Semestrální práce ke kurzu 4IT421 Zlepšování procesů budování IS	
Semestr	ZS 17/18
Autor	Bc. Jan Pippal (xpipj04)
Téma	Případová studie zlepšení SW procesů firmy Beton
Datum odevzdání	7.12.2017

Abstrakt

Tato práce se zabývá návrhy pro zlepšení softwarových procesů konkrétního podniku.

Cílem této práce je poskytnout soubor návrhů na zlepšení softwarových procesů podniku Beton.

Pro přiblížení situace v probíhajících procesech je využito procesní modelování v notaci BPMN a procesy jsou dále slovně popsány.

K identifikaci problémů v procesech je využito zdrojů spojených s agilními metodikami SCRUM, Extrémní programování a dále zdrojů týkajících se jednotlivých projektových disciplín. Zároveň jsou tyto zdroje dále využity pro sestavení návrhů řešení optimalizačních a procesních problémů.

Aplikací uvedených návrhů může dojít v podniku Beton ke zvýšení efektivity procesů a celkovému zlepšení dodávek v jednotlivých iteracích projektu.

Čtenář v této práci může načerpat inspiraci a v případě, že čelí podobným problémům, může uvedených návrhů v míře použít za účelem zlepšení vlastních softwarových procesů.

Klíčová slova: BPMN, Zlepšování procesů, SCRUM, XP, Optimalizace

Obsah

Úvod

Přibližně 1000 let zpět v historii se v Evropě o moc přetahovalo mnoho království. Vizitkou každého panovníka byla především silná rodová konexe, velikost armády, krása a velkolepost královských sídel. Nemít některý z těchto atributů by pro tehdejšího mocnáře znamenalo úpadek, či dokonce zánik. (Nash, nedatováno).

Dnes můžeme v podobné víře hovořit o podnicích různé velikosti a předmětu podnikání. Podobně jako v minulosti jsou podniky reprezentovány sadou atributů, bez kterých je jejich existence na současném trhu nemožná. Vedle hlavní činnosti podniku, jež je prodej služby, popřípadě výrobku, se tak na pozici rozhodujícího atributu dostala integrace podniku s tzv. „Informační" společností. Jedná se o společnost, která je založená na integraci informačních a komunikačních technologií do všech oblastí společenského života v takové míře, že se zásadně mění společenské vztahy a procesy (Jonák, nedatováno). Tato integrace je v podniku reprezentována vývojem a údržbou řešení pro podporu činností podniku a komunikace se zákazníkem.

Vysoce konkurenční prostředí, nákladovost projektů spojených s informačními technologiemi, časová náročnost a splnění očekávání stakeholderů – to jsou hlavní motivy, které ženou exekutivní ředitele a projektové manažery ke zvyšování důrazu na optimalizaci procesů spojených s vývojem a údržbou informačních systémů.

Studie společnosti Computer Economics Inc., která se věnuje zpracovávání rešerše ohledně výdajů v oblasti informačních technologií od roku 1990, uvádí, že průměrně 3 % z celkového obratu sledovaných podniků tvoří výdaje na oblast informačních technologií (Computer Economics, str. 10, 2017).

Bohužel, ne vždy se realizace takového projektu setká s úspěchem. To potvrzuje i Chaos report z roku 2015, kde se u středních až velkých podniků jedná o 24-31 % neúspěšných projektů. Faktor optimalizace zajišťuje přibližně 15 % z případného úspěchu. Proto můžeme konstatovat, že na vině jsou neoptimalizované, zdlouhavé a neefektivní procesy (Standish Group, 2015).

Cíl práce

Cílem této semestrální práce je vytvořit soubor návrhů pro zlepšení procesů konkrétní firmy Beton na základě analýzy jejich současného stavu a představení kritických problémů.

Návrhy na zlepšení budou vycházet z porovnání aktuálních „trendů" v oblasti vývoje software ve formě doporučených postupů a metodik a jejich aplikace na projektu.

V první části dojde k představení firmy a umístění předmětu podnikání do souvislosti s IT.

Ve druhé části je ve stručnosti představena agilní metodika SCRUM, která projekt ovlivňuje a jíž je snaha na projektu plně implementovat.

Ve třetí části je definován soubor důležitých procesů ve firmě formou diagramového zobrazení procesů v notaci BPMN včetně doplnění o slovní popis.

V poslední části je představen soubor návrhů na optimalizaci zmíněných procesů v podniku.

Společnost Beton

Společnost Beton je světovým lídrem v oblasti stavebních materiálů, která poskytuje vysoce kvalitní výrobky a spolehlivé služby. Cílem společnosti je sloužit potřebám svých zákazníků a vytvářet hodnoty tím, že se stává vysoce efektivní a inovativní společností na trhu stavebních materiálů.

