Semestrální práce ke kurzu 4IT421 Zlepšování procesů budování IS	
Semestr	ZS 2016/2017
Autoři – jméno, příjmení , xname	Patricie Benešová, xbenp44 Matej Hanzel, xhanm65 Hong Hoa Nguyen Thi, xnguh23 Aneta Musilová, xmusa00
Téma	Do Better Scrum
Datum odevzdání	18.12.2016

Abstrakt

Tato semestrální práce se věnuje agilní metodice Scrum, zejména možnostem jejího zlepšení. V první části jsou popsány základní principy agilního přístupu včetně porovnání s přístupem tradičním.

Následuje obecný popis metodiky Scrum – a to jak osob, které se na aplikaci metodiky podílejí, tak i artefaktů, které jsou během vývoje produktu vytvářeny, až k možnostem plánování projektu včetně plánování schůzek.

Další část práce se věnuje pojmu Velocity (rychlost) a jejímu využití při odhadech a plánování prací na projektu.

Těžiště této práce je v části, která se zabývá zlepšením provádění metodiky Scrum pomocí rozmanitých přístupů. Tato část také nabízí postupy, jak reagovat v krizových situacích, například při neplnění plánu, jak udržet tým stabilní a výkonný.

Poslední část se zabývá rolí Scrum Mastera, jakožto nepostradatelné osobnosti v tomto agilním přístupu vývoje software a v aplikaci zmíněných možnostech zlepšení aplikace Scrumu.

Klíčová slova

Scrum, agilní vývoj, scrum master, sprint, metodologie.

Obsah

1 Úvod

Oblast softwarového inženýrství se v posledních několika dekádách posunula (a stále posouvá) rychlým tempem kupředu. V devadesátých letech minulého století se začaly formovat přístupy k vývoji software, které se od těch stávajících (tradičních) zásadně lišily.

Mezi hlavní problémy tradičních metodik vývoje software lze zařadit složitost, malou flexibilitu a náročnou „byrokracii“ ve smyslu vytváření striktní dokumentace či revizí. Zejména kvůli poslednímu jmenovanému bodu nejsou tyto metodiky vhodné pro malé projekty a malé týmy.

Oproti tomu agilní metodiky kladou důraz především na rychlý vývoj v různě dlouhých cyklech, časté dodávky betaverzí, úzkou spolupráci s klientem a s tím spojenou vysokou flexibilitu či omezení „byrokracie“.

Semestrální práce se zabývá agilní metodikou Scrum. První část práce popisuje agilní principy a metodiku Scrum obecně. Druhá část práce se pak zabývá tím, jak metodiku vylepšit tak, aby týmy pracovaly efektivněji a dosahovaly kýženého výsledku v co nejkratším čase. Cílem práce je tedy popsat tyto metody a říci, jakým způsobem je praktikovat.

2 Principy agility

Tradiční metodiky vývoje software počítají s tím, že požadavky na vyvíjený produkt jsou stálé a nebudou se měnit. To je ale zřídkakdy skutečností. V krajním případě se tedy může stát, že zákazníkovi bude sice předán hotový produkt, který ale nevyhovuje jeho aktuálním požadavkům.

V agilním přístupu se toto stát nemůže. Díky častým konzultacím se zákazníkem a průběžnému dodávání betaverzí jsou nové požadavky zapracovávány průběžně a předávány v dalších verzích software.

Z tohoto vychází jeden z nejdůležitějších principů agilního programování, který říká, že jedinou cestou, jak prověřit správnost navrženého systému, je co nejrychlejší vývoj, předložení zákazníkovi a následná úprava podle jeho zpětné vazby.

Základní principy agilního přístupu k vývoji software tedy patří:

- Inkrementální vývoj s krátkými iteracemi – zákazník má možnost vývoj sledovat a připomínkovat, změny jsou následně zapracovány.
- Komunikace mezi vývojovým týmem a zákazníkem – v ideálním případě je zákazník přímo součástí vývojového týmu.
- Průběžné automatizované testování – testy by měly být sestavené ještě před samotnou implementací testované části, vždy by měla být aplikována kompletní sada testů, aby byla zajištěna integrace jednotlivých částí vyvíjeného software.

2.1 Manifest agilního programování

Vznik dokumentu Manifest agilního programování se datuje k únoru 2001 a byl sepsán odborníky z oblasti vývoje software v americkém státě Utah. Popisuje základní pilíře agilního programování nezávisle na konkrétní metodice.

Manifest staví na dvou základních principech:

- Umožnit změnu je efektivnější než se jí snažit zabránit.

- Je potřeba umět se vypořádat s nepředvídatelnými událostmi, protože nepochybně nastanou.

Na základě těchto principů dávají autoři přednost:

- individualitám a interakci před procesy a nástroji,
- fungujícímu software před obsáhlou dokumentací,
- spolupráci se zákazníkem před sjednáváním smluv,
- reakci na změnu před plněním plánu.

