Univerzita Hradec Králové
Fakulta informatiky a managementu
Katedra informačních technologií

Metody řízení IT projektu

Diplomová práce

Autor: Bc. Mariya Nagornyak
Studijní obor: Informační management

Vedoucí práce: Ing. Tereza Otčenášková, BA, Ph.D.

Hradec Králové Květen 2023

Prohlášení:

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

vlastnoruční podpis

V Hradci Králové dne 4.5.2023 Bc. Mariya Nagornyak

Poděkování:

Děkuji vedoucí diplomové práce Ing. Tereza Otčenášková, BA, Ph.D. za metodické vedení práce a rady při zpracování této práce.

Anotace

Řízení IT projektů je náročnou disciplínou vyžadující efektivní plánování, organizaci, koordinaci činností a průběžné vyhodnocování výsledků. Výběr správné metody pro řízení projektu je pak klíčovým rozhodnutím majícím zásadní vliv na výsledky projektu. Dostupná literatura na toto téma je však často teoretická a nezohledňuje specifika konkrétních projektů. Výběr vhodné metody tak může být pro projektové manažery obtížný. Cílem této diplomové práce je provést srovnání vybraných metod pro řízení IT projektů a demonstrovat jejich praktické využití na základě případové studie. Výstupem je praktický průvodce, který čtenářům usnadní porozumění hlavním rozdílům mezi vybranými metodami a přístupy, a tím jim umožní efektivně zvolit optimální metodu pro jejich konkrétní IT projekt.

Klíčová slova

Agilní metody, Fáze projektu, IT projekt, Metody řízení, porovnání metod, Projektový management, Scrum, Kanban, Waterfall metody

Annotation

Title: Methods of IT Project Management

IT project management is a complex discipline that requires effective planning, organization, coordination of activities, and continuous evaluation of results. The selection of the right method for project management is a crucial decision that can have a significant impact on project outcomes. However, the available literature on this topic is often theoretical and does not consider the specifics of individual projects, making it difficult for project managers to choose the appropriate method. The aim of this master's thesis is to compare selected methods for managing IT projects and demonstrate their practical application based on a case study. The outcome is a practical guide that seek to facilitate readers´ understanding of the main differences between the selected methods and approaches, enabling them to effectively choose the optimal method for their specific IT project.

Keywords

Agile methods, comparison of methodologies, IT project, Kanban, Management methods, Project management, Project phases, Scrum, Waterfall methods.

Obsah

Seznam obrázků

Seznam tabulek

1 Úvod

Řízení projektů v oblasti informačních technologií (IT) je náročnou disciplínou, která vyžaduje pečlivé a efektivní plánování, organizaci, koordinaci činností, a průběžné vyhodnocování výsledků, aby bylo dosaženo úspěšného dokončení projektu. Projekty v oblasti IT mohou být velmi různorodé, dynamické a často také komplexní, jelikož mohou zahrnovat širokou škálu technických systémů, aplikací, infrastruktury a integrací mezi nimi.

Každý projekt má i své specifické cíle, rozsah a požadavky. Při jejich realizaci je třeba dbát nejen na požadavky na funkčnost, ale také na výkon, bezpečnost a uživatelskou přívětivost. Změny v IT odvětví se vyvíjejí rychlým tempem, což vyžaduje pružnost, adaptabilitu a schopnost rychle se přizpůsobit novým podmínkám a změnám v projektu. Často jsou kladeny vysoké nároky a očekávání, jak ze strany zákazníků a stakeholderů, tak i interního týmu.

Správné řízení projektu je klíčovým faktorem pro úspěch projektu a projektový manažer má v tomto procesu zásadní roli, protože je zodpovědný za vedení projektového týmu, správu zdrojů, monitorování pokroků v projekt a řešení případných problémů. Na začátku tohoto procesu však musí zvolit správnou metodu a přístup, který bude nejlépe reflektovat specifické potřeby projektu, zákazníka a týmu.

Existuje mnoho různých metod a rámců pro projektové řízení. Od těch tradičních, jako je Waterfall, který se soustředí na plánování a sekvenční realizaci jednotlivých fází projektu, přes agilní metody, jako je Scrum nebo Kanban, které se zaměřují na iterativní a inkrementální vývoj s důrazem na spolupráci a adaptabilitu, až po hybridní přístup. Každá z těchto metod má své výhody a nevýhody. Vybrat tu správnou metodu pro konkrétní projekt může být náročné.

