Vysoká škola ekonomická v Praze

Fakulta informatiky a statistiky

KVALITNÍ BACKLOG

Author: Maxim Dužij, Jakub Holík, Vítězslav Slavík

Úvod

Cílem práce je definovat backlog a kontext jeho použití v oblasti agilních praktik při vývoji softwaru. V druhé části práce se zaměříme na metodiky a pravidla pro plnění backlogu (INVEST a DEEP), posouzení kvality (BAM) a špatné vzorce při jeho použití.

Abstrakt

Práce se zabývá kvalitou backlogu, hodnocením stavu backlogu a jeho plněním. Cílem bylo představit metodiku BAM skrze její definice a praktické použití. Nejdříve je definován pojem backlogu a jednotlivých jeho částí. Následně jsou představeny metodiky DEEP a INVEST, které mohou být použity pro kvalitní ztvárnění uživatelských příběhů a udržitelné plnění backlogu. Zároveň jsou představeny metody, které naopak mohou poškodit práci s backlogem. Ve výsledku se představí BAM metodika pro zhodnocení backlogu. Teoretická část je podpořena praktickou ukázkou aplikace metodiky BAM na malém backlogu a jejími výsledky.

Klíčová slova

Backlog, INVEST, DEEP, BAM

Obsah

1 Backlog a jeho úloha

Backlog je nejpoužívanější abstrakce v kontextu zavedení Scrum metodiky a někdy také označován jako „Srdce Scrumu“ (the heart of Scrum) (Rubin, 2013)

1.1 Definice

Cambridský slovník definuje slovo backlog jako veliký seznam věcí, které je potřeba udělat. (Anon., 2020) Tento obecný význam se ustálil v běžném pracovním prostředí a slovo je vnímáno spíše negativně.

Jako termín se slovo backlog používá převážně v odvětví vývoje softwaru. Zde je ke slovu backlog nutné přidružit oblast, kterou backlog mapuje. Pokud vezmeme jako příklad produktový backlog, dle (Sedano, et al., 2019) lze definovat jako list pracovních položek, který obsahuje například uživatelské příběhy vztažené ke specifickému produktu, neopravené chyby v programu a technická práce při vývoji softwaru (tzv. chores).

Tuto definici lze aplikovat i na další typy backlogů, které se však budou lišit oblastí, které se věnují. Těmto rozdílům je věnována samostatná kapitola “Typy backlog a jejich vztah”.

1.2 Backlog v kontextu agilního vývoje

S přechodem na agilní směr vývoje se změnily metody, procesy i nástroje používané týmy. Jedním z artefaktů, který doznal markantní změny byla dokumentace požadavků. Detailní specifikace softwaru, která je, pokud možno neměnná po čas vývoje, byla v přímém rozporu s druhým principem agilního vývoje “Vítáme změny v požadavcích, a to i v pozdějších fázích vývoje.” (Ken, et al., 2001) Bylo nutné přejít na flexibilnější řešení.

Odpovědí se stal produktový backlog. Oproti klasické dokumentaci požadavků je backlog vždy neúplný. Nedefinují se zde zbytečné funkce, které nejsou v blízké budoucnosti potřeba. Tím splňuje další z principů “*Jednoduchost--umění maximalizovat množství nevykonané práce--je klíčová*.” (Ken, et al., 2001)

Backlog je “živým” artefaktem. Položky se mění a vyvíjejí v čase. Vlastník mění priority položek backlogu podle aktuální potřeby. Tato dynamika poskytuje prostor na reakce týmu na zpětnou vazbu uživatelů.

Pro vývoj v agilním prostředí je využíván fenomén “dual-track agile”. Aktivity jsou rozděleny mezi dvě paralelně na sobě pracující skupiny. První skupina zahrnuje zkoumání byznys poptávky, vytváření mockupů a budování uživatelských příběhů. V druhé skupině je pozornost zaměřena na vývoj a implementaci. (Sedano, et al., 2019) Backlog pomáhá

propojit komunikaci obou skupin a pochopení produktu a jeho vývoj. Často bývá také protokolem diskuze nad jednotlivými položkami.

SCRUM definuje hned několik ceremonií pro které je backlog hlavním předmětem diskuze. První je schůzka pro plánování sprintu, na kterém se definuje cíl následujícího sprintu a k němu související položky do sprintového backlogu (více v kapitole 1.5.1).

Další ceremonií zahrnující práci s backlogem je posouzení sprintu („Sprint Review“), na které vlastník produktu objasňuje, které položky byly dokončeny a které nikoliv. Zhodnocuje se také aktuální situace na trhu, která ovlivňuje potenciální využití produktu a může vést k reprioritizaci položek v backlogu. (Schwaber & Sutherland, 2017)

1.3 Produktový backlog

Produkt backlog je informačním modelem mapující produkt. Jedná se zdroj pravdy pro všechny členy vývojářského týmu. (Larman, et al., 2012) Nejedná se však o místo pro specifikaci požadavků ani detailní návrh. (Sedano, et al., 2019)

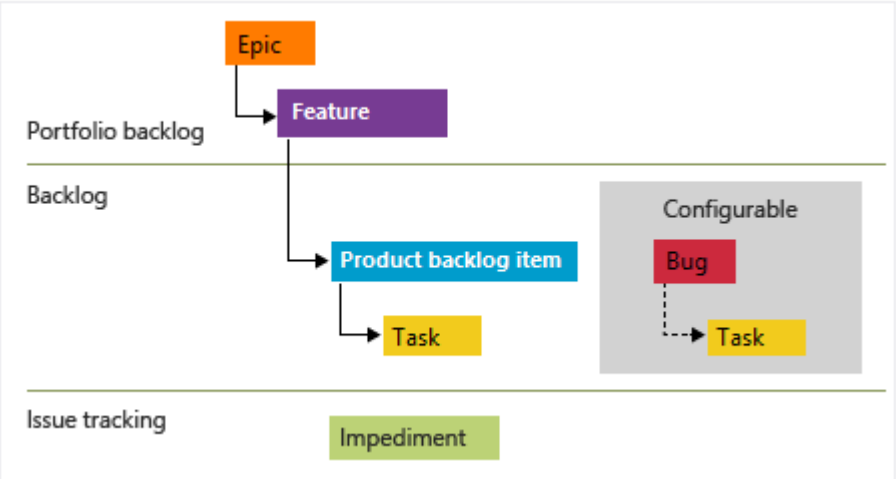
Položka produktového backlogu, zkráceně také někdy označována jako PBI (product backlog item), nemusí popisovat jen novou funkcionalitu pro uživatele. Dalšími příklady mohou být například inženýrská vylepšení (*aktualizace používaných knihoven*), cíle týmu (*zrychlení procesu testování*), výzkum (*prozkoumání možností dvoufázové autentizace uživatele*) nebo oprava defektů (*diagnostika a oprava chybného řazení zboží v košíku).* (Sedano, et al., 2019)