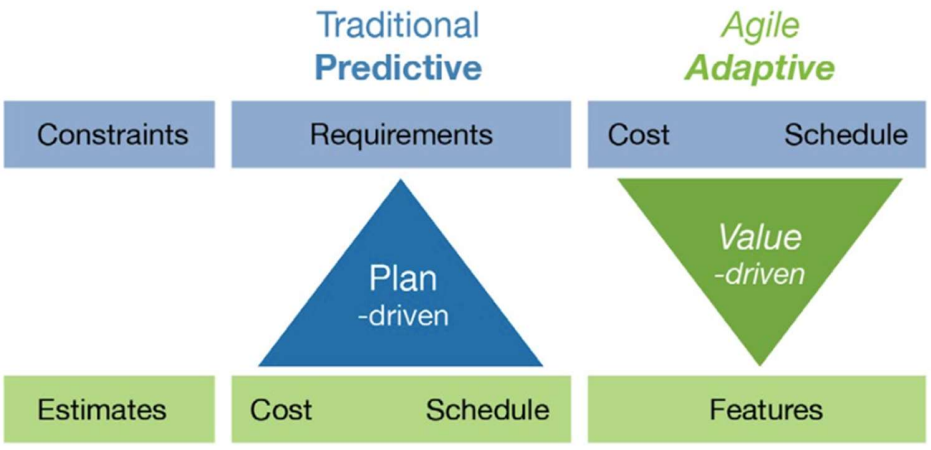
Žádná metodika není striktně daná, vždy je prostor pro její přizpůsobení konkrétnímu projektu. Jinak by daná metodika nemohla být považována za agilní.

3 Scrum

Scrum patří mezi agilní metodiky a její základní charakteristikou je dodávaní fungujících inkrementů zákazníkovi v pravidelných intervalech. Snahou Scrumu není říct, jak vykonávat práci na projektu, o což se ani nepokouší, předpokládá se, že týmy dělají všechno potřebné pro to, aby doručili žádaný produkt, a právě Scrum jim vytvoří vhodné podmínky a prostředí, aby tak mohli učinit.

Vznik Scrumu byl podnícen problémy, které se objevovali při řízení projektů vývoje software (Hundermark, 2015) kde šlo zejména o otázky příliš dlouhého uvolnění, příliš dlouhé doby stabilizace, složitého vykonávaní změn v průběhu projektu, klesající kvality výstupů, a šlo také o skutečnost, že neúspěchy kazí pracovní morálku.

Hlavním problémem bylo, že používané metody kladly vysoký důraz na detailně definovanou specifikaci vstupů, a metody přeměňující tyto vstupy na výstupy byly předem známy, jinými slovy, bylo je možné předpovědět, což pro vývoj softwaru není ideální a dalo by se uvažovat až o jeho nepoužitelnosti (Hundermark, 2015). Porovnání tradičního (prediktivního) a agilního (adaptivního) přístupu je zobrazeno na obrázku č 1.



Obrázek 1: **Porovnání tradičního a agilního (adaptivního) přístupu.** (Zdroj: Hundermark, 2015)

Z výše uvedeného a z obrázku se dá odvodit, že pro Scrum je charakteristické (Hundermark, 2015):

- Týmu je zadán jasný cíl.
- Tým je samo-organizující se.
- Tým pravidelně dodává inkrementy.
- Tým dostává zpětnou vazbu z okolí.
- Práce týmu směruje k neustálému zlepšování se.
- Organizace má přehled o progresivitě práce v týmu.
- Tým a management komunikují o pokrocích práce a rizikách.

3.1 Scrum tým

Model Scrum týmu je navržen tak, aby umožnil optimalizovat flexibilitu, kreativitu a produktivitu týmu (Schwaber, Sutherland, 2013). Týmy jsou tvořené výhradně třemi rolemi:

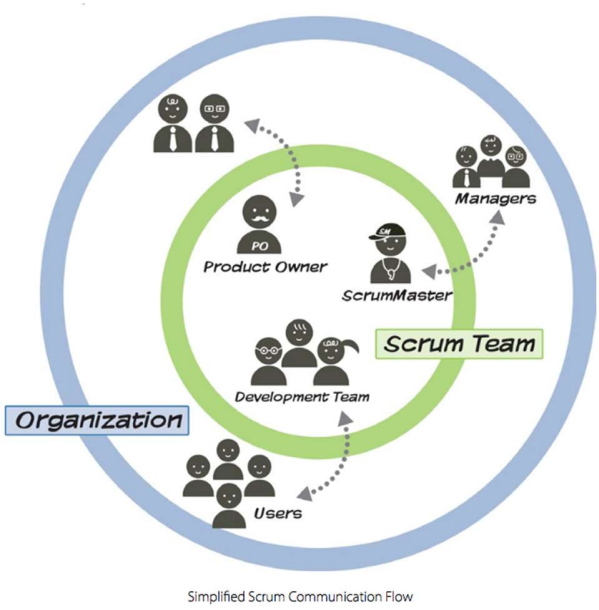
- Vlastník produktu, který má na starosti management produktu a ROI.
- Vývojový tým, tvořen 3-9 osobami.
- Role Scrum Mastera, který je zodpovědný za management procesů.

Výše zmíněná samo-organizace týmu se nedá ztotožnit s čistým laissez-faire přístupem. Přestože jsou stanoveny meze volnosti, v rámci kterých mají týmy jistou autonomii, Scrum týmy se vyznačují vysokou disciplínou.

Mimo Scrum tým se vyskytují také (Hundermark, 2015):

- Zákazníci, tedy ti, kteří projekt financují (známí též jako sponzoři), komunikují převážně s vlastníkem produktu.
- Uživatelé, ti, kteří budou využívat výstupy Scrum týmu, komunikují hlavně s vývojovým týmem, poskytují mu přímou zpětnou vazbu.
- Manažeři, kteří poskytují organizační rámec, ve kterém Scrum tým působí, je tady úzká komunikace se Scrum Masterem, který jim pomáhá v otázkách, jak nejlépe podpořit Scrum tým v jeho efektivitě.