Projekt „K zákazníkovi“

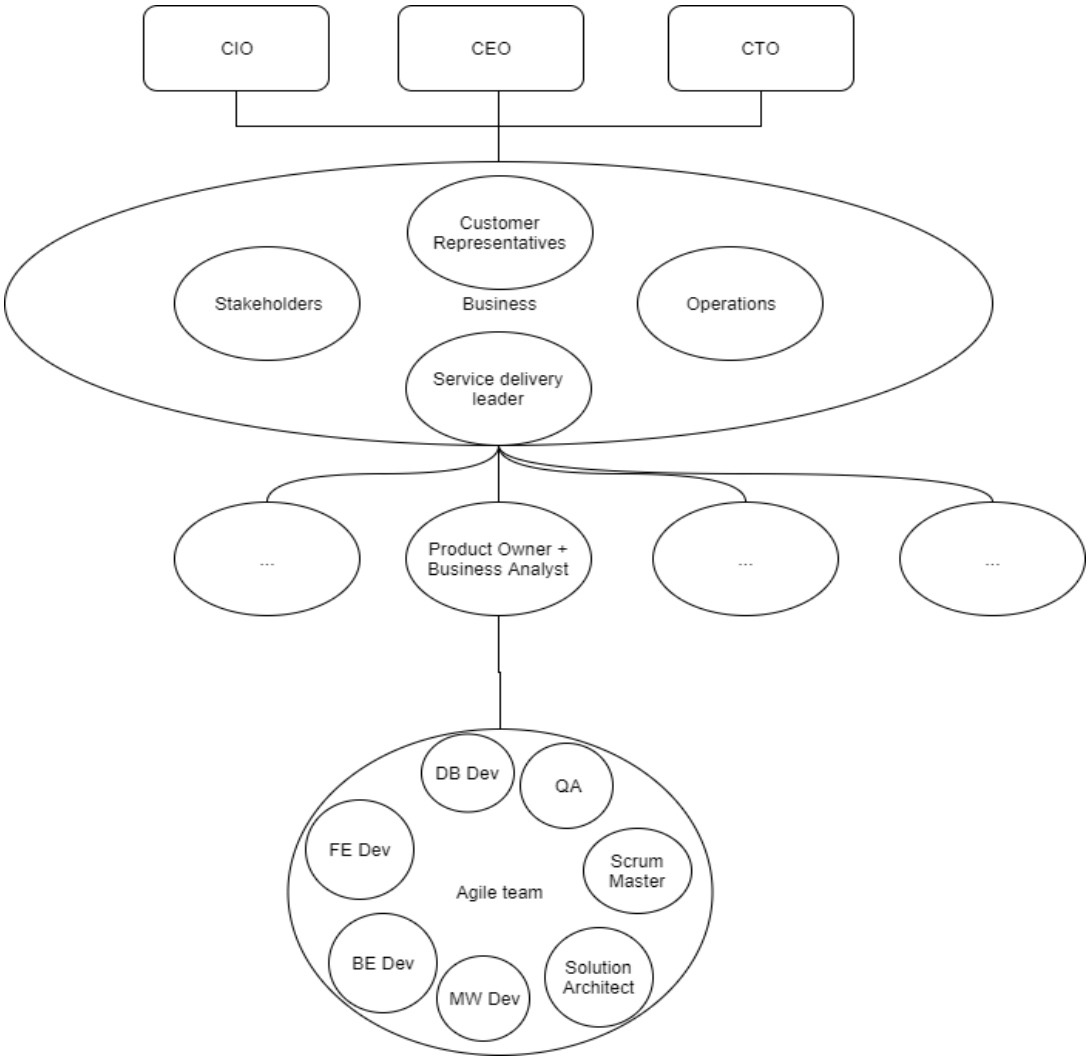
V roce 2015 se společnost Beton rozhodla učinit razantní krok vpřed směrem k zákazníkům. Celý segment podnikání, ve kterém se společnost Beton pohybuje neposkytuje příliš velký prostor pro konkurenční boj. Beton je beton a za posledních 100 let se v jeho přípravě a výrobě moc nezměnilo. Firma Beton se proto pootočila od čistě technologicky-cenového konkurenčního boje ke konkurenci na úrovni zákazníků. Usnadnění komunikace zákazníka s firmou Beton v oblasti zakázek se tedy stává prioritou pro nadcházející 2 roky vývoje softwarového řešení. Tato vize se oznámila veřejně a způsobila povyk mezi konkurenčními subjekty a zároveň vytvořila hromadu očekávání investorů a akcionářů společnosti Beton. Tento tlak na úspěch je důležitým aspektem ovlivňujícím celý projekt.

Finálním produktem tohoto projektu má být kompletní řešení pro propojení stávajících globálních systémů společnosti s nově vytvořeným webovým portálem, který umožní potencionálnímu zákazníkovi obsluhovat veškeré náležitosti procesu realizace objednávek.

K dnešnímu dni je produkt úspěšně vydán, avšak jedná se o produkt, který je oproti plánovanému v mnoha ohledech chudší. Hlavním důvodem pro ochuzení původního plánu jsou právě neoptimalizované procesy, které budou zvýrazněny dále v textu této semestrální práce.

Struktura projektu

Na projektu se podílí kompletní organizační složka IT společnosti Beton čítající přibližně 200 osob. Tato složka se skládá, jak z vlastních zaměstnanců, tak odborných konzultantů z outsourcingu. Z celkového počtu se 120 osob nachází v Praze, 60 v Madridu ve Španělsku a 40 v Novém Dílli v Indii. Celý projekt se snaží implementovat tzv. „Large-Scale Scrum“. Projekt se dále dělí na 8 produktových týmů po 10-15 členech. Každý produktový tým pracuje na části výsledného produktu. Na obrázku 1 je zachycena struktura podniku vzhledem k projektu.



Obrázek 1 Organizační vrstvy společnosti Beton (zdroj: autor)

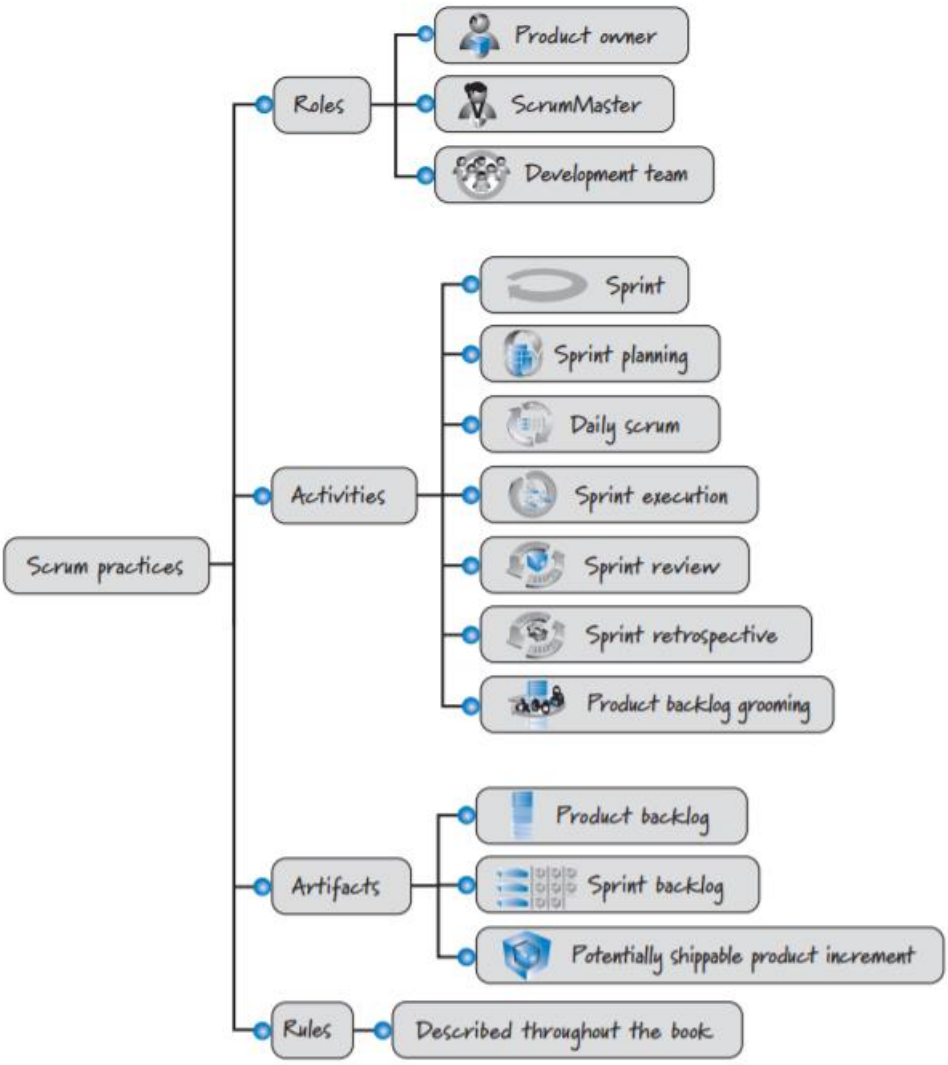
Scrum

Jedná se o agilní metodiku vývoje softwarového produktu. Odlišnost metodik agilních a rigorózních spočívá především v odlišném přisuzování váhy klíčovým aspektům projektu. Zatímco rigorózní metodiky se snaží postavit fungující projekt na přesně definovaných procesech a jejich detailním provádění, u agilních metodik se více cení schopnost reagovat na změnu a frekvenci dodávky fungující části (inkrementu). Nicméně přesně to nám osvětluje i Agilní manifest z roku 2002, podle kterého dáváme přednost (Beck a spol., 2001):

- individualitám a komunikaci před procesy a nástroji,
- provozuschopnému software před obsažnou dokumentací,
- spolupráci se zákazníkem před sjednáváním kontraktu,
- reakci na změnu před plněním plánu.