1.3.1 Product owner

Jedinou osobou odpovědnou za obsah, dostupnost a prioritizaci backlogu je vlastník produktu (“Product owner”). Má na starost jasné vyjádření položek a jejich řazení podle priority, aby bylo co nejlépe dosaženo cílů a poslání. Také zajišťuje, aby byl backlog přehledný a transparentní pro všechny zúčastněné. (Schwaber & Sutherland, 2017)

1.3.2 Hierarchické členění položek

Produktový backlog je složen z několika typů položek. Nejvyšší úrovní je tzv. Epic. Jedná se o iniciativu byznysu a slouží k definování směru vývoje. Je také popisován jako rozsáhlý uživatelský příběh převážně pro účely stakeholderů, uživatelů a zákazníků. (McDonal, 2018) Někdy se spolu s “Feature” řadí do portfolio backlogu, pokud tým vyžaduje přísnější dělení. Mezi takové položky patří například implementaci podpory mobilních zařízení, integrace s externími službami, zjednodušení uživatelského rozhraní. Tyto položky jsou kvůli své velikosti implementované skrze několik vydání (“releasů”) aplikace. (Microsoft Corporation, 2020)



Obrázek 1: Položky backlogu definované v Azure Devops nástroji pro správu backlogu (Microsoft Corporation, 2020)

O úroveň níže je “Feature” nebo také někdy nazvané jako “Theme”. Jedná se o jednotku softwaru, jejíž implementace trvá více než 1 sprint. Je k ní přistupováno jako ke kolekci sdružující související uživatelské příběhy. (Rubin, 2013) Klasicky jde o modul, přidávající aplikaci funkcionalitu.

Položka produktového backlogu (“Product backlog item”) je název pro uživatelský příběh. Tato položka je neformálním popisem aspektu softwaru. Jeho formulace se může nést například v duchu “Connextra” šablony. *Jako <uživatel> chci <akce>, tak aby <hodnota>*. (Sedano, et al., 2019)

Toto hierarchické členění zajišťuje dostatečnou úroveň detailu pro všechny členy týmu a zároveň poskytuje nadhled a celkový obrázek o směru vývoje.

1.4 Product Backlog Refinement

SCRUM metodika vybízí k tomu, aby se v každém sprintu věnoval určitý čas na tzv. “Backlog grooming”, jehož účelem je příprava backlogu pro další sprint. Jedná se o celotýmovou schůzku, ve které se mimo jiné řeší detailnější popis položek, jejich rozdrobení na menší části a také hodnocení jejich náročnosti. (Larman, et al., 2012)

Scrum vysloveně neříká, jak takový Backlog Refinement provádět, avšak často má podobu týmového workshopu v druhé polovině sprintu. Účelem je věnovat se položkám plánovaných pro dalších pár sprintů.

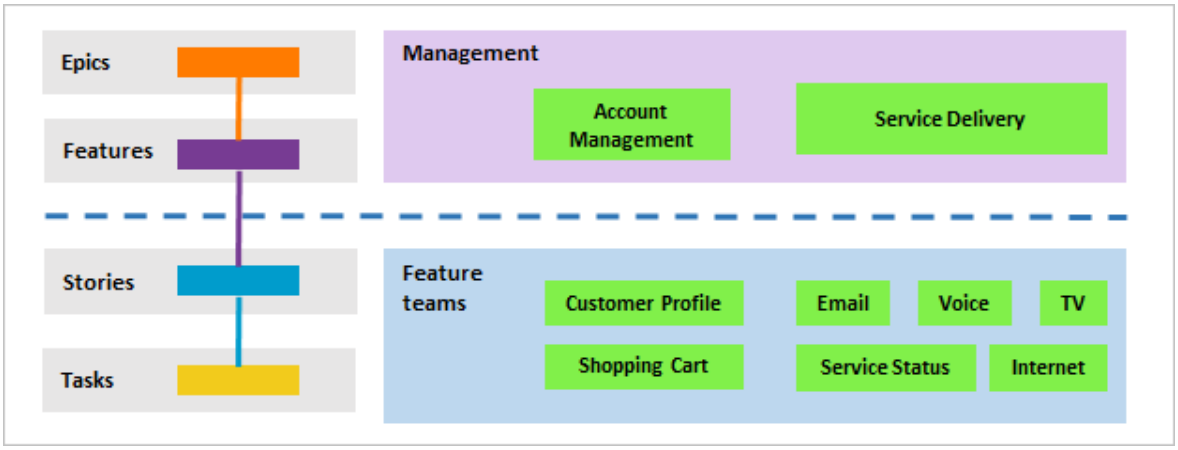
Příznaky špatného backlog refinementu se často objeví u plánování dalšího sprintu. Položky nejsou správně ohodnocené z hlediska náročnosti a vyvstávají důležité otázky ohledně uživatelských příběhů a jejich návrhu. To má za následek přesah plánování do samotného sprintu, což není žádoucí. (Larman, et al., 2012)

1.5 Další typy backlogů a jejich vztah

V předchozích kapitole jsme popsali produktový backlog a v této kapitole se podíváme na další typy.

1.5.1 Portfolio backlog

Tento backlog je nejvyšší úrovní hierarchie backlogů. Obsahuje položky definované na strategické úrovni. Tyto položky se vyznačují tím, že je třeba investovat velké úsilí pro implementaci. Portfolio backlog je vhodný převážně pro vlastníka produktu a slouží pro definici byznys perspektivy a ideje. (Arora & Shigihalli, 2019)



Obrázek 2: Rozdělení položek mezi dvě skupiny týmů v dual track agilním vývoji (Microsoft Corporation, 2020)

1.5.2 Sprintový backlog

Před zahájením sprintu v metodice SCRUM je potřeba mít definovaný sprintový backlog. Jeho obsah je předmětem ceremonie pojmenované povětšinou jako sprintová plánovací schůzka.

Sprintový backlog je podmnožinou toho produktového. Obsahuje jen ty položky, které napomáhají splnit cíl sprintu definovaný na začátku schůzky produktovým vlastníkem.

