project Management, Planning and Control: Managing Engineering, Construction and Manufacturing Projects to PMI, APM and BSI Standards. 6th Edition. Oxford: Butterworth-Heinemann, 2013. ISBN 9780080983240.
8. YADAV, S.R. a MALIK, A.K. Operations Research. India: Oxford University Press, 2014. ISBN 978-0-19-809618-4.
9. Flowchart Maker & Online Diagram Software [online]. [cit. 2021-03-17] Dostupné z: https://app.diagrams.net/
10. Visual Paradigm Online Express Edition [online]. [cit. 2021-03-17] Dostupé z: https://online.visual-paradigm.com/
11. CzechAgile [online]. [cit. 2021-04-03] Dostupné z: https://czechagile.cz/SCRUM-guide/
12. Agile SCRUM Process in Nuttshell [online]. [cit. 2021-04-05] Dostupné z: https://medium.com/@realjoselara/agile-SCRUM-process-in-a-nutshell-6ec32a59efb
13. KADLEC, Václav. Agilní programování: metodiky efektivního vývoje softwaru. Brno: Computer Press, 2004. ISBN 80-251-0342-0.

14. Saab Czech, a.s. [online].
Dostupné z: https://www.SAAB.com/markets/czech-republic
15. Manifesto for Agile Software development, 2001 [online]. [cit. 2021-04-27]
Dostupné z: https://agilemanifesto.org/iso/cs/principles.html
16. Burndown Chart: What Is It & How Do i Use It? [online]. [cit. 2021-04-27]
Dostupné z: https://www.projectmanager.com/blog/burndown-chart-what-is-it
17. Velocity [online]. [cit. 2021-04-28]
Dostupné z: https://www.SCRUMinc.com/velocity/
18. Velocity Chart [online]. [cit. 2021-04-28]
Dostupné z: https://www.agile-SCRUM.be/whats-great-SCRUM-methodology/velocity-chart/
19. What is SCRUM? [online]. [cit. 2021-05-07]
Dostupné z: https://www.SCRUM.org/resources/what-is-SCRUM
20. The SCRUM Framework Poster
Dostupné z: https://www.SCRUM.org/resources/SCRUM-framework-poster
21. YATIN. What is the difference between Traditional and Agile Project Management [online]. 2017[cit. 2021-05-07].
Dostupné z: https://upraise.io/blog/traditional-vs-agile-project-management-3/
22. JIRA [software]. ©2020
Dostupné z: https://www.atlassian.com/software/jira
23. Microsoft Project [software]. ©2019
Dostupné z: https://www.microsoft.com/cs-cz/microsoft-365/project/project-management-software
24. DOLNÍČEK, D., *Projektové řízení ve firmě* [ústní sdělení]. Saab Czech, s.r.o., Čelakovského 689, 684 01 Slavkov u Brna, 8. 2. 2021
25. BÍLEK, V. *Vývojový proces ve firmě* [ústní sdělení]. Czech, s.r.o., Čelakovského 689, 684 01 Slavkov u Brna, 11. 3. 2021
26. BÍLEK, V. *Práce s programem JIRA* [ústní sdělení]. Saab Czech, s.r.o., Čelakovského 689, 684 01 Slavkov u Brna, 8. 5. 2021

POUŽITÉ ZKRATKY A SYMBOLY

CDR	Critical Design Review
COTS	Commercial off-the-shelf
ERP	Enterprise Resourse Planning
FDD	Feature Driven Development
PDM	Product Data Management
PDR	Preriminary Design Review
SRR	System Requirement Review
TDD	Test Driven Development
XP	Extreme Programming

SEZNAM UŽITÝCH OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH

Příloha 1- Nedostatky projektů a jejich zpětné vazby

	NEDOSTATKY V PROJEKTU	ZPĚTNÁ VAZBA
PROJEKT 1	Z počátku nejasné požadavky od zákazníka, které se dokonce v průběhu projektu měnily, přičemž termín dodání v rámci udržení dobrých vztahů zůstal nezměněn. V rámci běžícího projektu byl zaveden nový nevyzkoušený systém DevOps pro trackování požadavků, či sledování "Percentage of completion", to způsobilo mnoho problémů.	Zákazník velmi ocenil naši schopnost adaptace na vyžádané změny. Systémy jsou významně využívané po celé zemi a mají velkou publicitu. Aktuálně jednáme o rozšíření objednávky o další systémy.
PROJEKT 2	Nedostatečné kapacity vývojářů, kteří byli alokováni na jiný projekt s větší prioritou. Z tohoto důvodu byl projekt pozastaven a dodán se sankcemi po termínu.	Po předání dodávky máme velmi pozitivní ohlasy, neboť speciálně v době pandemie vojáci nemohou používat v rámci tréninku hojně ostré zbraně, a tak je simulátor plně vytížen.
PROJEKT 3	Jako dodavatelé jsme neměli přímou vazbu na koncového zákazníka, neboť byl mezi námi prostředník. Tento prostředník nefungoval spolehlivě a docházelo tak často k nedorozuměním a k odkládání nezbytných rozhodnutí. Díky tomu byl zákazník při první dodávce nespokojen s kvalitou, zpětná vazba se k nám dostala pozdě, a proto i druhá dodávka měla své nedostatky.	S prvními dodávkami byl zákazník nespokojen, jakmile ale došlo k přímému spojení, vše se vyřešilo. Naše systémy jsou nyní značně využívány a těší se mezi uživateli velké oblibě.
PROJEKT 4	Jazyková bariéra, velká mezikontinentální distance mezi dodavatelem a zákazníkem. Z toho plynoucí nesrovnalosti.	Nízká zpětná vazba ze strany uživatele. Nicméně jsme obdrželi poptávku na generální servis a aktualizaci software na nejnovější verzi.
PROJEKT 5	Vzhledem k vágně popsaným požadavkům velmi netradiční projekt. Při instalaci systémů u zákazníka požadavky byly měněny.	Téměř žádná zpětná vazba. Dodané systémy pravděpodobně neužívány.