K těmto zásadám je přidáno 12 principů, jež jsou dostupné na stránkách agilního manifestu.

Organizaci celého Scrumu nejlépe vystihuje obrázek 2.



Obrázek 2 Scrumové praktiky (Rubin, 2012)

Ve stručnosti jsou popsány vybrané části z metodiky pro účely pochopení dění na projektu „K zákazníkovi“. Vysvětlení je seřazeno odlišně od obrázku pro lepší pochopení souvislostí.

Development team – tým vývojářů zahrnující architekty, programátory, testery, databázové specialisty, designéry a další, kdož jsou odpovědní za vypracování požadovaného produktu. Velikostně by se takový tým měl pohybovat mezi 5-9 členy.

Scrum master (SM)– pomáhá každému v týmu k dosažení maximálního výkonu prostřednictvím procesního vedení a poskytování rad v otázkách organizace scrumu. Dále napomáhá k rychlejšímu řešení případných problémů a usnadňuje komunikaci.

Product owner (PO) – je autorita odpovědná za rozhodování, která funkcionalita, změna a v jakém pořadí budou dodávána. Zajišťuje a komunikuje jasnou vizi celého scrum týmu. Aktivně spolupracuje se SM a vývojářským týmem a odpovídá na případné implementační otázky.

Product backlog – PO spolupracuje s interními a externími stakeholdery, aby sesbíral a vydefinoval položky do produktového backlogu. Ten tedy chápejme jako prioritně organizovaný list požadavků (práce) vyžadovaných k uskutečnění po scrum týmu.

Product backlog grooming – aktivita, která se věnuje představení produktového backlogu scrum týmu. Probíhá případné rozpadnutí příliš složitých úkolů, nebo diskuze nad nemožností realizace, některých položek. Výsledkem této aktivity by měl být produktový backlog doplněný o odhady pracnosti jednotlivých položek ze strany scrum týmu.

Sprint – iterace – opakovaný časový úsek až měsíc dlouhý. Na konci každého sprintu by měl být scrum tým schopen dodat něco přínosného pro uživatele, nebo zákazníka.

Sprint planning – aktivita, na které PO, SM a vývojářský tým plánují **sprint backlog** – zprioritizovaný produktový backlog vztažený na časový úsek právě jednoho sprintu.
V povědomí během této aktivity je potřeba brát jak časovou náročnost jednotlivých položek, tak jejich komplexitu.

Daily scrum – aktivita, která se provádí každý den. Před začátkem pracovního úseku na přibližně 15 minut se schází členové scrum týmu. Každý člen odpovídá na tři otázky:

1. Jak jsem pokročil od předchozího daily scrumu?
2. Co plánuju dělat nyní?
3. Existuje něco, co mě blokuje v pokračování?

Potentially shippable product inkrement – výsledek sprintu. Na konci sprintu by měl být k dispozici inkrement produktu, který přináší uživateli, či zákazníkovi hodnotu. V případě ukončení projektu na konci daného sprintu je výsledným produktem právě poslední dostupný finální produkt s inkrementem (Rubin, str. 14-28, 2012).

Story

Nebo také uživatelská story je požadavek na softwarový systém vyjádřen několika větami, ideálně za použití netechnického (uživatelského jazyka). Příkladem takové story, kde je požadavkem implementace košíku pro objednávání betonu online může být:

„Jako uživatel webové aplikace chci po dokončení objednávky betonu získat přehled všech mnou vybraných položek v přehledném nákupním košíku.“

Dále můžou následovat ve story informace, které nám přesněji vymezí omezení ať už byznysová, nebo technická. Např.

„

- *Nelze objednat na jedno nákladní auto více jak 20 tun betonu.*
- *Na obrazovce se musí zobrazit celý název produktu a jeho cena*
- *Uživatel může zvětšit obrázek produktu přejetím myši*

“

Přiloženy pak v mohou být prototypy designu, různé emailové konverzace osvětlující problémy, kladené otázky, závislost story na jiné (např. nejdříve je potřeba dokončit Vložení do košíku).

Procesy

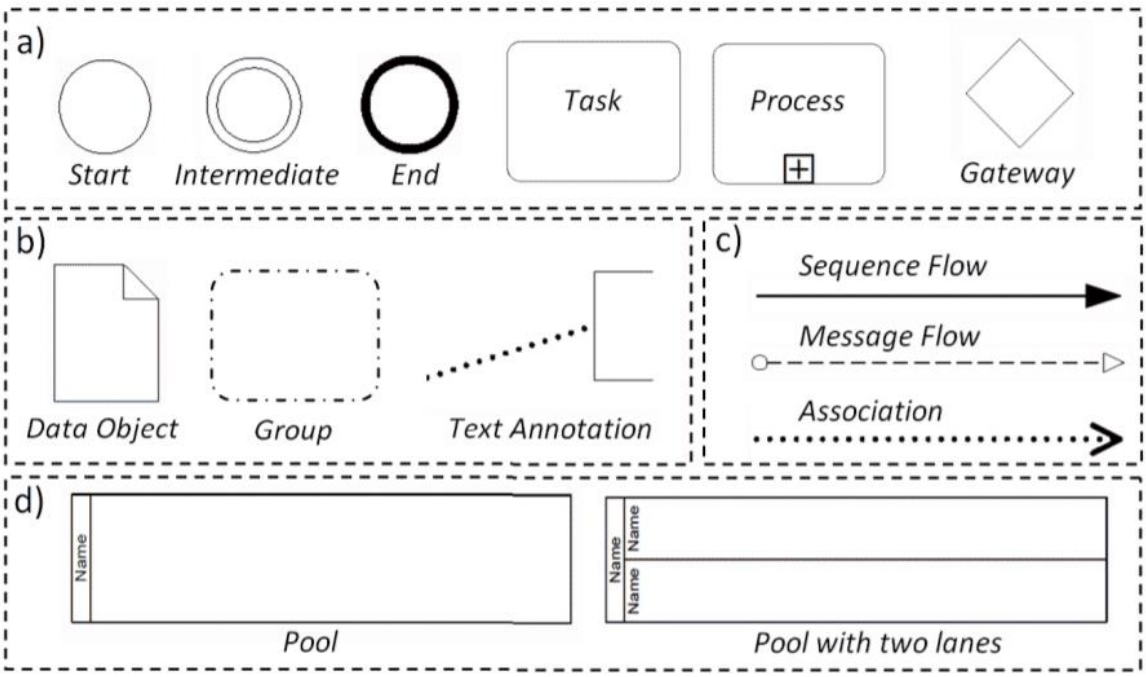
Procesem se rozumí:

„Softwarový proces je definován jako sada činností, metod, praktik a transformací, které lidé používají pro vývoj a údržbu software a dalších produktů (projektových plánů, návrhů, testovacích případů apod.)“ (Buchalcevová, str. 23, 2005).