Celkové schéma znázorňující prostředí a vztahy jednotlivých rolí je znázorněno na obrázku č. 2.



Obrázek 2: **Role a vztahy ve Scrumu a jeho okolí.** (Zdroj: Hundermark, 2015)

3.1.1 Vlastník produktu

Primární úlohou vlastníka produktu je zaměřit se na efektivitu, se kterou je produkt vytvářen, a to, že jde o ten správný produkt pro jeho zákazníky. Dále je odpovědný za s tím spojenou optimalizaci ROI (ziskovost investicí), má na starosti zajištění existence společné vize produktu, správu produktového Backlogu a v neposlední řadě má na starosti schvalování dodaných inkrementů produktu na konci každé iterace. Odpovídá také na otázky členů týmu týkajících se toho, co se bude vytvářet a proč (Hundermark, 2015).

3.1.2 Vývojový tým

Vývojový tým se skládá z profesionálů podílejících se na tvorbě výsledného produktu. Je zodpovědný za dodání inkrementů na konci každého Sprintu (Schwaber, Sutherland, 2013). Členové jsou programátoři, testeři, analytici, architekti, designéři, ale i uživatelé, jednoduše řečeno každý, kdo se podílí na práci a vytváří ten správný produkt pro jeho majitele (vlastník produktu) a to vše při zachování efektivity (Hundermark, 2015).

Vývojový tým, je jako takový samo-organizující se a to ve smyslu toho, že mu není řečeno, a to ani ze strany Scrum Mastera, jaký způsobem (jak) má dojít k proměně produktového Backlogu do uvolnitelného inkrementu. Členové vývojového týmu tedy nejsou řízeni žádným nadřízeným a na následných postupech by se měli domlouvat všichni členové. Z toho je dále patrné, že ve vývojovém týmu neexistují žádné další sub-týmy zaměřené na dílčí úlohy typu testování, byznys analýza, nýbrž jsou členové týmu cross-functional se všemi schopnostmi (skills) potřebnými k vytvoření produktu (Schwaber, Sutherland, 2013).

3.1.3 Scrum Master

Scrum Master řídí celý proces prací v týmů. Vzhledem k tomu, že Scrum tým je samo-organizovaný, dochází k tomu prostřednictvím vlivu spíše než autoritou Scrum Mastera. Je to

proto, že Scrum Master není v roli lídra, neměl by tedy nikomu nic přikazovat, ale spíše podporovat rozvoj týmu (Hundermark, 2015).

Scrum Master zajišťuje to, že metodice Scrum všichni v týmu správně rozumí, řídí se jí a dodržují její praktiky (Schwaber, Sutherland, 2013). Vede Scrum tým k samo-organizování, řeší možné spory a překážky, které brání vývojovému týmu pracovat na produktu, dále pomáhá produktovému vlastníkovi pochopit a plnit jeho roli správně, a v neposlední řadě napomáhá začlenit Scrum metodiku vně organizace, ve které je metodika užívaná (Hundermark, 2015).

3.2 Scrum schůzky (Scrum Events)

3.2.1 Sprint

Sprint představuje základní jednotku metodiky Scrum a je si ho možno představit jako kontejner pro všechny ostatní schůzky (kapitoly 3.2.3-3.2.6). Účelem Sprintu je splnit takzvaný Sprint cíl. Toho je dosaženo plánováním a dodáváním položek produktového backlogu produktovému vlastníkovi. V případě, že Scrum tým zjistí kdykoliv během Sprintu, že stanovený cíl již není dosažitelný nebo už nebude sloužit potřebám organizace, produktový vlastník je oprávněn okamžitě ukončit Sprint (Hundermark, 2015).

Scrum týmy si volí 1, 2, 3 nebo 4 týdenní trvání Sprintu. Délka je fixní a není ji možné prodlužovat či jakkoli měnit a to v okamžiku, kdy Sprint již začal. Tím je zabezpečen rytmus a jistý spád pro vykonávaní prací v týmů (Hundermark, 2015).

3.2.2 Backlog Odhadování (Backlog Grooming)

Cílem schůzky Backlog odhadování je zajistit, aby tým rozuměl celému Backlogu, který je složen z položek charakterizujících určitou funkcionalitu, respektive skupinu funkcionalit definovaných případů, též známých jako user stories. Pro správné a vyvážené naplánovaní je potřebné jednotlivé user stories ohodnotit tak, že bude vyjádřená jejich priorita takzvanými story points. Další úlohou story points je to, že se tým shodne na tom, že daný případ chápou stejně. Metodou, kterou je možné odhadnout jednotlivé případy, je například Planning Poker.

Backlog odhadování je možné vykonávat v půlce Sprintu tak, aby byl dostatek času na seznámení se s danou funkcionalitou jak ze strany vlastníka produktu tak vývojového týmu (Šochová, 2014).

3.2.3 Naplánování Sprintu (Sprint Planning)

Každý Sprint začíná jeho naplánováním. Smyslem této schůzky je pro Scrum tým plánovat práci, kterou bude mít na starosti vyřídit během tohoto Sprintu. Samotné plánování je obecně rozděleno do dvou částí (Hundermark, 2015).