Na trhu je dostupné značné množství literatury, která se věnuje této problematice, ale často se jedná o teoretické popisy, které nezohledňují specifika konkrétního projektu. Proto může být mnohdy i pro zkušené projektové manažery a týmy obtížné vybrat tu nejvhodnější metodu pro svůj projekt. Diplomová práce proto analyzuje některé dostupné zdroje a blíže prozkoumává jednotlivé metody. Na základě této analýzy a vlastní praktické zkušenosti autorka připravila průvodce,

který pomůže lépe porozumět hlavním rozdílům v jednotlivých metodách a přístupech, a tím umožnit nejen projektovým manažerům efektivně zvolit tu nejvhodnější metodu pro jejich konkrétní projekt.

2 Cíl práce a metodologie

Výběr správné metody pro řízení projektu je klíčovým rozhodnutím, které může mít zásadní vliv na výsledky projektu. Hlavním cílem této práce je proto shrnutí vybraných dostupných informací ohledně vodopádové a agilní metody rozšířené o zkušenosti autorky a jejích kolegů z praxe dodavatelské firmy a poskytnutí nástroje a doporučeného postupu při volbě metody pro efektivní řízení IT projektu dle jeho specifických parametrů.

V teoretické části se tato diplomová práce zaměří na obecný úvod do tématu, kdy je definován IT projekt, obvyklé fáze jeho životního cyklu, role v týmu na projektu. Pozornost bude věnována zejména roli projektového manažera a základním znalostem a dovednostem, kterými by měli projektoví manažeři disponovat pro úspěšnou realizaci IT projektu. Následně autorka nastiňuje základní principy různých metod pro řízení projektu a jejich silné a slabé stránky pro jednotlivé situace.

V praktické části diplomové práce bude na příkladu projektu z praxe autorky demonstrován proces výběru vhodné metody pro řízení projektu a popsány projektové procesy, které byly součástí vývoje konkrétního produktu. Pozornost bude věnována tomu, jaká kritéria jsou důležitá při výběru metody pro řízení projektu, a to na základě specifických faktorů a kritérií, jako jsou povaha projektu, tedy především jeho rozsah a komplexita, dále míra komunikace s klientem a jeho zapojení do projektu, týmová dynamika (velikost týmu, role, samoorganizace), a další faktory.

V diplomové práci jsou využity dvě hlavní metody výzkumu. První z nich je analýza dostupných zdrojů a informací o metodách řízení projektů a také analýza konkrétní části realizovaného projektu. Druhou metodou syntéza těchto poznatků spolu s osobními zkušenostmi autorky. Tyto informace byly následně zpracovány a shrnuty do přehledné tabulky, která představuje hlavní výstup výzkumu. Přínosem je zejména možnost jejího použití pro začínající IT projektové manažery, kterým může sloužit jako podklad pro rozhodování o volbě optimální metody pro jejich projekty. Samozřejmě může být užitečná i pro zkušené IT projektové manažery, kteří hledají jiné přístupy a metody pro zlepšení řízení svých projektů.

S ohledem na cílovou skupinu čtenářů se autorka rozhodla nepoužít žádný konkrétní nástroj pro rozhodování, ale výstup připravila ve formě tabulky, která je uživatelsky přívětivá a nenutí zájemce o danou problematiku učit se pracovat s dalším externím nástrojem.

Pro doplnění nutného kontextu je v praktické části diplomové práce popsáno také zadání ukázkového projektu, průběh projektu včetně projektových procesů a příkladu použití potřebných nástrojů (Confluence, Jira), se kterými autorka běžně pracuje. Následně jsou popsány také některé modelové rizikové situace, se kterými se může projektový manažer v průběhu řízení IT projektu setkat a které mohou být mitigovány správnou volbou metody řízení projektu.

Tato práce odráží konkrétní potřeby dodavatelské firmy na vytvoření podkladů pro začínající projektové manažery a výběr metody řízení projektu. Výsledky výzkumu a závěry prezentované v této práci se tak omezují pouze na metody řízení projektů, které jsou používány v dodavatelské firmě. Rovněž kritéria pro výběr metody a jejich indikátory jsou zvoleny na základě toho, co je běžně používáno v dodavatelské firmě, jejíž projekt autorka analyzuje. Jelikož každá firma a projekt může mít své vlastní unikátní požadavky, procesy a podmínky, které mohou ovlivnit volbu vhodné metody řízení projektu, nemusí tato práce kompletně obsáhnout všechny různé situace a varianty, které se mohou v průběhu projektu vyskytnout. Může nicméně posloužit jako inspirace a podklad nejen pro firmy podobného zaměření.

Všechny údaje týkající se dodavatelské firmy a jejího klienta jsou anonymizovány, aby nemohlo dojít k jejich identifikaci a byla zajištěna ochrana citlivých informací.

3 Vymezení IT projektového managementu

Pro vysvětlení projektového managementu existuje několik definic. Znění každé z definic je rozdílné, významem jsou však všechny velmi podobné. Projektový management je proces řízení zabývající se koordinací zdrojů, jehož pomocí lze dosáhnout požadovaného cíle (Němec, 2022). Tato činnost umožňuje efektivně plánovat, realizovat a dokončit libovolný typ projektu.