V této kapitole jsou vytypovány procesy, které probíhají v IT oddělení společnosti Beton na projektu „K zákazníkovi“. Procesy jsou zobrazeny na BPMN diagramu, dále doprovázeny slovním popisem a identifikovány nejvýraznější problémy jejich současné aplikace. Tyto procesy jsou vybrány z celého projektu, jelikož jsou pro dosažení úspěchu projektu klíčové a jedná se o nejlepší kandidáty na optimalizaci. V kapitole Návrhy zlepšení budou rozepsány hlavní návrhy pro zlepšení těchto procesů.

BPMN 2.0

Business Process Modeling Notation (BPMN) je metoda vývojového diagramu, která od začátku do konce modeluje kroky plánovaného podnikového procesu. Hlavním klíčem k řízení podnikových procesů je vizuální popis detailního sledu podnikových činností a informačních toků potřebných k dokončení procesu.



Obrázek 3 Elementy BPMN (zdroj: https://www.lucidchart.com/pages/bpmn)

Základními elementy na obrázku 3 jsou v BPMN 2.0 jsou objekty kategoricky rozdělené na

Tokové objekty – události, činnosti, brány

Spojovací objekty – sekvenční tok, asociace, tok informací

Plavecké dráhy – bazény, dráhy

Artefakty – datové objekty

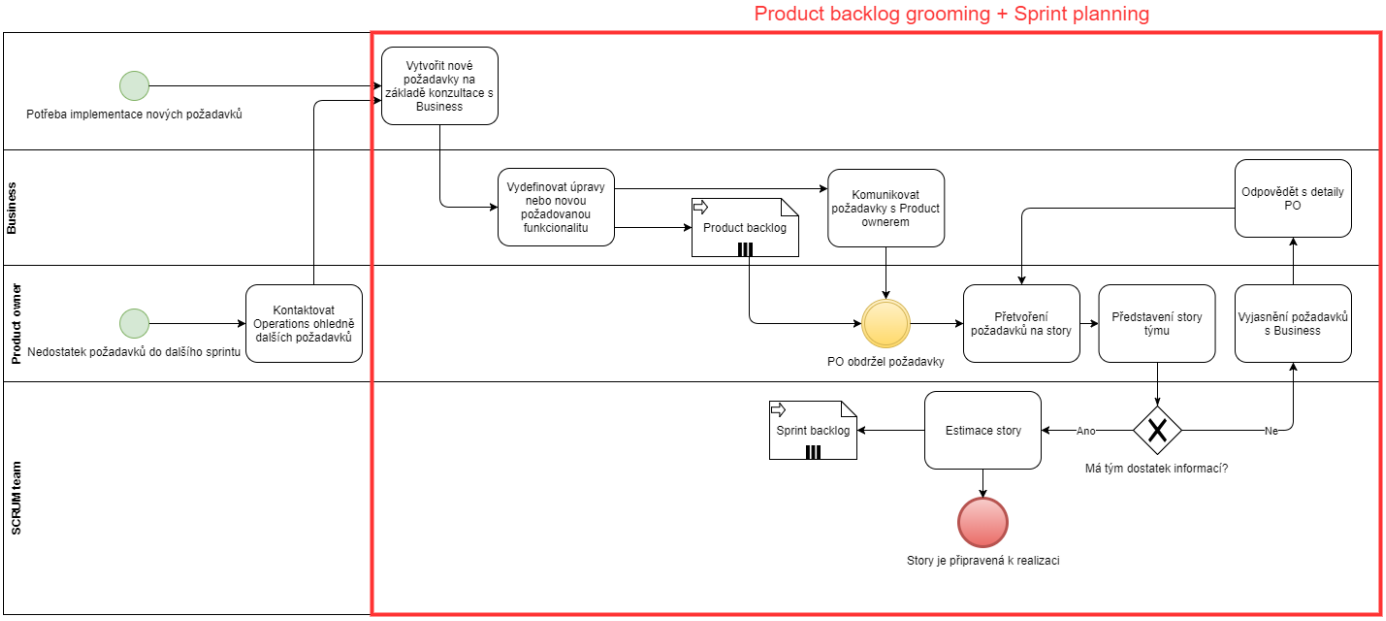
(Ciaramella a spol., 2014)

Proces : Vytvoření požadavků

Proces Vytváření požadavků je alfa a omega celého dění na projektu. Právě v tomto procesu se určuje rozsah funkčních i nefunkčních požadavků celé části produktu. Bez tohoto procesu by žádný z týmů nevěděl, co se žádá, nebo která z již implementovaných funkcí je nevyhovující a je jí potřeba změnit, či odstranit. Abstraktní vrstva zvaná Operations a Business se zde stará o vypracování funkčních požadavků na výsledný produkt. Mezi Operations chápejme pracovníky delivery managementu, který se stará o zajišťování zakázek z pohledu přímých dodávek betonu, či stavebních materiálů. Tito lidé tedy mají přímý kontakt s potřebami plynoucími z pozice logistiky, výroby, plánování. Druhá vrstva Business je složena z části podniku, která se stará o vyřizování faktur, objednávek, plateb, produktových katalogů. Pokud je sprint u konce a čeká se na další požadavky na aplikaci, je to právě kombinace těchto dvou vrstev, kdo zhodnotí do jaké míry je implementované řešení v souladu s požadavky. Práce Product ownera (PO), zde zahrnuje převedení těchto hrubých požadavků do podoby stories, jak bylo definováno v kapitole Scrum. Tyto stories následně PO prezentuje týmu, který na základě své expertízy spolu s business analytikem stanoví, zda obsahují dostatek informací potřebných k implementaci požadavku. Pokud je třeba další ujasnění, putuje story zpět k Business, kde potřebné informace doplní. Proces graficky znázorněn na obrázku 4.