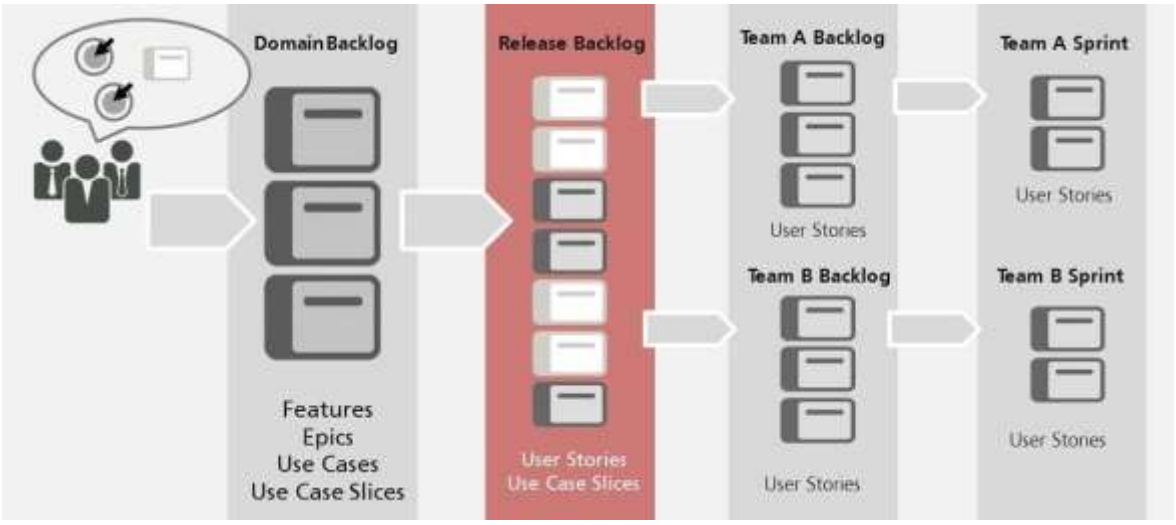
Jelikož je kapacita sprintu omezená, je limitován i obsah jeho backlogu. Každá položka by měla být ohodnocená člověkem, který ji bude v daném sprintu provádět. Náročnost všech položek zařazených do backlogu je poté rovná kapacitě týmu za daný sprint.

Během plánování dochází ještě k rozpadu položek na menší jednotky. Ohraničení této jednotky by mělo být dostatečně granulární, aby práce na ní nepřekročila jeden den.

Na schůzce nazvané „Denní SCRUM“ členové týmu probírají práci provedenou na položkách v tomto backlogu a odhadují práci na položkách, které se chystají udělat.

1.5.3 Release backlog

Tento backlog je podmnožinou produktového backlogu. Oproti sprintovému backlogu, u něhož jsou položky časově ohraničeny, v release backlogu je ohraničení na základě verze. Obsahuje definované a prioritizované položky, které je třeba implementovat do dalšího vydání produktu. (Anon., 2020)



Obrázek 3: Diagram různých backlogu dle jejich velikostí od nejobsáhlejšího po nejmenší (microTOOL, 2020)

Tento typ backlogu není standardní součástí vývoje podle metodiky SCRUM. Podle (Cohn, 2009) je release backlog ve světe agilního vývoje nadbytečný, jelikož se málokdy položky v daném vydání nemění. Další vrstva dle něj může přinést zmatek v týmu a náročnost na komunikaci, prioritizaci a správu backlogů.

2 Praktiky a pravidla plnění backlogu: metody DEEP a INVEST

Použitelnost produktového backlogu a efektivita vývojového týmu je přímo závislá na kvalitě uživatelských příběhů vkládaných do backlogu. Je potřeba neustále kontrolovat kvalitu vkládaných a již vložených uživatelských příběhu do produktového backlogu s cílem zajistit co nejhladší průběh práce celého týmu, kterému slouží backlog jako zadání práce na následující sprint.

Jakým způsobem je možné dlouhodobě udržet backlog v ideálním stavu? Jakými kroky zajistit jeho plnění udržitelným způsobem, aby toho mohl tým využít? Na tyto otázky se bude následující kapitola snažit nalézt odpovědi díky analýze dvou předních metodik, INVEST a DEEP, které se snaží nasměrovat vlastníka produktu ke vkládání uživatelských příběhů standardizovaným způsobem.

2.1 Rozbor metodiky DEEP

Metodika DEEP je nástrojem pro vlastníka produktu k efektivnímu řízení uživatelských příběhů v backlogu. Jedná se o jednoduchou, lehce zapamatovatelnou metodiku, kterou lze rychle aplikovat. Dle spoluzakladatele, Roman Pichler, je potřeba pravidelných backlog refinementů k zajištění backlogu odpovídajícího metodice DEEP.

Název metodiky je složen ze čtyř klíčových atributů: **D**etailed appropriately, **E**stimated, **E**mergent a **P**rioritized. Každý z atributů tvoří dohromady celek, který pomáhá backlogu zůstat udržitelný a na kvalitní.

Detailed appropriately (Dostatečně přesné): Uživatelské příběhy s vysokou prioritou jsou detailně popsané a jejich krajní případy pečlivě zanalyzovány na rozdíl od příběhů s nižší prioritou, neboť ty budou zpracovány až posléze. Příběhy také obsahují dostatečné množství kontextuálních informací umožňující týmu problematiku pochopit a diskutovat.

Estimated (Odhadnuté): Všechny položky v backlogu by měly mít odhad náročnosti. Ty s vyšší prioritou mají samozřejmě přesnější odhad pracnosti na základě většího detailu popisu a dodaného kontextu než ty s prioritou nižší. Odhady by měly být považovány za proměnné s možností přehodnocení při každém sprintu z důvodu rozvíjení se znalostí a zkušeností týmu. Uživatelský příběh také může obsahovat odhad hodnoty pro zadavatele na základě možné přidané hodnoty produktu nebo možného přínosu zisku po implementaci. Tyto hodnoty dosazuje vlastník produktu.

Emergent (Vyvíjející): Backlog se v průběhu času vyvíjí. Mění se priority jednotlivých příběhů, jejich popis, odhad pracnosti nebo hodnoty pro vlastníka produktu. Nové příběhy přibývají nebo se ruší. Řídí se heslem, že nic není věčné.

Prioritized (Prioritizované): Položky v backlogu jsou seřazeny dle priority, kterou určuje klient nebo vlastník produktu. Obecně platí, že položky s nejvyšší prioritou by měli dodat co největší hodnotu vytvářenému produktu za co nejnižší cenu – tedy pracnost uživatelského příběhu.