První část má odpovědět na otázku "Co můžeme dodat na konci tohoto Sprintu?". Jde o skutečně detailní "workshop požadavků". Je zde přítomen vlastník produktu a to hlavně proto, aby navedl vývojový tým na správný směr a i proto, aby odpovídal na otázky jeho členů. Tým může taktéž v této části využít znalosti z jejich rychlosti z minulosti jako prediktor (Yesterday‘s weather). Druhá část je označovaná jako workshop "design" a odpovídá na otázku "Jak budeme dělat práci na projektu"(Hundermark, 2015).

Výsledkem plánování Sprintu je Sprint Backlog, jehož obsah je seznam úloh a plán, který musí být týmem plněn. Sprint Backlog má nejčastěji podobu fyzické tabule, respektive může jít o jednoduchý softwarový nástroj zobrazující jednotlivé úkoly vzhledem ke stanovenému plánu (Hundermark, 2015).

3.2.4 Denní Scrum (Stand-up)

Během denního Scrumu se členové vývojového týmů setkávají, a to s cílem odkomunikovat a synchronizovat práce na projektu a vytvořit tak plán na následujících 24 hodin. Tato schůzka trvá maximálně 15 minut. Protože vývojový tým je samo-organizující se, tento typ spolupráce je esenciální pro zabezpečení kontinuálního pokroku v práci bez přestávky a vyhnutí se tak možnému vytvoření překážek v práci některého člena vývojového týmu (Hundermark, 2015).

3.2.5 Sprint Demo

Těžištěm Sprint Demo je produkt, respektive jeho nasaditelná část, která byla vývojovým týmem vytvořena. Jde tedy o ukázku nových dokončených částí během Sprintu. Primárním účelem této schůzky je kontrola, co vývojový tým dodal a získání zpětné vazby od zákazníka. Tím dochází k přizpůsobení plánu (produktový backlog) pro následující Sprint. Sprint Dema se účastní vývojový tým, produktový vlastník, ScrumMaster a zákazník (uživatel) a dochází k němu na konci každého Sprintu. V ideálním případě dochází ke stanovení cíle pro následující Sprint (Hundermark, 2015).

3.2.6 Restrospektiva

Závěreční schůzkou Sprintu je retrospektiva. Následuje okamžitě po Sprint Demu a není ji možno nikdy vynechat. Na rozdíl od Sprint Dema, který je zaměřen na produkt, retrospektiva je zaměřena na proces. Jde o způsob, jakým členové Scrum týmu spolupracují, včetně jejich zručností, použitých praktik a nástrojů. Účelem retrospektivy je neustálé zlepšování způsobu práce (Hundermark, 2015).

3.3 Scrum Artefakty

3.3.1 Produktový Backlog

Produktový backlog je seznam položek představujících práci, kterou je potřebné vykonat v určitém čase. Jednotlivé položky jsou popsány na funkční úrovni a může je do produktového backlogu přidávat kdokoliv, ale jedině vlastník produktu určuje v jakém pořadí budou vývojovým týmem vykonávány (Hundermark, 2015).

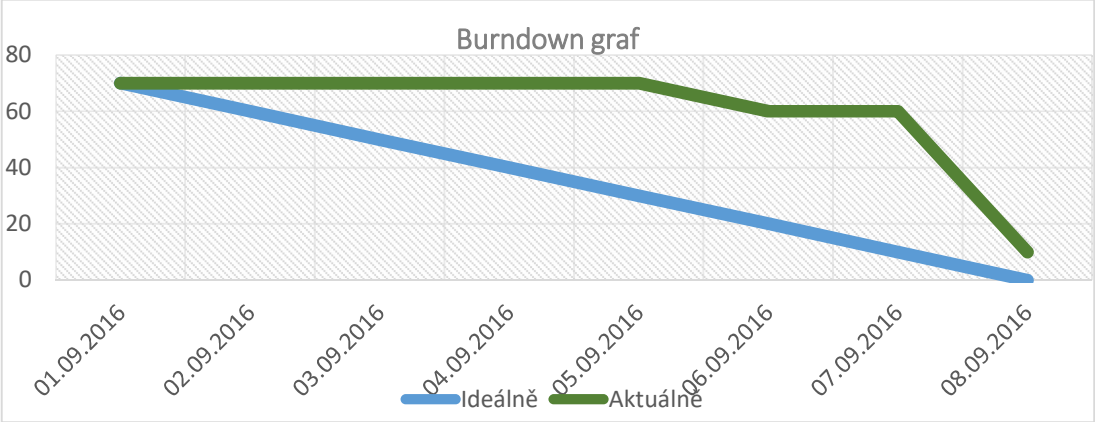
Základní vlastností produktového backlogu je, že jde o žijící dokument. Jednotlivé požadavky v podobě položek produktového backlogu časem (s každým Sprintem) vznikají, upravuje se jejich úroveň detailu, zpřesňují se, jiné naopak zanikají. Jiný typ změn se může týkat úpravy vztahu mezi hodnotou, časem a náklady (Hundermark, 2015).

3.3.2 Sprint Backlog

Sprint backlog představuje seznam položek z produktového backlogu, které jsou plánované pro konkrétní Sprint (Schwaber, Sutherland, 2013). Většina vývojových týmů volí způsob vizualizace pomocí tabule, známé z metodiky Kanban. Je zde jasně vidět, jaká práce je naplánována na Sprint a její aktuální stav, ve kterém se daná úloha právě nachází (Hundermark, 2015).