Problémy

- Na projektu neexistuje dlouhodobější plán – vize – vše se děje na úrovni sprintu
- Požadavky Operations a Business nejsou nijak redukovány, analyzovány
- Sprinty mají nerovnoměrně rozložený rozsah – někdy moc málo, někdy příliš
- Estimace komplexnosti a časové náročnosti nemají žádný vliv na plánování
- PO není součástí procesu stanovení požadavků – je pouze jejich poslem
- Většina stories neobsahuje kvalitní Akceptační kritéria a design



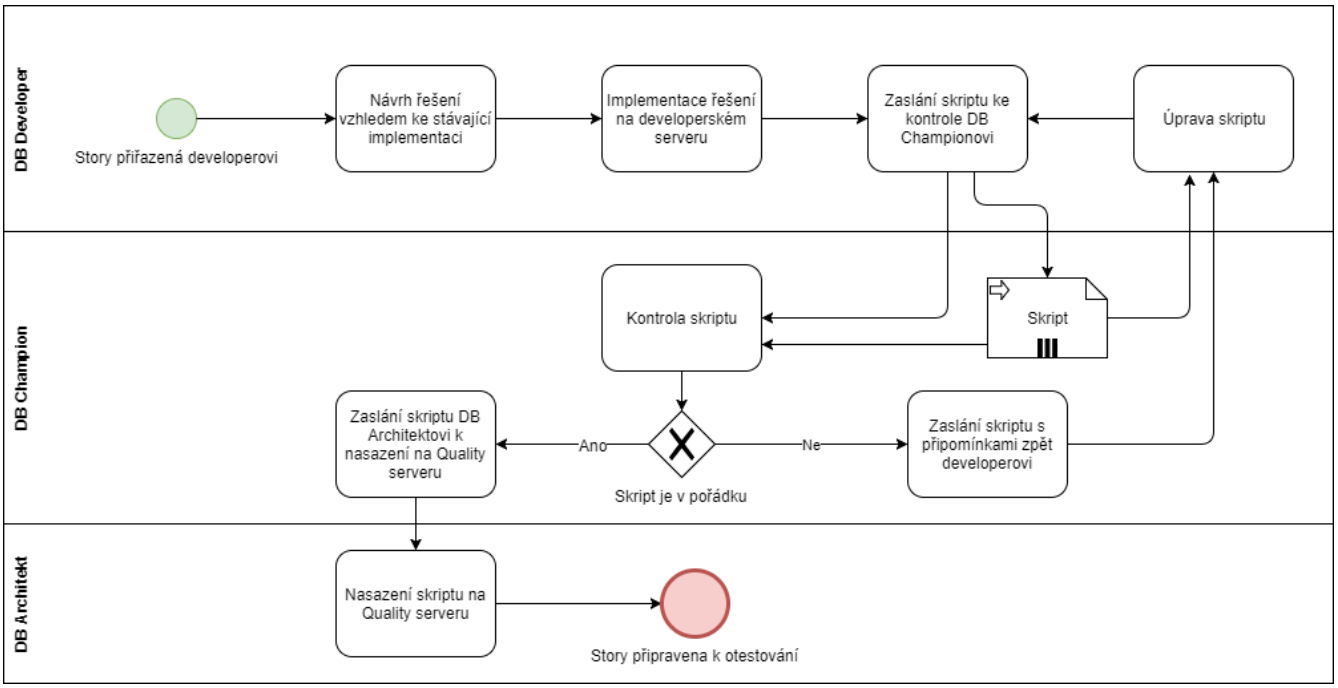
Obrázek 4 Proces vytváření požadavků (zdroj: autor)

Proces: Nasazování databázové změny

Tento proces patří do rodiny procesů vztahujících se k nasazování aplikace na testovací prostředí (Quality) a je v mírné alternaci přizpůsobitelný všem kategoriím vývoje. Po sprint planningu, kdy dojde k přidělení odpovědných developerů k jednotlivým stories, si je každý developer vědom svých přidělených úkolů, které je třeba v rámci sprintu dokončit. Každý developer pečlivě rozmyslí všechny aspekty nutné k implementaci řešení. Následnou implementaci (v tomto případě nejčastěji T-SQL skripty) nasadí na vývojářské prostředí databáze. Poté aplikovaný skript zašle DB Šampionovi ke kontrole. Pokud nejsou v skriptu žádné problémy, je tento skript zaslán DB Architektovi k nasazení na testovací prostředí, kde dochází k testování členy QA týmu. Proces graficky znázorněn na obrázku 5.

Problémy

- Vývojáři netestují svůj kód – nevytvářejí jednotkové a integrační testy
- Veškeré skripty procházejí kontrolou kódu pouze od Šampiona – ten je těmito kontrolami přetížen
- V DB vrstvě neexistuje verzovací systém – práce jednoho vývojáře může přímo ovlivnit práci druhého
- DB Architekt neplní svojí roli, místo toho se věnuje pouze nasazování skriptů
- Neprobíhá žádná komunikace mezi vývojáři různých vrstev – to vede k integračním chybám a blokaci



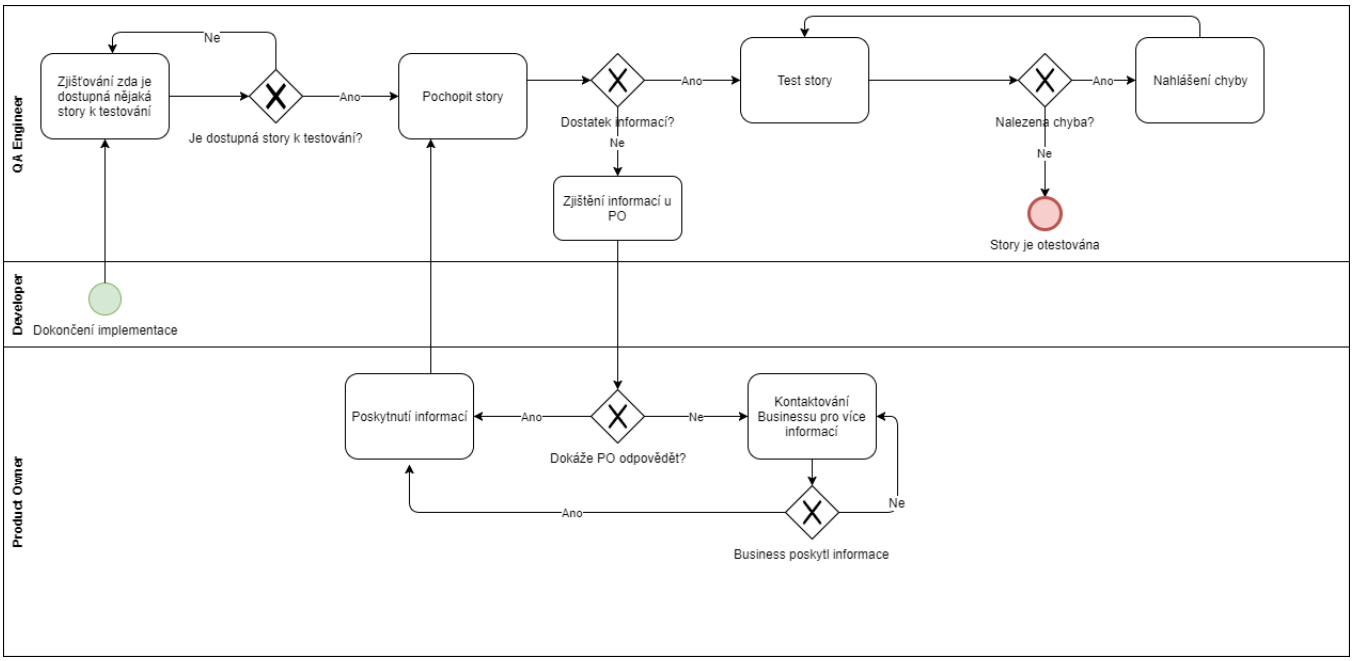
Obrázek 5 Proces nasazování databázové změny (zdroj: autor)

Proces: Testování

Proces testování následuje po implementaci jednotlivých částí story a jejich nasazení na testovací server. Prací člena QA týmu je zjišťovat, zda není nějaká ze stories připravena k testování. Pokud ano, stráví QA čas ve snaze pochytit veškeré aspekty story k testování. Nedostatek informací případně komunikuje s PO, který zajistí odpovědi na otázky.
S dostatkem informací může QA pokročit k samotnému testování. Nalezenou chybu při testování případně QA nahlásí do systému. Pokud v implementaci story nenalezne chybu, je story otestována a zavřena. Proces graficky znázorněn na obrázku 6.