Backlog odpovídající DEEP metodice a jejím klíčovým atributům by měl být výstupem backlog refinementu, což pomáhá lepší prioritizaci jednotlivých uživatelských příběhů.

2.2 Rozbor metodiky INVEST

Na rozdíl od metodiky DEEP, která se aplikuje na backlog samotný, metodika INVEST se zaměřuje na samotné uživatelské příběhy v backlogu. Sestavena Bill Wakem, lze použít na validaci uživatelského příběhu vůči jednotlivým kritériím INVEST.

Zároveň se, stejně jako v případě DEEP, jedná o akronym pro jednotlivé kritéria a to: **I**ndependent, **N**egotiable, **V**aluable, **E**stimable, **S**mall, **T**estable.

Independent (Nezávislé): Jednotlivé uživatelské příběhy jsou na sobě nezávislé, co se týče specifikace a implementace. Neměly by se ani čáastečně překrývat v těchto oblastech a jsou tedy i plánovatelné nezávisle na sobě.

Negotiable (Diskutovatelné): Příběh by měl ve své definici pojmout jádro požadované funkcionality nebo problému a poskytnout prostor pro diskuzi ohledně jeho implementace a detailů, které nejsou definované. Můžeme říct, že uživatelský příběh by měl být v přeneseném slova smyslu připomínkou pro tým a vlastníka produktu, aby jej dopodrobna prodiskutovali a rozšířili o poznámky, návrhy na testování a podobně.

Valuable (Hodnotné): Hodnota pro zákazníka nebo zadavatele produktu by měla být z uživatelského příběhu jasně zřetelná pro vývojový tým a vlastníka produktu. Příběh by měl být formován takovým způsobem, aby bylo zřejmé, proč je pro zákazníka důležitý.

Estimable (Odhadnutelné): Správně vytvořený příběh by měl být především odhadnutelný. Není třeba zcela přesný odhad pracnosti, postačí rámcový odhad pro snazší zařazení do fronty a určení priority pro zákazníka. V podstatě se jedná o funkci diskutovatelnosti, velikosti a zkušenosti týmu. Příběh můžeme odhadnout, pokud mu detailně rozumíme, a to právě po prodiskutování veškerých aspektů týmem. Zároveň menší příběhy jsou snadnější na odhadnutí než ty větší. Ve výsledku ale vše závisí na úrovni zkušenosti vývojového týmu.

Small (Malé): Malé příběhy jsou obecně ohodnocovány s větší přesností na základě snazší uchopitelnosti vývojovým týmem. V opačném případě je složité pochopit celý rozsah příběhu, který mohl být rozdělen na více menších částí s nižší pracností.

Testable (Testovatelné): Uživatelský příběh je možné ohodnotit jako správný, pokud je testovatelný. To znamená, že vývojový tým zná dostatečně jeho popis a cíl na to, aby na něj dokázal napsat test. Test zároveň dokazuje dosažení cíle příběhu. (Sharma, 2019) (Wolpers, 2019)

2.3 Špatné techniky plnění backlogu

Metodiky DEEP nebo BAM popisují praktiky správné práce s backlogem a jeho plněním. Tyto ovšem nepopisují, nebo jsou vágní v popisu druhé strany problematiky, a to jsou metody špatné práce s backlogem. Tato podkapitola poukáže na nejčastější chyby při práci s backlogem a jeho naplňování z více úhlů pohledu.

2.3.1 Obecný pohled

Není třeba **vytvářet všechny uživatelské příběhy na začátku projektu** tak, aby pokryly celý produkt, jen proto, že rámec projektu je omezený. Je totiž nereálé odhadnout potřeby projektu v následujících měsících.

Neméně návazná je tvorba příliš mnoha položek v backlogu a na jejich základě **tvorba sprintů na dlouhou dobu dopředu**. Pokud vlastník produktu vytvoří sprinty například na půl roku dopředu, jakým způsobem poté bude reagovat na nenadálé nebo akutní potřeby trhu, na který produkt míří? V případě vysoce konkurenčních trhů se doporučuje plánovat nanejvýš tři až čtyři sprinty dopředu.

Přemíra nebo absence akceptačních kritérií dokáže značit nekvalitně zpracovaný uživatelský příběh. Na začátku cyklu příběhu není potřeba mít stanovená akceptační kritéria, jejich stanovení ovšem usnadní řízení a práci s příběhem. Naopak nadbytek kritérií naznačují absenci komunikace vlastníka produktu s vývojovým týmem. Není stanovené číslo na ideální počet akceptačních kritérií, obecně se bere počet tři až pět za odpovídající.

2.3.2 Pohled vlastníka produktu

Jednoduché **rozpadnutí dokumentu požadavků na jednotlivé menší kousky** v podobě uživatelských příběhů je nedostatečné pro správně fungující backlog. Ukazuje to na neexistující přínos role vlastníka produktu. Příběh by měl být diskutován a upravován týmem a předem vůbec zpracován vlastníkem produktu.

Naopak **dominantní chování vlastníka produktu při zpracování požadavků** je také nežádoucí. Vlastník produktu nemá specifikovat co a jak zpracovat v rámci příběhu.

Toto je právě úkolem vývojového týmu na základě předchozí diskuze celého týmu. Vlastník produktu má především specifikovat motivující „proč“.

INVEST. Pokud tato metodika není použita, nedá se mluvit o kvalitním zpracování příběhu a v dlouhodobém pohledu se její ignorování projeví na fungování celého týmu a vývoje produktu.

2.3.3 Pohled vývojového týmu

Submisivní chování vývojového týmu škodí při interakci s vlastníkem produktu, který ve výsledku nemusí vymýšlet argumenty pro své požadavky a jednoduše je předkládá k vypracování. Povinností každého člena týmu by mělo být diskutování jednotlivých požadavků vlastníka produktu, což vede k lepšímu fungování projektu.

Technický dluh a chyby by neměly být opomíjeny vývojovým týmem. Tento by měl požadovat po vlastníkovi produktu adekvátní zdroje pro nápravu chyb v produktu a vyrovnání technického dluhu.

Vývojový tým by neměl být využíván na **100 % jeho možného výkonu** po celou dobu projektu. V takovém případě začne přirozeně klesat produktivita jednotlivých členů, protože nemají prostor na regeneraci. Každý člen týmu se bude soustředit pouze na dokončení svého úkolu a nebudou mít možnost na konzultace jiných uživatelských příběhů jejich kolegů. Malé příběhy budou více opomíjeny ve prospěch větších a budou se v backlogu hromadit. Výsledkem bude také snížení společného porozumění uvnitř týmu.