3.3.3 Burndown a Burnup graf

Volitelným artefaktem v metodice Scrum je Burndown nebo Burnup graf. Jde o vizuální zobrazení toho, jak celkově probíhají práce na produktu vyjádřeny prostřednictvím story points v čase. Ideálním případem je, když jsou inkrementy dodávány postupně a rovnoměrně rozložené v čase. Často se však stává, že na začátku má graf neměnnou tendenci a tudíž, že s časem není dodávaná žádná funkcionalita, není hotov žádný ze stanovených user stories. Příklad je zobrazen na obrázku č. 3.



Obrázek 3: **Porovnání burndown grafů.** (Zdroj: vlastní zpracování)

4 Rychlost (Velocity)

Rychlost je známa spíše pod anglickým názvem Velocity. Dle Oliveiry (2014) rychlostí myslíme rychlost týmu neboli za jak dlouho dokáže tým dokončit user story ohodnocenou story points. Pokud tedy tým dokončí na konci Sprintu dvě user story každý o osmi story points, rychlost týmu je 16. Průměrná rychlost se spočítá dle následujícího vzorce:

$$\frac{součet\ dokončených\ story\ points}{počet\ dokončených\ Sprintů}$$

Pokud pak v dalším Sprintu tým dokončí tři user story dohromady za 20 story points, průměrná rychlost (velocity) by se spočítala jako součet 16 a 20 děleno 2. Výsledná průměrná rychlost týmu by tedy byla 18.

Dle Scrum Inc (2016) existují dva hlavní důvody, proč rychlost sledovat. To, kolik story points je dokončeno v průběhu Sprintu, pomáhá týmu zjistit, zda změny, které dělají, mají nějaký efekt na produktivitu týmu. V tomto ohledu slouží i vrcholovému vedení firmy k hodnocení jednotlivých týmu a sledování, jak se týmům vede.

Dalším důvodem je lepší odhad počtu story points, ke kterým se členové týmu zaváží při plánování Sprintu. Pro lepší odhady je vhodné dokončit úspěšně alespoň tři Sprinty a z nich spočítat průměrnou rychlost. To, za kolik předešlých Sprintů budeme měřit rychlost, pak záleží na konkrétní firmě. Pro vlastníka produktu (Product owner) tak může rychlost týmu značit, za jak dlouho je tým schopen dokončit všechny user story v backlogu.

5 Možnosti zlepšení aplikace Scrumu

V následující kapitole jsou představeny jednotlivé možnosti, jak vylepšit Scrum. Včetně toho je smyslem této kapitoly, aby inspirovala týmy k hledání nových způsobů práce, které vedou k vyšší kvalitě, rychlejší dodávce, a také aby týmy měly při práci větší radost.

5.1 Stabilní týmy (Stable Teams)

Udržujte týmy stabilní. Nebude-li to nezbytně nutné, neposílejte členy jednoho týmu do druhého. Působení v jednom týmu posiluje týmového ducha, takové týmy mají přehled o svých schopnostech a znalostech, díky čemuž mohou nejen navrhovat kvalitní řešení, ale také lépe předpovídat trvání a směr projektu.

Vítaným plusem stabilních týmů je také to, že znají svou rychlost (velocity). Rychlost nových týmů značně kolísá a na začátku je takřka neznámá.

Jak mají vypadat stabilní týmy? (Waters, 2011)

- Zaměřit se na produkt – soustředit se na určitý výrobek na dobu neurčitou umožní týmu o produktu získat skutečné poznatky, díky tomu při každém novém spuštění bude mít stále k dispozici nabyté *know-how*.
- Jedna obchodní oblast – každý tým by měl mít přiřazenou pouze jednu obchodní oblast, kterou by zastřešoval. Díky tomu tým získá jasnou představu o prioritách a lépe porozumí obchodním cílům dané oblasti. Na těchto stavebních kamenech pak tým může vybudovat silné obchodní vztahy.
- Multi-disciplinovaný – každý tým by měl za ideálních podmínek mít všechny schopnosti, které jsou potřebné k dodání projektu a to od jeho prvotních myšlenek až po samotné uvolnění produktu koncovým uživatelům.
- Na jednom místě – ačkoliv to není vždy možné nebo ekonomicky výhodné, členové týmu by měli být u sebe. Nejenže by měli být ve stejném městě nebo budově, ale měli by přímo sedět ve stejném prostoru, kde spolu mohou komunikovat tváří v tvář tak často, jak je to jen potřebné.

5.2 Jako včerejší počasí (Yesterday's Weather)

Týmy často slibují něco, co nezvládnou vyplnit. Slib dávají pokaždé, třebaže je jasné, že se bude opakovat ten samý scénář: opět slib nedodrží – nedodají slibovaných sedm úkolů, ale například pouze pět. Jak se přestat točit v kruhu? Podle Sutherlanda (nedatováno) do dalšího Sprint Backlogu zařaďte přesně tolik úkolů, kolik se stihlo udělat v předchozím Sprintu.

Ve většině případů je počet odhadovaných úkolů dokončených v posledním Sprintu nejspolehlivějším ukazatelem toho, kolik bude dokončeno úkolů v příštím Sprintu.