Rozdíl mezi klasickým projektovým managementem a IT projektovým managementem je pouze ten, že se v rámci projektového managementu řeší IT projekty. Definice IT projektu je popsaná v následující kapitole.

3.1 Definice projektu a IT projektu

Projekt je hlavním prvkem projektového řízení. Existuje více definic vysvětlujících tento pojem, význam definic však zůstává vždy téměř shodný.
Dle standardu IPMA je projekt definován jako *"jedinečný časově, nákladově a zdrojově omezený proces realizovaný za účelem vytvoření definovaných výstupů (rozsah naplnění projektových cílů) v požadované kvalitě a v souladu s platnými standardy a odsouhlasenými požadavky." Oproti tomu standard PMI definuje tento pojem jako „dočasné úsilí podniknuté pro vytvoření jedinečného produktu, služby nebo výsledku"* (Doležal a kol., 2016, s. 17).

Projekt je řízený proces, který má přesně stanovený začátek a konec, a existují pro něj jasná pravidla řízení a regulace. Projekt je složen z řady aktivit a úkolů, má určen cíl, který je nutno splnit na konci realizace projektu, a má stanoven rámec pro čerpání zdrojů potřebných pro jeho realizaci.

Výstupem neboli výsledkem projektu je produkt či služba. Aby bylo docíleno úspěšného výsledku, je potřeba dodržovat tři pilíře projektového managementu – čas, náklady a dostupnost zdrojů (viz Obr. 1). Čas je limitní faktor pro plánování dílčích aktivit projektu. Náklady jsou finančním projevem využívání zdrojů v časovém úseku. Posledním pilířem je dostupnost zdrojů (lidský kapitál, technologie, know-how apod.), které jsou vyčleněny na daný projekt. Je žádoucí dosáhnout úspěšného ukončení projektu, a proto je nutné usilovat o vyrovnanost těchto tří pilířů. K tomu slouží plán projektu, na jehož základě je projekt

koordinován. V ideálním světě by za pomoci dobře připraveného plánu existovala vysoká pravděpodobnost úspěšného dokončení projektu. V praxi je však rovnovážný stav často narušován různými rizikovými situacemi a externími faktory (Svozilová, 2011).

Obr. 1 – Základny projektového managementu
Zdroj: upraveno dle Svozilová (2011)

Jako IT projekt se obvykle označuje projekt, který se zabývá tvorbou nějaké informační technologie. Mezi tyto projekty patří například vývoj softwarových aplikací, tvorba informačních systémů, nasazování IT infrastruktury atp.

Rozdíly mezi byznys projektem a IT projektem jsou následující (Perens, 2011):

- Odolnost vůči změně – v IT projektech jsou změny velmi časté a posunují čas dokončení.
- Hodnocení postupu projektu – v IT projektech je přidaná hodnota vidět obvykle až na konci projektu, kde je vidět, zda je řešení funkční
- Nesrozumitelnost konečného výsledku – je těžké na začátku IT projektu představit klientovi, jak bude vypadat výsledný produkt.
- Přesnost zadání projektu – u IT řešení klient často neví, jaké funkcionality má softwarové řešení mít.

V porovnání se stavebním projektem, ve kterém je cíl projektu od začátku specifikován a pro stavebního experta jasný, v IT projektech naopak často dochází k situaci, kdy zákazník nemá zcela jasnou představu o tom, co by mělo být pomocí

softwarového řešení dosaženo a jak má aplikace fungovat. Také není zcela možné klientovi v IT projektu na začátku projektu ukázat, jak produkt bude vypadat, oproti stavbě, kde lze v počáteční fázi vytvořit jasnou představu, jak bude dům vypadat. Další slabinou IT projektu je časový odhad trvání projektu, jelikož během vývoje se může často měnit zadání a provádět doplňující vývojové práce, které značně oddalují termín dokončení projektu.

Pokud by se mělo porovnat hodnocení průběhu obou typu projektů, tak v IT projektu je výsledek obvykle vidět až na konci. Oproti tomu ve stavebním průmyslu je plnění projektu jasně vidět v průběhu toho, jak se dům staví. Výsledkem stavebního projektu je hmotný předmět, kdežto u IT projektu vzniká nehmotná hodnota. Projekt ve stavebnictví je od začátku vnímán jako pozitivní změna, což u IT projektu často jako negativní změna. Důvodem je, potřeba do nových změn uvést i koncové uživatele, kteří se budou muset s novým produktem naučit pracovat.

Nicméně v dnešní době se IT projekty objevují napříč všem odvětvím, jako například v bezpečnostních sektorech, zdravotnictví, automotive, školství atd. Tudíž přívětivost ke změnám se u zaměstnanců stále zvyšuje s jejich schopností se učit novým věcem.