2.3.4 Pohled Scrum týmu

Opomíjení **refinementu backlogu** vyústí v celkovou nízkou kvalitu produktového backlogu. Z pohledu zadavatele totiž není nic dražšího než prvek produktu, který nepřináší žádnou hodnotu a pokud se bude refinement zanedbávat, pár takových prvků se může dostat do vývoje. Obecně se doporučuje věnovat refinementu 10 % celkového času Scrum týmu. Opačné nebezpečí přináší příliš detailní nebo častý refinement, jehož výsledkem je příliš detailní backlog.

Absence **definice připravenosti** jednotlivých uživatelských příběhů zabraňuje přidání jednotlivých příběhů do sprintu. Přitom jednoduchý seznam atributů značně zlepší práci Scrum týmu a obecně i kvalitu výsledku uživatelského příběhu. (Wolpers, 2019)

3 Posuzování kvality a analýza backlogu. Metodika BAM

V této kapitole bude popsána metodika analýzy backlogu BAM (Backlog Assessment Method). Jedná se o velmi jednoduchou metodiku na zavedení a pochopení. Má za úkol zhodnotit aktuální stav projektového backlogu na základě kterého se dají např. zavádět pravidla a opatření pro naplnění backlogu do budoucna.

Důvodů, proč metoda analýzy jako je BAM musela vzniknout, je hned několik. Ten první je potřeba objektivně hodnotit něco tak rozmanitého jako backlog a jeho položky (work items), ze kterých se skládá. U rozsáhlejších backlogu, pro které byla metodika BAM vyvinutá a vyzkoušená, mohou mít položky různou strukturu a míru podrobnosti (Berntsson Svensson, et al., 2019).

Další důvod je možnost zhodnotit, zda aktuální stav backlogu odpovídá aktuálním potřebám v projektu nebo firmě. Zavedení metodiky BAM včas může např. předejít situacím kdy položky budou obsahovat buď málo nebo hodně informací. V některých stádiích rozhodování se mohou položky opětovně odkládat do dalšího sprintu kvůli nedostačujícím informacím. Naopak pokud budou položky popsané více, než je potřeba, může to vést k delším diskusím během retrospektivy nebo delšímu rozhodování, zda danou položku umístit do dalšího sprintu. Stejně tak, pokud budou úlohy moc obsáhlé a podrobné, může odhad, kolik práce daná položka zabere, být příliš vysoký.

Metodika BAM je jednoduchá pro zavedení ve společnostech, které využívají agilní praktiky pří vývoji softwaru. Autoři metodiky jí řadí mezi SPI (Software Process Improvements) metodiky a dá se aplikovat na backlogy různých velikostí a různorodou strukturou jednotlivých položek. BAM také nabízí pomůcky pro klasifikaci položek backlogu. Na základě výsledků, které metoda BAM nabízí, můžeme posoudit, zda aktuální stav backlogu vyhovuje všem aktuálním potřebám a požadavkům vývojářského týmu. Položky backlogu se posuzují podle čtyř perspektiv: Scope, Abstraktion/Level, Maturity, Detail. Jednotlivé kroky metodiky BAM jsou v detailu rozebrány v dalších kapitolách.

Tato kapitola a její podkapitoly se budou opírat ve velkém o článek z dvacáté mezinárodní konference XP 2019, kde se tato metodika popisuje a názorně ukazuje na příkladu. Jelikož je metodika BAM na moment napsání této semestrální práce velice mladá, nebyly nalezeny žádné další články, kde by se metodika rozebírala nebo používala v praxi kromě příkladu z článku konference.

V článku se rozebírá analýza projektového, produkčního a sprintového backlogu švédské telekomunikační společnosti Telenor Sweden. Tato společnost již léta úspěšně používá metodiku Scrum pro vývoj svých softwarových řešení, a proto byla vybrána jako kandidát pro implementaci analyzační metodiky BAM (Berntsson Svensson, et al., 2019).

3.1 Relevantní Backlog

Prvním krokem pro zavedení metodiky BAM je výběr konkrétního backlogu, na který se bude metodika aplikovat. Může se jednat o produktový nebo sprintový backlog. Důležité je, aby analýza probíhala právě jednoho konkrétního backlogu.

3.2 Relevantní položky

Dalším krokem je výběr položek z backlogu. Jak už bylo zmíněno v úvodu, metodika BAM počítá s tím, aby měly položky backlogu různou strukturu např. diagramu užití, scénářů užití nebo user story formát.

Pro přesnější analýzu je dokonce autory doporučeno schválně vybírat položky backlogu takové, které mají různý formát.

Počet položek, které se pro analýzu vybírají, by měl odpovídat asi deseti procentům od celkového počtu položek. V názorném příkladě, který uvádějí autoři metodiky, těchto deset procent odpovídalo stovce položek. Autoři metodiky v tomto kroku doporučují vybírat od padesáti do sta položek v závislosti od celkové velikosti backlogu.

3.3 Relevantní zainteresované strany

Na projektu, který backlog využívá se obvykle podílí mnoho členů týmu a lidí s různými rolemi, znalostmi a zkušenostmi. Tudíž pro první posouzení je přípustné zvolit tým čtyř až šestí lidí.

Lidi v týmu pro posouzení kvality backlogu musí být osoby, které bezprostředně s backlogem pracují. V praxi se často stává, že stejná osoba zastupuje několik rolí. V tomto případě se taktéž podílí na udělení bodů pro hodnocení backlogu několikrát.

V závěru článku je výběr správných zainteresovaných stran zařazen do rizik, se kterých metodika BAM musí počítat. Výběr lidí, které se zavedením metodiky nesouhlasí nebo pracují velice omezeně může vést ke značnému zkreslení celé analýzy.

3.4 BAM perspektivy

Metodika BAM obsahuje čtyři perspektivy, podle kterých se hodnotí každá ze zvolených položek. Každou z perspektiv můžeme označit za úhel pohledu na danou položku.

Rozsah (Scope): první perspektivou je rozsah položky. Pro zařazení do této perspektivy může pomoct odpověď na otázku: jedná se o změnu, která ovlivní jen konkrétní část produktu nebo se jedná o větší úpravu, která povede ke změně na více místech?

Jednotlivé kategorie této perspektivy se označují takto:

1. Izolovaná změna (Isolated change)
2. Změna v podsystému (Subsystem)
3. Změna v aplikaci (Application)
4. Změna v procesu (Process)
5. Změna platformy (Platform)
6. Globální změna (Enterprise)

Jak i samotní autoři metodiky zmiňují, v praxi je zcela běžné, že položky produkčního backlogu jsou obecně globálnější, než položky backlogu pro konkrétní sprint (Berntsson Svensson, et al., 2019).