Jako včerejší počasí (Yesterday's Weather) umožňuje týmům budovat přesnější Sprint backlog, omezuje týmy ve stanovování svých cílů – vede je k tomu, aby si nekladly zbytečně vysoké cíle, které mohou ohrozit projekt. (Sutherland, nedatováno)

5.3 Rojení (Swarming)

Swarming funguje podobně jako párové programování, kdy několik členů týmu společně pracují na jednom úkolu, aby byl včas dokončen. Obecně, týmy potřebují projít fází formování, během které se vzájemně poznávají, poté fází bouře, během které dochází ke vzniku konfliktů v týmu a jejich řešení, než začnou vyvíjet a být fungující jednotkou. Dejte proto každému členu týmu dostatek prostoru, aby se mohl realizovat. (Linders, 2013)

Ideou této praktiky je propojit vývojáře, aby se společně podíleli na jednom úkolu a nahradit tak scénář samotářských vlků.

5.4 Šablona přerušení (Interrupt Pattern)

Během jakéhokoliv úkonu se lze setkat s nečekanou událostí, která způsobí změnu původního plánu, je proto důležité myslet na rezervu. Ve Scrum metodice se jedná o Šablonu přerušení (Interrupt Pattern), která vede týmy k tomu, aby si předem definovaly časovou rezervu pro nečekané úkoly. Tato prevence má sloužit k tomu, aby nedocházelo k překročení časového limitu projektu.

Má tři jednoduchá pravidla, která zajistí, že v týmu nedojde k narušení vývoje (Hundermark, 2015):

- Tým si vytvoří rezervu pro neočekávané položky na základě historických dat. Například 30 % týmové práce je způsobeno v průměru neplánovanými úkoly, které se nečekaně objevily při sprintu. Pokud se pohybuje rychlost (velocity) týmu okolo 60 story points, pak 20 story points je vyhrazeno pro rezervu neočekávaných položek.
- Všechny žádosti musí projít rukama vlastníka produktu, aby je roztřídil. Vlastník produktu některým žádostem může přidělit nízkou prioritu, pokud je zřejmé, že neovlivní podnikatelský plán. Naopak na mnoho jiných žádostí se může dostat až při dalším Sprintu, třebaže mohou zvednout hodnotu podnikatelského plánu již teď. Některé žádosti jsou kritické a ty je potřeba vyřešit v právě probíhajícím Sprintu. Vlastník produktu takové žádosti zařadí do rezervy pro neočekávané položky.
- V případě, že rezerva začne přetékat, např. když vlastník produktu dá více bodů, než předtím ve Sprintu přiřadil, tým musí zrušit činnost, Sprint přeplánovat a management upozornit, že data dodávky budou posunuta.

5.5 Čistý kód na denním pořádku (Daily Clean Code)

Čistým kódem se rozumí kód, který je srozumitelný a udržovatelný, tyto faktory lze měřit např. časem, který vývojář potřebuje k přečtení cizího kódu a jeho porozumění – to je vůbec první krok, nežli se sám pustí do úprav. Jestliže je vývojář při čtení kódu ve stresu a v úzkostech, lze téměř s jistotou prohlásit, že je kód těžko pochopitelný a měl by projít revizí (za podmínky, že kód čte seniorní vývojář). (Techdev, 2015)

Mimo tyto faktory by kód měl být také bez chyb. Pokud se chyby identifikují, opravujte je včas, ideálně do 24 hodin, abyste na konci každého dne měli přeložený a plně fungující zdrojový kód. Berte to jako nácvik pro udržení pořádku, který běžně praktikujete i doma – jakmile vidíte ve vlastní domácnosti hmyz, neváháte a okamžitě jej likvidujete, pokuste se tuto praktiku zavést i v pracovním prostředí.

Jestliže chyba po úspěšné identifikaci nebyla okamžitě odstraněna, přiřaďte ji do Sprint Backlogu a vyřešte ji později v ten samý den nebo den následující. To umožní vývojářům dokončit aktuální úlohy bez přerušení.

Ačkoliv je Scrum v oblasti vývojářských praktik velmi zdrželivý, disciplína, dodržování štábní kultury ve zdrojovém kódu a správa kvality designu je klíčovým podnětem vedoucí k úspěchu.

5.6 Nouzový plán (Emergency Procedure)

> Někdy se i přes veškeré úsilí mohou plány rozpadat jako domeček z karet. Pokud se stane, že narazíte na hrozbu, která se s vysokou pravděpodobností může stát skutečností, najeďte na nouzový plán, který byl výhradně navržen pro tuto situaci. Nezdržujte se pátráním po příčině problému a nezabývejte se zbytečně tím, co se okolo vás děje, nouzový plán vám pomůže. (Hundermark, 2015)
>
> Podle Hundermarka (2015) je úlohou Scrum Mastera, aby se ujistil, že se jeho tým bude okamžitě v takových situacích řídit pokynů nouzového plánu.

Kroky nouzového plánu (Hundermark, 2015):

- Změňte způsob, kterou je práce dokončena, udělejte cokoliv jiného.
- Hledejte pomoc – častým řešením bývá najít pomoc zvenčí týmu.
- Snižte rozsah projektu – vyřaďte funkce s nízkou prioritou a přeplánujte projekt tak, aby byly dodány ty oblasti či funkce, které tým může aktuálně zvládnout. Tým nemusí dodat všechno a nemusí být příliš pozdě, aby byly splněny cíle Sprintu. Stále je možné skončit v přijatelném stavu.
- Ukončete běh – v případě, že se nedaří požár uhasit, pak bude nejlepší Sprint přerušit a jednoduše začít znovu. Ne každý Sprint lze uběhnout až do cílové roviny, i to, že se vzdáte a poučíte, je skvělý výsledek, díky kterému může být příští Sprint dotažen do konce.
- Informujte vedení, jak budou ovlivněna data spuštění projektu.