3.2 Fáze životního cyklu projektu

Fáze životního cyklu projektu definují činnosti, které je potřeba provést v daný okamžik, včetně konkrétních výstupů požadovaných v dané fázi. V jednotlivých fázích je rovněž určeno, do jakých aktivit projektu bude kdo zapojen (Karešová, 2022).

Svozilová (2011) uvádí, že každý projekt prochází pěti fázemi. Tyto fáze lze vystihnout otázkami, jež napovídají, co je jejich podstatou:

1. Inicializace – „Čeho se má dosáhnout?“

V první fázi je projektu přidělen projektový manažer (PM). Klient a všechny zainteresované strany (tzv. stakeholders) definují rozsah projektu – rozpočet, cíl, časový plán atd.

2. Plánování a návrh řešení – „Co vývoj bude obnášet?“, „Jak vývoj bude probíhat a co bude potřeba udělat?“

V této fázi jsou již veškeré potřebné informace z předešlé fáze shromážděny, a je tak možné začít s plánováním, jehož výsledkem je časový harmonogram konkrétních aktivit, odhad nákladů a návaznost jednotlivých aktivit. V této fázi lze určit i možná rizika a zaznamenat je do tzv. Risk logu.

3. Realizace řešení – „Jak to zařídit?“

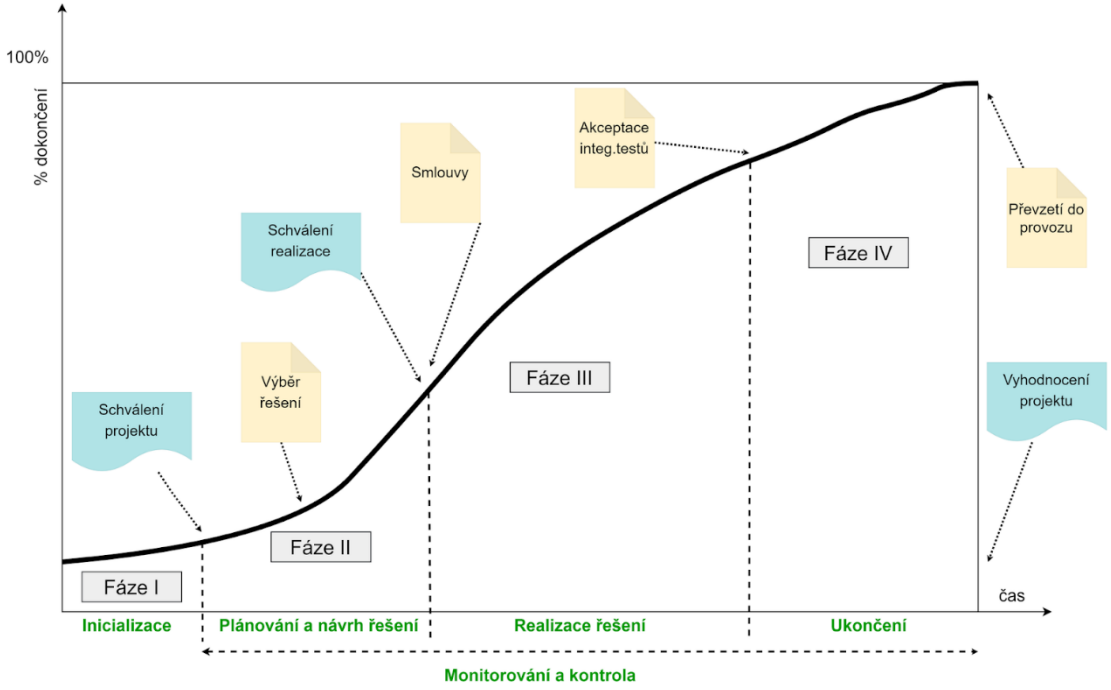
Ve třetí fáze, navazující na předchozí, je započata realizace projektu. V této fázi probíhá vytváření architektury produktu a stanovují se milníky projektu. Je nutné nastavit a dodržovat správnou komunikaci, a to nejen uvnitř týmu, ale i směrem ke všem zainteresovaným stranám. Důležitým úkolem je sledovat časový plán, rozpočet projektu a kvalitu produktu.

4. Monitorování a kontrola – „Dodržuje se plán a zadání?“

Monitorování a kontrola jdou ruku v ruce se všemi předchozími fázemi. Kontrola probíhá jak v průběhu, tak i na konci projektu. Během projektu pomáhá monitorování vyvarovat se odchylkám od plánu a včas vyřešit problémy tak, aby se výsledky co nejvíce shodovaly se stanoveným plánem. V této fázi je kontrolována jak kvalita procesů, tak i kvalita produktu.