Úroveň abstrakce (Abstraction/Level): další perspektivou je úroveň abstrakce kterou položka má. Jedná se o změnu v lokálním algoritmu, nějaké systémové vylepšení nebo obecnou vizi do budoucna? Tato perspektiva se může měnit i pro položky, které se zařazují do stejného rozsahu.

Jednotlivé kategorie této perspektivy se dají označit takto:

1. Vize (Vision)
2. Cíl (Objective)
3. Systémová úroveň (Feature)
4. Funkční úroveň (Function)
5. Komponentní úroveň (Component)
6. Lokální algoritmus (Local algorithm)

Zralost (Maturity): při zařazení položky do této perspektivy by se mělo odpovědět na otázku, jak často se zavedená změna bude měnit v budoucnu. Autoři metodiky uvádí, že v praxi se do backlogu, které se používají v počátečních fázích většinou zařazují věci s nižší zralostí a stejně tak např. backlog současného sprintu obvykle obsahuje položky s již vyšší úrovní zralosti.

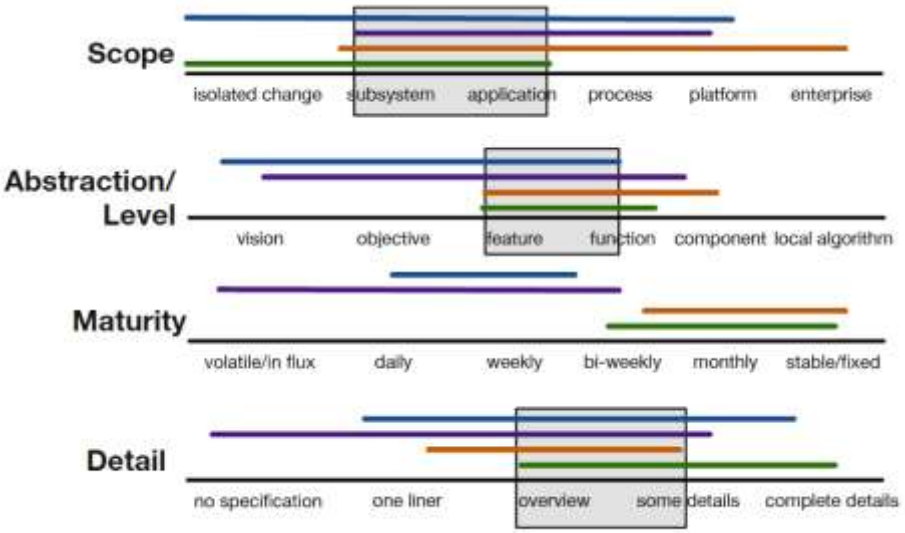
Jednotlivé kategorie této perspektivy se dají označit takto:

1. Nestabilní (Violate/in flux)
2. Denně (Daily)
3. Každý týden (Weekly)
4. Jednou za několik týdnu (Bi-weekly)
5. Každý měsíc (Monthly)
6. Stabilní (Stable/Fixed)

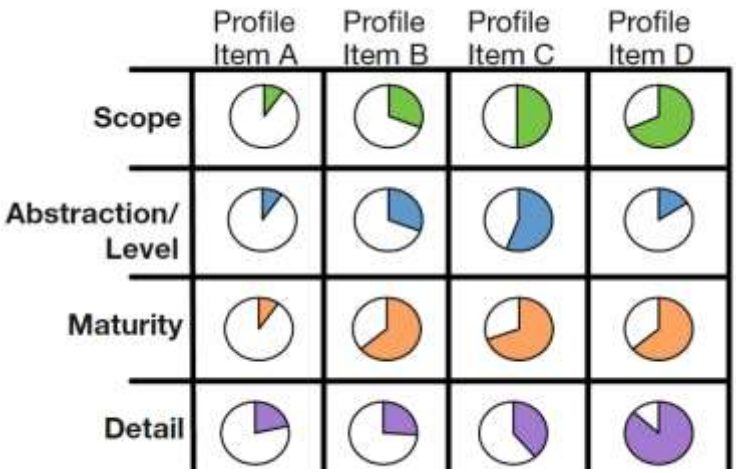
Specifikace (Detail): tato perspektiva by měla kategorizovat položky dle jejich úrovně popisu od těch nejobecnějších po popis konkrétní implementace. V praxi čím blíž je položka se zadáním připravená k samotné implementaci (sprint backlog), tím vyšší je úroveň specifikace. Autoři metodiky uvádí, že neexistuje taková úroveň specifikace, která by vyhovovala ve všech stádiích vývoje. Zařazení položky backlogu do této perspektivy však dává konkrétní výstup pro posouzení, zda je specifikace v kontextu daného backlogu dostačující nebo nikoliv.

Jednotlivé kategorie této perspektivy se dají označit takto:

1. Bez specifikace (No specification)
2. Jedna věta (One liner)
3. Několik vět (Overview)
4. Specifikace (Some details)
5. Detailní specifikace (Complete details)



Obrázek 4: Výstupy analýzy – graf zařazení do kategorií



Obrázek 5: Výstupy analýzy – Nejčastěji objevující se profily položek

Když je zhodnocení položek ukončeno, výstupy se dají využívat k mnoha účelům. V článku se uvádí dva názorné příklady, jak se výstupy hodnocení dají využít. Takových příkladů se ale dá vymyslet více a je jen na společnosti, jak s výstupy naloží a které kroky na základě toho udělá. Taktéž se nabízí možnost sledovat změny v kategoriích v průběhu času a v různých stádiích projektu.

Doporučení od autoru metodiky zní, že je lepší dělat několik menších analýz backlogu než jednu velkou.

Obrázek 4: Výstupy analýzy – graf zařazení do kategorií představuje názorný graf, který vizuálně sumarizuje hodnocení a zařazení do jednotlivých kategorii ve čtyřech perspektivách. Šedé obdélníky vyznačují průměrné hodnoty pro celý backlog, pokud takové existují. Už na základě těchto středních hodnot je možné udělat nějaký posudek o stavu backlogu.

Například pokud by střední hodnota v hodnocení produktového backlogu perspektivy Specifikace byla nízká, je možné se pro budoucí plnění backlogu domluvit o dodržení detailnějších specifikací.

„Správné“ hodnoty backlogu nejdou odhadnout obecně a článek je taky neudává. BAM metodika je pouze nástroj pomocí kterého dokáže společnost nebo tým posoudit o aktuálním stavu a udělat kroky správným směrem.