5.7 Používání Scrumu ke zlepšení Scrumu (Scrumming the Scrum)

Podle autorů článku *Teams that Finish Early Accelerate Faster* (2014) další možností, jak zlepšit úspěšnost týmu používající metodiku Scrum, je identifikovat tu nejdůležitější překážku z předchozího Sprintu nejčastěji během Retrospektivy a v dalším Sprintu ji odstranit.

Autoři navrhují vytvořit z této překážky user story i s akceptačními testy a vložit ho do backlogu Sprintu. Výsledek pak mohou členové týmu prezentovat na Sprint Demo prezentaci (podobně jako u běžného dokončeného user story).

Týmy tak mohou v každém Sprintu odstranit další a další překážky. Tyto týmy mají vyšší šanci, že dokončí všechny „upsané“ story points včas nebo dříve, než je konec Sprintu. Pokud tým vybere překážku s největší prioritou, projeví se zlepšení i v ukazateli rychlosti týmu (viz kapitola 4).

5.8 Metrika spokojenosti (Happiness Metric)

To, jak se členové týmu cítí, je při zlepšování provádění metodiky Scrum dle autorů *Teams that Finish Early Accelerate Faster* (2014) jednou z nejlepších metrik. Pokud jsou lidé nešťastní

či frustrovaní, projeví se to i v jejich práci. Tento pocit může přijít například, pokud mají pracovníci pocit, že má firma problémy nebo existuje nějaká velká překážka, se kterou se musí nějak vypořádat.

Zavedení a praktikování této metody provádí Scrum Master. Jeho úkolem je monitorovat spokojenost týmu. K tomu mu poslouží dvě otázky:

- Jak spokojení jste s firmou?
- Jak spokojení jste se svou rolí?

Členové týmu ohodnotí své pocity na škále od jedné do pěti. Odpovědi si Scrum Master zaznamená a uchová. Může tak předpovědět pokles v rychlosti týmu (viz kapitola 4) a může provést určité kroky k nápravě. Podle Jeffa Sutherlanda (2012) je vhodné tuto metodu kombinovat s metodou „Používání Scrumu ke zlepšení Scrumu“ (viz předchozí podkapitola). Zjištěný důvod nespokojenosti může tým pak dle této metody zahrnout jako položku ve Sprint backlogu a brát ji jako překážku, která se musí odstranit.

Pocit spokojenosti může také působit jako prevence tzv. syndromu vyhoření, které se u některých osob objevuje v případech, kdy mají dlouhou pracovní dobu, po kterou se musí mentálně soustředit na práci. Metrika spokojenosti pak může vést ke snížení počtu hodin pracovní doby. To se samozřejmě nemusí líbit vedení firmy, ovšem je důležité mít na paměti, že agilní metodiky jsou založeny na důvěře ve členy týmu.

5.9 Týmy, co končí brzy, udělají více práce (Teams That Finish Early, Accelerate Faster)

Častým problémem týmu provádějících Scrum je, že se „upíší“ ke většímu množství práce, než jsou schopni vykonat v rámci jednoho Sprintu. Dle autorů článku *Teams that Finish Early Accelerate Faster* (2014) takovéto selhání pak může zabraňovat týmu ve zlepšování.

Týmy, které dokončují práci pozdě, frustrují vlastníka produktu (Product owner), který pak tlačí na přijetí více práce do dalšího Sprintu. To ovšem nejsou členové týmu schopni splnit a celý proces se opakuje.

Autoři článku radí využít hned několik již dříve zmíněných metod pro odstranění překážek, které mohou vést k nucenému přerušování práce. Dále je také vhodné použít metodu Včerejšího počasí (Yesterday’s Weather), která může týmu pomoci s odhadem práce na další Sprint. Jeff Sutherland (nedatováno) také radí zavazovat se raději k menšímu množství práce.

6 Vliv Scrum Mastera

Scrum Master je naprosto zásadní postava při praktikování metodiky Scrum. Je zodpovědný za to, že tým provádí Scrum tak, jak má, a respektuje metody, hodnoty a praktiky metodiky Scrum. (Moreira aj, 2010)

Dále komunikuje s vývojáři a managementem a částečně „chrání“ tým před tím, aby management nekladl týmu nějaké překážky zabraňující plynulému a produktivnímu vývoji. Zároveň pokud se nějaké překážka vyskytne, úkolem Scrum Mastera je ji odstranit nebo alespoň napomoci jejímu odstranění. (Moreira aj, 2010)

Zúčastňuje se všech schůzek, které Scrum metodika předepisuje, a dohlíží na jejich správný chod. Primárně se zaměřuje na denní schůzky, které o týmu mohou leccos prozradit.

(Moreira aj, 2010) Také vede Sprint Restrospektivy, na kterých se snaží prosazovat výše uvedené metody (pokud je jich třeba) a snaží se tým motivovat k jejich používání.

Z uvedeného vyplývá, že hlavní prací Scrum Mastera je komunikace s lidmi, a proto by měl Scrum Master mít adekvátní odborné znalosti, pozitivní přístup k práci, vysokou sociální inteligenci a vyvinuté „soft skills“. Zásadně ovlivňuje výsledný produkt i spokojenost zákazníka.

7 Závěr

Cílem semestrální práce bylo nejdříve popsat vlastnosti a specifika, ale především možnosti zlepšení agilní metodiky Scrum. Na úvod byly popsány zásady agilního programování, další části se zabývají detailním popisem metodiky Scrum i jejich specifik, problémů a způsobů vylepšení včetně jejich uvedení do praxe.