5. Ukončení – „Jak projekt správně zakončit?“

Ve finální fázi PM ukončuje projekt a hodnotí výsledek. V tento okamžik je důležité celkovou práci, která byla na projektu provedena, zhodnotit. Mezi hlavní aktivity této fáze patří – nasazení produktu (tzv. release), prezentace nového produktu, komunikace se zainteresovanými stranami (tzv. stakeholder), sepsání informací o proběhlé práci, zdařilých úkolech i nevydařených aktivitách (tzv. lessons learned).



Obr. 2 – Fáze životního cyklu projektu
Zdroj: upraveno dle Šebek (2014)

Na Obr. 2 lze vidět grafické znázornění průběhu projektu, a klíčové aktivity, které se řeší na konci každé fáze, jako například sepsání smlouvy, akceptační protokoly, převzetí produktu do provozu.

Detailnější průběh projektu potom závisí na zvolené metodě řízení projektu, zda se půjde cestou Agile nebo Waterfall. Podrobnější popis a rozdíl mezi těmito metodami je popsán později v samostatné kapitole.

3.3 Role v týmu

Role v týmu se výrazně liší v závislosti na typu projektu a zvolené metodě řízení projektu. V této kapitole je definovaná základní sestava týmu, která se používá na téměř jakémkoliv projektu.

Byznys analytik

Byznys analytik je specialista, který zkoumá problém zákazníka, hledá řešení a formalizuje jeho koncept v podobě požadavků, kterými se vývojáři mohou řídit při vytváření nového produktu. Primární pracovní odpovědností byznys analytika

je komunikovat se zainteresovanými stranami, získávat a analyzovat získané informace, ale k tomu samozřejmě potřebuje mít znalosti v dané oblasti. Byznys analytik pracuje s požadavky ve všech fázích vývoje softwaru a hraje roli jakéhosi prostředníka mezi vývojovým týmem a zákazníkem.

Práci byznys analytika lze rozdělit do následných kroků. Nejdříve se vydefinují potřeby zákazníka a je důležité pochopit jeho problémy, které se mají řešit. Samostatně nebo s pomocí týmu formuluje koncept řešení a následně přemění koncept v technické úkoly se specifickými požadavky na budoucí produkt. K vytvoření těchto požadavků na IT byznys analytik používá různé techniky analýzy jako například: vytváření modelů procesů a struktur, prototypy uživatelského rozhraní a případy použití. Dále zadání musí být schváleno zákazníkem a předáno do vývojového týmu. Během celého vývoje byznys analytik je týmu k dispozici a v případě potřeby opět komunikuje se zákazníkem a probírá jeho potřeby.

Manažerské role

Na řízení IT projektů se podílí několik typů manažerů (Lucky Hunter, 2023), kteří mají na starost řízení dodávky IT řešení dle požadavků zadavatele/zákazníka. Níže autorka popisuje zejména typy manažerů, kteří se do projektu zapojují přímo, tj. řízením samotné IT dodávky a řízení vztahu s klientem.

Prvním z popisovaných manažerů je Projektový manažer. Je zodpovědný za úspěšné doručení projektu v dohodnutém čase a kvalitě. Jeho prací je zejména optimalizace dostupných zdrojů a řízení projektových nákladů dle daného rozpočtu.

Dalším typem manažera je Program manažer. V porovnání s projektovým manažerem, program manažer řídí větší celek portfolia projektu. Projektový manažer většinou řídí jeden projekt, který má jeden cíl. Program čili portfolio projektů se skládá z několika projektů, kde každý projekt má definován svůj cíl a výstupem má být určitý produkt. Proto program manažer je vlastně člověk, který řídí více projektových manažerů. Program manažer v první řadě komunikuje s projektovými manažery, aby věděl, v jakém stavu je program jako celek.

Třetím typem manažera je (Service) Delivery manažer. V některých firmách je delivery manažer totožný s projektovým manažerem. Ideálně by to však měl být člověk, který má technické znalosti a schopnost vyznat se v technické části projektu.

Zejména při produktovém vývoji, ale i v případech, kdy se jedná o dodávku na zakázku, je součástí týmu také role Produkt manažera, který důkladně zná produkt, a to včetně jeho marketingových aspektů, tj. nakolik je produkt zajímavý pro uživatele a co uživatelům přinese. Tato role vlastníka produktu někdy bývá také nazývána Product Owner, a je blíže definována v praktické části projektu.

Klíčovou roli při dojednávání parametrů a podmínek IT zakázky má (Key) Account manažer nebo také Business Development manažer. Jeho role na projektu je zásadní zejména ještě před jeho začátkem, když probíhá jednání o nabídce, rozpočtu a smluvním zajištění. Po dohodě nad podmínkami projektu bývá jeho role spíše eskalační. Tedy jeho zapojení roste v momentě, kdy se některé aspekty projektu rozchází s dohodnutými parametry, a není v silách PM tento problém vyřešit. Zároveň se stará o udržování vztahu s klientem, jeho spokojenost, a identifikuje další byznys příležitosti.