Výstup hodnocení pro společnost Telenor Sweden, o které článek pojednává, byl např. použit pro vytvoření nejčastěji se objevujících se profilu položek (viz Obrázek 5: Výstupy analýzy – Nejčastěji objevující se profily položek). Zjistilo se, že položky, které se v perspektivě zralosti zařadili do kategorie „Nestabilní“, měli také nízký rozsah a úroveň abstrakce.

Metodika BAM bere za úkol zabraňovat situacím kdy se v backlogu mohou objevit moc abstraktní nebo příliš konkrétní položky. Při správném použití má tým, který s backlogem pracuje jasno jaká míra podrobnosti je ideální pro daný backlog, kolik času je potřeba pro případnou analýzu konkrétní položky a jak mají položky backlogu vypadat, aby proces selekce nebo mazání položek byl co nejrychlejší.

3.5 Zhodnocení backlogu

Posledním krokem je samotné zhodnocení backlogu. Každý z účastníku analytického týmu zařadí každou z náhodně vybraných položek právě do jedné kategorie z popsaných perspektiv v předchozím kroku.

Šablona, kterou každý ze skupiny pro hodnocení backlogu dostane může vypadat takto:

PERSPEKTIVA

ROZSAH	Izolovaná změna	Změna v podsystému	Změna v aplikaci	Změna v procesu	Změna platformy	
ÚROVEŇ ABSTRAKCE	Vize	Cíl	Systémová úroveň	Funkční úroveň	Komponentní úroveň	Lokální algoritmus
ZRALOST	Nestabilní	Denní změny	Změny každý týden	Změna jednou za několik týdnu	Každý měsíc	Stabilní
SPECIFIKACE	Bez specifikace	Popis jednou větou	Popis několika větami	Specifikace	Detailní specifikace	

Jednotlivé položky nejsou anonymizované a jak uvádí autoři metodiky, pokud se na některém z hodnocení členové týmu nedohodnou, je na místě diskuse o dané položce.

3.5.1 Příklad zhodnocení backlogu

V rámci této semestrální práce vyzkoušíme metodiku BAM na těchto pěti ukázkových položkách, které se vzali z reálného backlogu. Jednotlivé texty položek byly z důvodů důvěrnosti pozměněné.

1. Sepsání dokumentace
    a. Sepsání uživatelských manuálů a technické dokumentace projektu. Popis architektury.
2. Update balíčku
3. Přidání fulltextové filtrace do tabulky XXX
4. Lokalizace textů
    a. Kompletní lokalizace textu, popisků, textu emailů do jazyků češtiny a angličtiny. Text dané lokalizace by měla poskytovat vnější služba. Mělo by se zohlednit přidání dalších jazyků do budoucna.
5. Přidání platební metody
    a. Přidání platební brány PayPal.

Zhodnocení probíhalo v týmu 3 lidí, kteří se podíleli na psaní této semestrální práce, což do jisté míry značně ovlivňuje celkové hodnocení. Reálné hodnocení by měli provádět lidi z týmu, kteří backlogu a jednotlivým položkám rozumí a pracuji s nimi. Tento příklad je pouze ukázkový.

Osoba 1

PERSPEKTIVA

ROZSAH	Izolovaná změna **1 3**	Změna v podsystému	Změna v aplikaci **2**	Změna v procesu **4,5**	Změna platformy	
ÚROVEŇ ABSTRAKCE	Vize	Cíl	Systémová úroveň **1 2 4**	Funkční úroveň **3,5**	Komponentní úroveň	Lokální algoritmus
ZRALOST	Nestabilní	Denní změny	Změny každý týden **3**	Změna jednou za několik týdnu **1**	Každý měsíc **2**	Stabilní **4,5**
SPECIFIKACE	Bez specifikace **2**	Popis jednou větou **3**	Popis několika větami **1,5**	Specifikace **4**	Detailní specifikace	

Osoba 2

PERSPEKTIVA

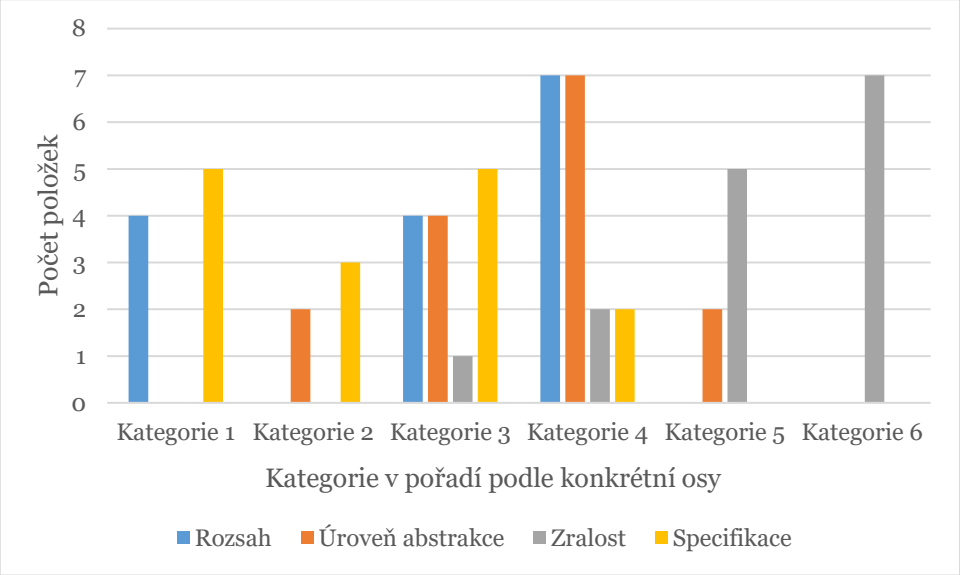
ROZSAH	Izolovaná změna **1**	Změna v podsystému **2, 5**	Změna v aplikaci **3, 4**	Změna v procesu,	Změna platformy	
ÚROVEŇ ABSTRAKCE	Vize	Cíl **4**	Systémová úroveň	Funkční úroveň **3, 5**	Komponentní úroveň **1, 2**	Lokální algoritmus
ZRALOST	Nestabilní	Denní změny	Změny každý týden	Změna jednou za několik týdnu **3**	Každý měsíc **1 2**	Stabilní **4,5**
SPECIFIKACE	Bez specifikace **2 3**	Popis jednou větou **5**	Popis několika větami **1**	Specifikace **4**	Detailní specifikace	