Pro kvalitní použití metodiky Scrum je důležité, aby všichni členové týmu pochopili principy fungování Scrumu, a také je dodržovali. To platí samozřejmě i pro možnosti zlepšení. Každý člen týmu musí chápat, jakou roli v týmu zastává a jaké jsou jeho povinnosti. Musí také respektovat role ostatních členů týmu.

Jedním z nejvážnějších problémů lidí, kteří se dříve nesetkali s agilním vývojem (potažmo Scrumem), je nerespektování základních principů i konkrétních úkonů metodik. Například vývojáři se vyhýbají popisování user story, nechtějí debatovat o odvedené práci či zaujímají odmítavý postoj k retrospektivním schůzkám.

Pro tým, který se se Scrumem setkává poprvé, je přínosné uspořádat školení, kde bude vysvětleno nejen co a jak funguje, ale hlavně proč to tak funguje a jak je to prospěšné pro fungování týmu.

Ohledně dokumentace a plánování menších projektů lze říci, že pro tvorbu Product Backlog, Sprint Backlog, plánování a přehled Sprintu či user story lze úspěšně použít reálnou tabuli nebo jiné nástroj s minimální funkcionalitou.

Pro metodiku Scrum je stěžejní rovnováha mezi rozsahem vyvinutých funkcionalit, kvalitou jejich zpracování a časem dodání. Taktéž by nikdy nemělo být opomíjeno testování.

Ideální fungování metodiky Scrum nastává v případě, že Product Owner pouze navrhuje user stories, kterým také přiřazuje prioritu. Kompetence výběru úkolů z User Stories, které budou splněny v následujícím Sprintu, je čistě na vývojovém týmu.

Pro efektivní aplikaci některé možnosti zlepšení Scrumu je ve většině případů zásadní pozice Scrum Mastera, který tyto postupy řídí, případně dohlíží na jejich průběh a dodržování. Díky tomu mohou práce na projektu postupovat plynule a podle plánu, případné potíže při vývoji lze bez větších problémů eliminovat.

8 Literatura

HARRISON, Neil, Joel RIDDLE a Jeff SUTHERLAND, 2014. *Teams that Finish Early Accelerate Faster: A Pattern Language for High Performing Scrum Teams*. 47th Hawaii International Conference on System Science. ISBN 978-1-4799-2504-9

HUNDERMARK, Peter, 2015. *Do Better Scrum: An unofficial set of tips and insights into how to implement Scrum well*. Cape Town.

LINDERS, Ben, 2013. *How Swarming Helps Agile Teams to Deliver*. In: [online]. [cit. 2016-10-15]. Dostupné z: https://www.infoq.com/news/2013/02/swarming-agile-teams-deliver

MOREIRA, Mario E., Michael LESTER a Steve HOLZNER. 2010. *Agile for DUMMIES: CA Technologies Edition* [online]. USA: Wiley Publishing. [cit. 2016-12-05]. ISBN 78-0-470-87693-0.

OLIVEIRA, Catia, 2014. *Velocity: How to Calculate and Use Velocity to Help Your Team and Your Projects*. In: Scrum Alliance [online]. Feb 6, 2014 [cit. 2016-10-15]. Dostupné z: <https://www.scrumalliance.org/community/articles/2014/february/velocity>.

SCRUMINC, 2016. *Velocity*. In: ScrumInc [online]. [cit. 2016-10-21]. Dostupné z: <https://www.scruminc.com/velocity/>.

SCHWABER, Ken, SUTHERLAND Jeff, 2013. *The Scrum Guide*. [online]. [cit. 2016-10-21] Dostupné z: <http://www.scrumguides.org/docs/scrumguide/v1/scrum-guide-us.pdf>.

SUTHERLAND, Jeff, 2012. *Exciting discussions with Jeff Sutherland, the inventor of SCRUM*. In: Future Of Project Management: people, trends, and ideas on Project Management [online]. [cit. 2016-10-21]. Dostupné z: <https://futureofprojectmanagement.com/2012/01/06/exciting-discussions-with-jeff-sutherland-the-inventor-of-scrum/>.

SUTHERLAND, Jeff. Teams that Finish Early Accelerate Faster. In: *Published Patterns* [online]. [cit. 2016-10-21]. Dostupné z: <https://sites.google.com/a/scrumplop.org/published-patterns/retrospective-pattern-language/teams-that-finish-early-accelerate-faster>.

SUTHERLAND, Jeff. *Yesterday's Weather*. In: Scrumplop [online]. [cit. 2016-10-21]. Dostupné z: https://sites.google.com/a/scrumplop.org/published-patterns/value-stream/estimation-points/yesterday-s-weather

ŠOCHOVÁ, Zuzana, 2014. Backlog Grooming. In: *Zuzis's Blog* [online]. [cit. 2016-10-26]. Dostupné z:<http://soch.cz/blog/management/agile/scrum-management/backlog-grooming/>.

TECHDEV, 2015. Getting Clean, how to apply clean code principles. In: Techdev [online]. [cit. 2016-10-22]. Dostupné z: https://techdev.de/getting-clean-how-to-apply-clean-code-principles/
WATERS, Kelly, 2011. *The Value of Stable Teams* [online]. [cit. 2016-10-22]. Dostupné z: <http://www.allaboutagile.com/the-value-of-stable-teams/>.