Problémy

- Testování neprobíhá agilně – začíná až v momentě, kdy je story implementována
- Neexistuje přímá komunikace mezi QA a vývojářem – dokončení implementace, status chyby
- Neexistuje test case management
- QA nemá ve většině případech kvalitní podklady a dostatek informací pro testování
- Nulový reporting činnosti QA



Obrázek 6 Proces testování (zdroj: autor)

Návrhy zlepšení procesů

Následování best practices v metodice SCRUM

Jedna z nejčastějších chyb při aplikaci agilních metodik je snaha užívat si, co nejvíce volnosti, bez svazujících procesů. To vede často k opomíjení tzv. best practices scrumu, kterým se věnuje mnoho publikací, blogů, fór. Na projektu společnosti Beton taková ignorace vyústila ve kombinaci všech aktivit do jedné velké neorganizované sešlosti celého scrum týmu.
V jednom neefektivně dlouhém meetingu se snaží PO s týmem jednak vytvořit popis stories, zároveň postihnout potřebné detaily a vytvořit odhady a naplánovat sprint. Už jenom tímto sáhodlouhým výčtem je jasné, že se na meetingu nevytvoří takřka žádný hodnotný výstup. Výstupem z tohoto meetingu je mnoho otázek, nesmyslných číselných odhadů a nejčastěji závazek dokončit všechny požadavky v rámci následujícího sprintu. A to je chyba.

Problémy stávajícího procesu a jak je řešit:

„Na projektu neexistuje dlouhodobější plán – vize – vše se děje na úrovni sprintu“

Scrum nabízí takzvané Epiky (Epics) – jejich cílem je vytvořit větší celky, které můžou být rozpadnuty na více menších částí (Story) např. ve firmě Beton se může jednat o epics – Online objednávka, Stahování faktur, Mobilní aplikace, Redesign. Na základě epiků je pak možné ve scrumu dělat i dlouhodobější plánování např. na 2-4 sprinty dopředu a dle toho alokovat potřebné zdroje.

„Požadavky Operations a Business nejsou nijak redukovány, analyzovány“

Zde je hlavním problémem oddělení PO a Businessu, kteří pouze komunikují o výsledku. PO by měl být zapojen spolu s rolí byznys analytika (a případně i designéra) v procesu

tvorby požadavků. Může tak už v samotném počátku vyjasnit některé informace vztahující se k implementaci, kvalitněji vystihnout požadavek v akceptačních kritériích a zároveň zastavit nerealizovatelné požadavky a místo nich dodat týmu požadavky akceptovatelné.

„Sprinty mají nerovnoměrně rozložený rozsah – někdy moc málo, někdy příliš“

To je výsledkem neefektivního odhadování komplexnosti stories, nebo neschopností pracovat s těmito odhady. Jednak je třeba zjistit, zda-li převážně PO a scrum master využívají zažité postupy pro plánování sprintů. Častou chybou, zde je odhadování příliš velkých požadavků, které je doporučováno nejdříve rozpadnout na menší lépe odhadovatelné části. Problém ale může pramenit přímo z odhadů scrum týmu. Zcela jednoduše se zde projeví nekoncentrovatelnost na obsah story popřípadě nedostatek zkušeností s danou implementací. Tyto oblasti by měl pokrývat scrum master.

„Estimace komplexnosti a časové náročnosti nemají žádný vliv na plánování“

Jak již bylo zmíněno výše. Získat odhady od týmu nebývá těžké. Těžká je správná volba, jak s těmito odhady naložit. Těžko dojde k plánování, když se s čísly neprovede žádná matematika. Zdrojů jak postupovat je dostupných hodně a občas je třeba zkusit experimentovat.

„PO není součástí procesu stanovení požadavků – je pouze jejich poslem“

PO by měl být součástí procesu stanovení požadavků. Zákazník totiž může chtít v počátku raketoplán, ale spokojí se i s letenkou na Havaj.

„Většina stories neobsahuje kvalitní Akceptační kritéria a design“

Zde se projevu především píle a důslednost PO. Je pouze v jeho (případně byznys analytika) schopnostech (především komunikačních) převést požadavky do stories. Pro sestavu potřebných informací se doporučuje sestavit předlohu (template), jak má story vypadat. Předlohypomáhají uchovat konzistenci důležitých informací.

Vývojářské principy plynoucí z Extrémního programování

Extrémní programování

Extrémní programování je jedna z agilních metodik, která si získala největší oblibu během roku 2000. Jedná se o soubor praktik a nápadů, kde do centra pozornosti a odpovědnost je stavěn vývojář.

Základní aktivity a programovací principy

Mezi čtyři základní aktivity Extrémního programování patří

- **Kódování**
- **Testování**
- **Naslouchání**
- **Návrh**

Definované principy

- Plánovací hra
- Krátké releasy
- Metafora
- Jednoduchý design
- Testování
- Refaktorizace
- Párové programování
- Kontinuální integrace
- Kolektivní odpovědnost
- 40-ti hodinový pracovní týden
- Zákazník na místě
- Standardy kódování

Není účelem této práce definovat všechny tyto principy. Proto vyberu pouze ty, které jsou aplikovatelné na problémy plynoucí z procesu.

Problémy stávajícího procesu a jak je řešit:

„Vývojáři netestují svůj kód – nevytvářejí jednotkové a integrační testy“

Jednotkové **testy** nejvíce ovlivňují kvalitu produktu. Vzhledem k faktu, že jednotkové testy jsou „nejlevnější“ na vytvoření a dokážou pokrýt většinu implementace, jejich nevytváření je holé podkopávání kvality produktu. Většina chyb na úrovni kódu může v konečném důsledku způsobit velké a často i finanční důsledky, pokud neodhalena docílí produkčního prostředí. Vytváření jednotkových testů dodává jistotu do vývojářovi práce.