Osoba 3

PERSPEKTIVA

ROZSAH	Izolovaná změna **1**	Změna v podsystému **5**	Změna v aplikaci **2 3 4**	Změna v procesu,	Změna platformy	
ÚROVEŇ ABSTRAKCE	Vize	Cíl **4**	Systémová úroveň **2**	Funkční úroveň **3,5, 1**	Komponentní úroveň	Lokální algoritmus
ZRALOST	Nestabilní	Denní změny	Změny každý týden	Změna jednou za několik týdnu	Každý měsíc **1 2**	Stabilní **4,5 3**
SPECIFIKACE	Bez specifikace **2 3**	Popis jednou větou **5**	Popis několika větami **1 4**	Specifikace	Detailní specifikace	

Na základě testovacího zhodnocení backlogu a pěti zkušebních položek se vytvořil tento graf Ve výsledném grafu pro osu y představovala počet položek v dané kategorii a osa x konkrétní kategorii. Jednotlivé barvy pak představují perspektivy.



Graf 1: Výsledky ukázkového zhodnocení

Zhodnocení položek dá týmu přehledný obraz o stavu backlogu. BAM ovšem neříká, které kategorie jsou u položek správné, o tom si má udělat přehled již každý sám. V naší ukázce bychom například zjistili, že hned dvě položky nemají žádnou specifikaci. Účel takové položky „Update balíčku“ možná zná vývojář, který položku vytvořil, avšak není přenositelná na další osobou, který by mohla úkol vykonat místo autora.

Z grafu (viz. Graf 1: Výsledky ukázkového zhodnocení) lze také vyčíst rostoucí trend „Zralosti“. To značí, že položky jsou méně často měnící se. Tak jak jsou na začátku definovány, tak povětšinou zůstávají, dokud se nedokončí. To znamená, že v týmu je zanedbáván backlog refiniment a postupné detailizování položek. Rozsahem jsou položky povětšinou věnované aplikaci.

Backlog by měl mít ideálně rovnoměrnou distribuci úrovně abstrakce skrze položky. Avšak v našem případě velmi malého vzorku je docela jasné, že toho nelze dosáhnout.

Závěr

V první části semestrální práce jsme se podívali na tendence, které vedly ke vzniku backlogu. Popsali jsme backlog jako centrum komunikace a pravdy v agilních metodikách. Probrali jsme také hierarchické členění položek a jejich význam s příklady užití. Představeny byly i další typy backlogů než je produktový. Z metodiky SCRUM jsme zmínili ceremonie zahrnující backlog a jeho management v podobě „backlog refinement“.

V druhé části práce jsme se podívali na metodiky správného plnění. Byly zmíněny metodiky INVEST a DEEP vedoucí ke správné definici samotných položek tak, aby podporovali vývoj aplikace a zjednodušovali rozvoj a diskuzi nad backlogem. Z závěrečné kapitole této části byly zmíněny špatné praktiky týkající se nejen backlogu samotného, ale i vývojového procesu.

V poslední části jsme se zabývali metodikou Backlog Assessment Metodou. Byla popsána motivace pro vznik této metody a její rozsah. V další kapitole byly rozebrány jednotlivé dimenze náhledu na položky a jejich vysvětlení. V konečné fázy bylo na malém příkladu prezentován příklad hodnocení položek a informací, které je možno z takového hodnocení vyčíst.

V rámci týmu jsme se shodli, že metodika poskytuje užitečný náhled na stav backlogu a vyplňuje mezeru mezi dalšími nástroji, které se věnují plnění a vylepšení backlogu.

4 Citovaná literatura

Cohn, M., 2009. *Why There Should Not Be A Release Backlog*. [Online] Available at: https://www.mountaingoatsoftware.com/blog/why-there-should-not-be-a-release-backlog
[Přístup získán 17 11 2020].

Anon., 2020. *Backlog meaning in the Cambridge English Dictionary*. [Online] Available at: https://dictionary.cambridge.org/dictionary/english/backlog
[Přístup získán 5 Listopad 2020].

Anon., 2020. *What's the Difference: Product, Release, and Sprint Backlogs*. [Online] Available at: https://www.aha.io/roadmapping/guide/release-management/product-release-sprint-backlog
[Přístup získán 17 11 2020].

Arora, T. & Shigihalli, U., 2019. *Azure DevOps Server 2019 Cookbook*. 2nd editor místo neznámé:Packt.

Berntsson Svensson, R., Gorschek, . T., Bengtsson, P.-O. & Widerberg, J., 2019. *BAM - Backlog Assessment Method*. místo neznámé, Springer Open.

Ken, B., Grenning, J. & Mike, B., 2001. *Principy stojící za Agilním Manifestem*. [Online] Available at: https://agilemanifesto.org/iso/cs/principles.html
[Přístup získán 7 11 2020].

Larman, C., Deemer, B., Benefield, B. & Vodde, B., 2012. *Scrum Primer: A Lightweight Guide to the Theory and Practice of Scrum*. místo neznámé:autor neznámý

McDonal, K., 2018. *KBP Media*. [Online] Available at: https://www.kbp.media/themes-epics-features-user-stories/
[Přístup získán 8 11 2020].

Microsoft Corporation, 2020. *Define features and epics, organize backlog items - Azure Boards*. [Online] Available at: https://docs.microsoft.com/en-us/azure/devops/boards/backlogs/define-features-epics
[Přístup získán 8 11 2020].

microTOOL, 2020. *What are backlogs? – Knowledge base*. [Online] Available at: https://www.microtool.de/en/knowledge-base/what-are-backlogs/
[Přístup získán 11 17 2020].

Rubin, K., 2013. *Essential Scrum: A Practical Guide to the Most Popular Agile Process*. místo neznámé:autor neznámý

Schwaber, K. & Sutherland, J., 2017. *The Scrum Guide*. místo neznámé:autor neznámý

Sedano, T., Ralph, P. & Péraire, C., 2019. *The Product Backlog*. Montreal, autor neznámý

Sharma, G., 2019. *DEEP-DIVE into Product Backlog - INVEST in Stories - Create SMART Tasks*. [Online]
Available at: https://www.linkedin.com/pulse/deep-dive-product-backlog-invest-stories-create-smart-gaurav-sharma-1c/
[Přístup získán 13 Listopad 2020].

Wolpers, S., 2019. *Ideas on How to Improve Your Product Backlog Management Techniques*. [Online]
Available at: https://www.scrum.org/resources/blog/ideas-how-improve-your-product-backlog-management-techniques
[Přístup získán 9 Listopad 2020].