Do projektu mohou zasahovat samozřejmě další typy manažerů, kteří celý projektový tým ovlivňují nepřímo, proto výčet výše nemůže být kompletní. Jedná se například o roli Resource manažera, který je zodpovědný za rozvoj jednotlivých členů týmu, nebo další liniové manažery jednotlivých členů týmu.

Vývojář

Díky této roli může projekt opravdu vzniknout. Je to člen týmu, který má na starost vývoj produktu. Vývojáři se mohou lišit dle používaných technologií nebo takzvaných programovacích jazyků, pomocí kterých se vytváří software. Základní rozdělení vývojářů je Backend vývojář a Frontend vývojář. Backend vývojář vyvíjí logiku softwaru, kterou uživatel nevidí (pozadí aplikace) a Frontend vývojář tu stranu, kterou uživatel vidí, když software používá.

Tester

Tester je zodpovědný za kvalitu produktu. Tester převezme od vývojáře software pro testování a v testovacím prostředí tento software otestuje. Pokud software funguje, jak má, může vzápětí dokončený finální software “odcházet“ k zákazníkovi.

IT Operations (ITOps) tým

Jedná se o část týmu, která má na starost několik aktivit. ITOps zahrnují vše od síťových operací až po dohled nad virtuálními a fyzickými komponentami v rámci firemního prostředí. Nejčastější operace, které provádí ITOps role, jsou:

- spuštění řešení – zálohování dat, konfigurace serverů a obnova systémů po výpadku nebo aktualizaci,
- správa infrastruktury – správa cloudových dat, zabezpečení, sítí, řízeného hardwaru a softwaru,
- správa konfigurací – dokumentace hardwarových konfigurací, zavádění nových konfigurací pro optimální výkon IT služeb,
- řízení provozu IT – monitorování výkonu infrastruktury, provádění auditů infrastruktury, správa softwarových licencí,
- předvídání problémů – identifikace problémů, včasná detekce potenciálních problémů (Servicenow, 2023).

3.4 Definice IT projektového manažera

IT PM je role zodpovídající za dokončení projektu v požadovaném termínu s odpovídající kvalitou a předem definovaným finančním rozpočtem. Hlavním úkolem projektového manažera je řídit projekt jako celek a vhodně pracovat se zdroji, které má k dispozici. To zahrnuje koordinaci projektu a stanovení priorit jednotlivých kroků a úkolů, řešení všech vznikajících problémů a rizik, plánování a rozdělování úkolů mezi členy týmu, zajišťování potřeb svého týmu, kontrola kvality produktu a v neposlední řadě i práce s rozpočtem projektu.

Stát se PM není tak jednoduché, nestačí jen absolvování kurzu projektového řízení, ale je třeba si projít praxí a postupně tak získávat nejen explicitní znalosti a dovednosti, ale především ty tacitní, které jinak získat nelze. Rovněž je zapotřebí rozvíjet i své osobní charakteristiky a umění pracovat s lidmi. Více o dovednostech projektového manažera je popsáno v následujících odstavcích.

Všechny projekty se od sebe liší, a tak se liší i úkoly projektového manažera. Nicméně lze definovat některé základní úkoly manažera, které se vyskytují téměř ve všech projektech.

Plánování termínu dokončení

PM s pomocí byznys analytika či architekta musí dát dohromady požadavky klienta, zanalyzovat celkový obsah a objem této práce, a vydefinovat případná rizika a komplikace, která mohou nastat v průběhu vývoje. Pomocí těchto podkladů je manažer schopen stanovit kvalifikovaný odhad data dokončení jednotlivých milníků a následně i celého projektu. Dalším krokem bývá prodiskutovat odhadnuté termíny se zákazníkem a společně si je projít a vysvětlit jaké aktivity budou muset proběhnout, aby se daný milník naplnil.

Při plánování dokončení projektu je nutné také myslet na sestavu týmu. Ne vždy platí, že čím větší je tým, tím rychleji bude práce dokončena. Je potřeba myslet na to, že se práce v týmu rozděluje a je na sebe nějakým stylem vázaná. Některé části vývoje nemohou probíhat paralelně čili není možné, aby na jednom úkolu pracovalo několik lidí. Proto je při plánování termínů potřeba myslet na to, jak jsou určité fáze práce na sebe navázané a v jaké fázi kolik členů týmu může pracovat souběžně.