„Veškeré skripty procházejí kontrolou kódu pouze od Šampiona – ten je těmito kontrolami přetížen“

Kolektivní odpovědnost – v týmu by teoreticky neměl existovat člen, jehož absence by způsobila neschopnost týmu doručit produkt. Proto je potřeba sdílet informace a schopnosti v týmu. Role DB Šampiona by tak měla být rozpuštěna a pozice sdílena s ostatními databázovými vývojáři.

„V DB vrstvě neexistuje verzovací systém – práce jednoho vývojáře může přímo ovlivnit práci druhého“

Neexistence verzovacího systému vyžaduje abnormálně vysokou koordinovanost vývojářů. Ta je většinou ignorována a proto, se setkáváme v projektu bez verzovacího systému s častými konflikty na úrovni vývojářského kódu. Změna jednoho, může zničit práci druhého. Zároveň takový systém neumožňuje vrátit se zpět při nalezení kritického problému. Pro zachování principu **kontinuální integrace** by mělo být dostupné úložiště s možností kooperace více vývojářů na jednom projektu.

„DB Architekt neplní svojí roli, místo toho se věnuje pouze nasazování skriptů“

Jednoduchý návrh - hlavní úlohou Architekta je tvořit a udržovat architekturu. Schvalovat změny, diskutovat optimalizaci, navrhovat řešení a integraci, dodávat diagramy navrhovaného systému.

„Neprobíhá žádná komunikace mezi vývojáři různých vrstev – to vede k integračním chybám a blokaci“

Komunikace je klíčovým aspektem na jakémkoliv projektu. Obzvláště v případech, kdy na sobě jednotlivé implementační vrstvy přímo závisí (DB-MW-FE) je důležité udržovat komunikaci o prováděných změnách. Jistou možností této závislostní komunikace je kvalitní daily scrum a koncentrovanost během product backlog groomingu a sprint planningu. (Tutorials Point, 2016)

Testování ve SCRUMu (agile)

Zhruba před deseti roky se většina softwarových projektů držela po sobě jdoucích fází. Nejdříve bylo třeba analyzovat, poté navrhnout, implementovat a nakonec otestovat. Integrace a testování těchto projektů probíhaly na konci celého vývojářského cyklu, který mohl trvat měsíce i roky. [testing agile]

V agilním týmu je důraz na dodání vysoce kvalitního produktu s přidanou hodnotou a každý člen týmu vyvíjí maximální úsilí. Programátoři vytváří jednotkové a integrační testy svého kódu. Agilní tester se soustředí, aby scrum tým dodával zákazníkovi potřebnou kvalitu produktu. Funguje tedy jako most mezi kvalitou produktu a zákazníkem a jeho úkolem je eliminace případných rizik spojených s rozdíly mezi požadavky a implementací. Proto se nejčastěji agilní tester soustředí na takzvané byznysové testy (popř. akceptační). Zajišťuje, že požadovaná funkcionalita podléhá testování, že aplikace dokáže vydržet jistý nápor uživatelů, že zachovává stabilitu při výpadku energie, že je zabezpečená proti úniku citlivých údajů, že je uživatelsky přívětivá a intuitivní.

Hlavním rozdílem oproti testování v agilním a tradičním projektu je především v často se opakovaných iteracích. Vzhledem k tomu, že jeden sprint může být velmi krátký – (14 dní), se klade v agilních projektech u testera důraz především na produktivitu a efektivnost testování. Často se proto opomíjí veškeré důležité části procesu testování, jako je test analýza, příprava test strategie, design testovacích scénářů a reporting. Platí, že čím dříve se s testováním začne, tím lépe. Proto by se agilní tester měl aktivně zapojit už ve fázi product backlog groomingu a případné nejasnosti a připomínky k jednotlivým položkám backlogu ihned komunikovat. Ve chvíli, kdy je dostupná definice a dostatek informací ke každé story, je potřeba připravit testovací scénáře a strategii testování. Jakmile je pak story implementována a připravená k testování, zbývá testerovi pouze zaznamenat výsledky testů, případně otestovat znovu již opravené nahlášené chyby.

Problémy stávajícího procesu a jak je řešit:

„Testování neprobíhá agilně – začíná až v momentě, kdy je story implementována“

Zde by měla přijít především snaha ze strany QA. Na projektech se povětšinou nenajde nikdo, kdo by tlačil agilní testování, pokud to samo QA nedělá. Ve výsledku se sice

implementace story dokončí v rámci sprintu, nicméně jejich otestování a tím pádem nasazení na produkční prostředí dojde nejčastěji až ve sprintu následujícím. Správně by se tedy QA měl zapojit už ve fázi product backlog groomingu. Vyjasnit si otázky (pokud jsou) a přispět svým odhadem ohledně délky testovací fáze. S dostatkem informací může QA připravit veškeré testovací scénáře ještě před implementací. To pak redukuje čas potřebný k finálnímu otestování story pouze na čas potřebný ke spuštění testů a nahlášení případných chyb.

„Neexistuje přímá komunikace mezi QA a vývojářem – dokončení implementace, status chyby“

Jedním z hlavních požadavků na jakoukoliv pozici do agilního týmu je především schopnost spolupráce a komunikace. Je velice důležité nastavit správné komunikační kanály mezi vývojáři a testery. Ve chvíli, kdy je story připravená k otestování, měl by vývojář upozornit testera. Ve chvíli, kdy tester najde chybu, měl by jí nahlásit a zároveň upozornit vývojáře. Tento styl komunikace šetří čas a zvyšuje efektivitu týmu. Dobrou praktikou je také využít daily scrum krátké setkání k zdůraznění blokací, či kritických chyb.

„Neexistuje test case management“

Ačkoliv jeden z principů agilového vývoje je přednost provozuschopnému softwaru před obsáhlou dokumentací, je to často chápáno jako pobídka k zavrhnutí celého systému testovacích scénářů. To je ovšem velká chyba. V kompetenci QA týmu je definice granularity testovacích scénářů. Minimálně by se však měla granularita pohybovat na úrovni akceptačních kritérií. Minimálně je nutné zachovávat sadu testovacích scénářů pro účely regresního testování.

„QA nemá ve většině případech kvalitní podklady a dostatek informací pro testování“

Nedostatek podkladů je odpovědností PO. QA může pouze přispět svými připomínkami v product backlog groomingu.

„Nulový reporting činnosti QA“

Obecně by měl probíhat reporting QA na denní bázi a to formou daily scrumu. Dále by měl být otevřený kanál pro komunikaci závažných chyb a jejich sledování. (Crispin a Gregory, 2009).

Atlassian software naplno

Atlassian Corporation je softwarová společnost vyvíjející produkty pro vývojáře, projektový management a management obsahu. Nejznámějším produktem je aplikace pro sledování životního cyklu požadavku. Mezi nástroje z rodiny Atlassian využívané na projektu firmou Beton patří:

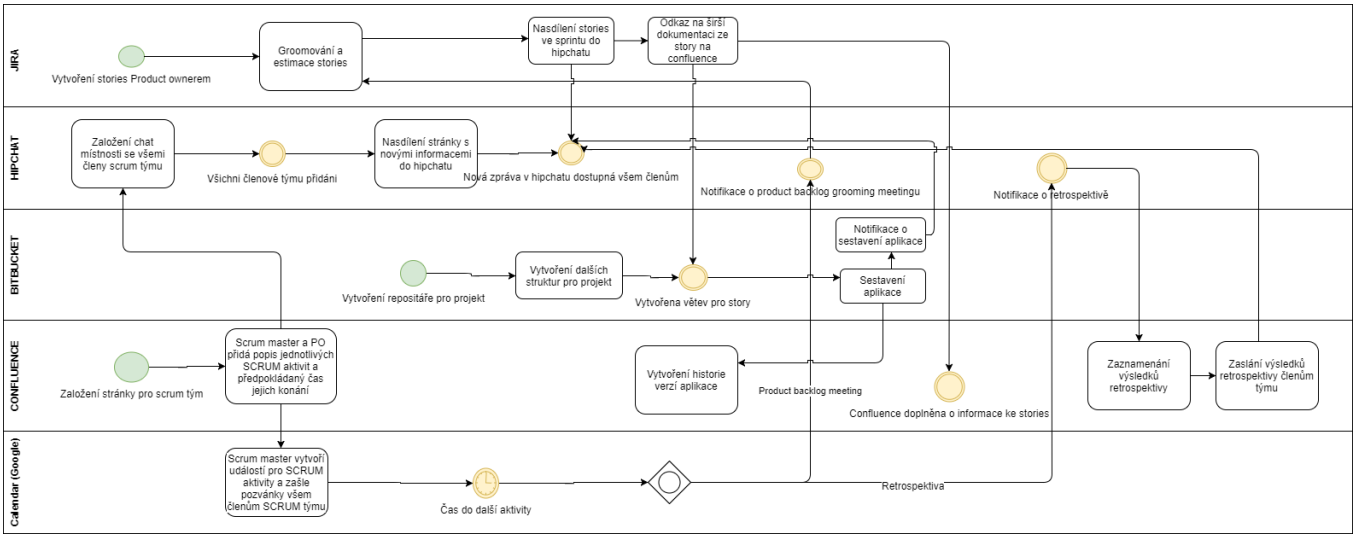
Jira Software – nástroj pro plánování, sledování a release ticketů (požadavků)

Confluence – nástroj pro management znalostí a informací v podniku

Bitbucket – nástroj pro správu kódu, verzování, kolaboraci, nasazování

Hipchat – nástroj pro týmovou komunikaci – umožňuje chat, videokomunikaci, screensharing

Ačkoliv na projektu dochází k využívání jednotlivých nástrojů, je kompletně opomíjena možnost integrace jednoho nástroje do druhého. A právě v tomto směru je poskytnut další návrh na zlepšení procesu, a to integrace těchto nástrojů. Na následujícím diagramu (obrázek 7) je zobrazeno jedno z možných řešení.



Obrázek 7 Atlassian integrace proces (zdroj: autor)

Závěr

Je jisté, že v této semestrální práci není možné pokrýt veškerou problematiku uvedených procesů. Zároveň si je autor vědom, že uvedené rady mohou být pro čtenáře až příliš abstraktní a v reálné aplikaci vyvstávají další otázky. Cílem této práce bylo sestavit sadu návrhů pro optimalizaci, či zlepšení procesů daného podniku Beton. V rámci procesního modelování byly představeny hlavní problémy, které daným opakovaným procesům brání v efektivním plnění. V poslední kapitole došlo k vytvoření mnoha návrhů, jak problémy řešit na základě volně dostupných zdrojů a dosáhnout tak optimalizace procesů v prostředí agilové metodiky SCRUM. Tato práce může sloužit ostatním v případě, že hledají odpovědi na otázky, nebo se nacházejí v podobné situaci jako společnost Beton. Koncept této práce byl zároveň představen projektovému týmu společnosti Beton, která se rozhodla část navrhovaných řešení implementovat.

Zdroje

BECK, K., BEEDLE, M., BENNEKUM, A., COCKBURN, A., CUNNINGHAM, W., FOWLER, M., GRENNING, J., HIGHSMITH, J., HUNT, A., JEFFRIES, R., KERN, J., MARICK, B., MARTIN, R., MELLOR, S., SCHWABER, K., SUTHERLAND, J. and THOMAS, D. (2001). Manifesto for Agile Software Development. [online] Agilemanifesto.org [cit. 2017-12-04]. Dostupné z: http://www.agilemanifesto.org/

BUCHALCEVOVÁ, Alena. Metodiky vývoje a údržby informačních systémů. 1. vyd. Praha : Grada, 2005. 163 s. Management v informační společnosti. ISBN 80-247-1075-7

CIARAMELLA, A., CIMINO, M., LAZZERINI, B., MARCELLONI, F.. (2009). Using BPMN and Tracing for Rapid Business Process Prototyping Environments.. ICEIS 2009 - 11th International Conference on Enterprise Information Systems, Proceedings. 206-212. [cit. 2017-12-04]. Dostupné z : https://www.researchgate.net/publication/220711864_Using_BPMN_and_Tracing_for_Rapid_Business_Process_Prototyping_Environments

COMPUTER ECONOMICS, 2017. IT spending and staffing benchmarks 2017/2018 [online]. Computer Economics Inc. [cit 2017-12-04] Dostupné z: https://www.computereconomics.com/temp/ISS2017Ch1ExecSum99112.pdf

CRISPIN, L., GREGORY, J., 2009. Agile Testing: A practical guide for testers and agile teams. Addison-Wesley. Dostupné z: http://aksitha.com/Software%20Testing/AGILE_TESTING_-_A_PRACTICAL_GUIDE_FOR_TESTERS_AND_AGILE_TEAMS.pdf ISBN-13: 978-0321534460

JONÁK, Zdeněk. Informační společnost. KTD: Česká terminologická databáze knihovnictví a informační vědy (TDKIV) [online]. Praha: Národní knihovna ČR, 2003-[cit. 2013-05-17]. Dostupné z: http://aleph.nkp.cz/F/?func=direct&doc_number=000000468&local_base=KTD.]

NASH, Tim, nedatováno. Medieval Kings – It was good to be king in medieval times. [online]. The Finer Times. [cit. 2017-12-04] Dostupné z: http://www.thefinertimes.com/Ancient-History/it-was-good-to-be-king-in-medieval-times.html

RUBIN, K. 2012. Essential Scrum: A practical guide to the most popular agile process. Upper Saddle River, NJ: Addison-Wesley, pp.4 5. ISBN 0-13-704329-5

STANDISH GROUP, 2015. Chaos report 2015 [online]. Standish group [cit 2017-12-04]. Úryvek dostupný z : https://www.computereconomics.com/temp/ISS2017Ch1ExecSum99112.pdf

TUTORIALS POINT, 2016. Extreme programming tutorial [online]. Tutorials Point (I) Pvt. Ltd. [cit. 2017-12-04]. Dostupné z : https://www.tutorialspoint.com/extreme_programming/extreme_programming_tutorial.pdf