Koordinace projektu

PM by měl mít přehled o všem, co se na projektu děje. Je to nutné z toho důvodu, aby v případě nějakých komplikací mohl zasáhnout a efektivně problém vyřešit. Pokud nebude mít přehled, co se vše děje na jeho projektu od samého začátku, tak bez znalostí je velice těžké udělat správné rozhodnutí. Mezi důležité aktivity patří například organizování nákupu hardwaru či softwaru, koordinování jejich implementace, ujišťování se, zda je všechno správně nakonfigurováno a splňuje veškeré požadavky. Také je dobré, když PM sleduje i průběh ostatních projektů z portfolia. V případě že projekty v portfoliu jsou na sebe vázané (jeden projekt je prerekvizitou jiných), je důležité sledovat aktuální stavy prerekvizitních projektů, aby časový harmonogram jednotlivých projektů odpovídal realitě. Také je potřeba, aby PM komunikoval s ostatními zainteresovanými stranami, což je například zákazník nebo IT partner a dodával jim aktuální informace o stavu běžícího projektu.

Řešení problémů

Řešení problémů je dovednosti projektového manažera, která usnadňuje efektivní řešení problémů kombinací kreativního myšlení a silných analytických dovedností. PM by měl být schopen pomocí týmu identifikovat problém, sepsat seznam všech možných řešení, vyhodnotit řešení, vybrat a zdokumentovat nejlepší řešení a vytvořit plán opatření. Ať je problém jakýkoliv, používání systematického přístupu k řešení problémů může pomoct projektovému manažerovi být efektivnějším.

Vedení dokumentace

Je velmi důležité sbírat veškeré informace o projektu a mít je zaznamenané v dokumentaci. Nemusí to být nutně pouze dokumentace, ale mohou to být i důležité emaily, ve kterých se řešila nějaká problematika. Velmi často se na projektech stává, že zákazník změní v průběhu názor, a poté je jeho nové tvrzení v rozporu s tím předchozím (tvrdí, že se na začátku domluvili na něčem jiném). Pro takové případy je dobré mít argumenty a podklady vzájemné domluvy ve formě sepsaného zápisu, kde lze ověřit zodpovědnosti zapojených osob. V praxi se také často stává, že klient odsouhlasil změny na projektu, ale zapomněl je, a proto je důležité mít vše podložené důkazy.

Práce s rizikem

PM by se měl snažit předvídat veškeré pozitivní a negativní události, které by mohly narušit průběh nebo dokončení projektu. Proto je důležité včas zasáhnout a navrhnout řešení, jak by se dalo těmto komplikacím předejít. Jeden z takových běžných problémů může být nemoc zaměstnance (proto je třeba dbát na zastupitelnost v týmu), nedostatek zařízení nebo jejich porucha, výpadek internetu atp.

Sledování rozpočtu

Je důležité kontrolovat rozpočet projektu a vědět, jakou rychlostí se rozpočet čerpá. V případě, kdy je projekt typu Fix Time Fix Price (FTFP), je důležité dbát

na to, aby byl rozpočet správně rozdělen a využíván. V takovém modelu spolupráce se dodavatel zavazuje k dokončení projektu v určeném čase a za pevnou cenu, a to bez ohledu na množství času nebo výši nákladů nutných k dokončení projektu. V případě, kdy se rozpočet vyčerpá, je nutné prokonzultovat tuto skutečnost s klientem a sdělit mu, kolik finančních prostředků se vyčerpalo a s jakým výsledkem, poté ho případně požádat o navýšení rozpočtu.

Sledování tempa práce

PM je zodpovědný za to, aby všechny úkoly byly dokončeny včas a v co nejlepší kvalitě. Nejčastěji se k tomu používají různé nástroje, jako například:

Jira – nástroj pro management úkolů, jejich sledování a vizualizaci, zaznamenávání odpracovaných hodin, dokumentaci toho, co a jak bylo zrealizované a zaznamenání toho co ještě zbývá udělat.

MS Project – nástroj, který pomáhá při sledování průběhu a případně posunu původního harmonogramu projektu, tzn. Jaké úkoly a v jakém časovém horizontu se mají vyřešit, kolik zaměstnanců se zapojí na realizaci daných úkolů a na jak dlouho.

Confluence – nástroj, který slouží jako znalostní báze pro podporu týmové spolupráce a sdílení veškerých potřebných informací na jednom místě.

Schůzky fyzické nebo přes komunikační kanály – z pohledu PM slouží k udržení tempa práce, k validaci plnění cílů, připomenutí a zdůraznění důležitosti dodržování termínů, k rychlému odpovídání na otázky a řešení případných problémů, k diskutování nad řešením úkolů nebo také k získání zpětné vazby od týmu.

Co se týká práce projektových manažerů tak přibližně 80 % jejich pracovního času zabírají schůzky, plánování a monitorování projektu, a zbylých 20 % na emailovou, telefonní a chatovou komunikaci (Solontai, 2022).

4 Znalosti IT projektového manažera

PM potřebuje mít určitý soubor měkkých a tvrdých dovedností, ale spíše než pro svou profesi, tak pro odvětví, ve kterém pracuje. Dále je zásadní, aby manažer měl povědomí o standardech a normách projektového managementu, které mu pomohou v řízení jakéhokoliv projektu. Existuje několik standardů, kterými by se měli řídit manažeři všech typů. Hlavní metody a standardy jsou zmíněny v samostatných kapitolách.

4.1 Základní znalosti a dovednosti IT projektového manažera

Jak bylo zmíněno výše, každý PM musí mít soubor měkkých a tvrdých dovedností. Dle nejlepšího PM roku 2019 v soutěží Ukrainian IT Awards, Tarasa Fedoruka (АБЛИЦОВА, 2021), mezi **měkké dovednosti** projektového manažera patří:

Umění se přizpůsobit

Pro úspěšné řízení projektu je nutné, aby PM dokázal efektivně koordinovat a řídit různorodé úkoly. PM musí být schopný zajistit přiřazení úkolů členům týmu a současně zpracovávat své vlastní úkoly. Zároveň PM sleduje a analyzuje průběh projektu, předvídá jeho budoucí vývoj a kalkuluje zbývající aktivity na projektu. Dohlíží na plnění úkolů a pravidelně informuje klienta a management o aktuálním stavu projektu. Od PM je tedy vyžadována schopnost efektivního plánování, organizace a koordinace, stejně jako schopnost řídit současně více úkolů a zvládat stresové situace.

Tolerance stresu

Řídit velké množství úkolů je velmi stresující, a proto musí mít manažer vysokou toleranci vůči stresu. Mimo toho, že PM řeší řadu různorodých úkolů a problému, tak ještě komunikuje s lidmi, což je samo o sobě náročná aktivita. Proto je velmi důležité udržet neutralitu a problémy řešit v klidu. PM se tedy musí naučit zvládat stresové situace a pečovat o svou mentální hygienu. Nesmí se také zapomínat

na fakt, že PM musí umět pracovat nejen se svými emocemi, ale i s emocemi ostatních lidí.

Schopnost se rozhodovat

Během své práce PM provádí různá rozhodnutí, jak malá, tak i kriticky ovlivňující projekt. Ne vždy je jednoduché k nějakému rozhodnutí dojít, a pokud rozhodnutí udělá, tak o správnosti rozhodnutí se přesvědčí až později, což je také velmi stresující. Nicméně PM by se neměl bát rozhodování, ale brát jej jako část své zodpovědnosti.

Schopnost komunikovat

Manažer musí umět správně formulovat své myšlenky, umět klást správně otázky, a umět vyzdvihnout hlavní myšlenku z odpovědi. Dále je také důležité, aby PM uměl správně předávat informace o stavu projektu zákazníkovi, technickému týmu či managementu a s každým uměl komunikovat na jiné technické úrovni. Dále by měl umět pracovat s očekáváními, jelikož jak u vývojového týmů existují jistá očekávání, tak především u stakeholderů bývají zpravidla odlišná očekávání o výsledku. Proto je důležité, aby se PM pokusil tato očekávání sjednotit a aby šli všichni „jednou cestou“ a za stejným cílem.

Cizí jazyk

V dnešní době díky globalizaci a propojení firem skrze celý svět má značnou důležitost i umět cizí jazyk (například angličtinu). Spousta informací a dokumentací se eviduje právě v anglickém jazyce, a proto je klíčové pro projektového manažera umět tento jazyk.

Mezi **tvrdé dovednosti** patří schopnosti, které pomáhají projektovému manažerovi vypořádat se s pracovními povinnostmi a odpovědnostmi. Tvrdé dovednosti je možné se naučit prostřednictvím kurzů a odborného cvičení či je lze získat praxí. Tyto dovednosti jsou obvykle zaměřeny na konkrétní úkoly a procesy, jako je použití nástrojů nebo softwarů. Mezi tyto dovednosti patří:

Práce se software

Mezi základní software, se kterými musí PM umět pracovat, patří Microsoft Office, nástroje od Adobe, Jira a Confluence od Atlassianu a MS Project. Existuje široká škála toho, s čím by měl umět PM pracovat, ale zde velmi záleží, s jakými nástroji pracuje daná společnost a který nástroj projektový manažer potřebuje pro daný projekt. V dnešní době je na internetu mnoho zdrojů zabývajících se tím, jaký nástroj je pro daný účel vhodný.

Znalost metod projektového řízení

Hlavními metodami řízení IT projektu jsou Waterfall, Agile a jiné hybridní metody nezbytná je znalost především agilních metod zvaných Scrum a Kanban. PM by měl být schopen vyzdvihnout plusy a mínusy jednotlivých metod a zvolit tu správnou pro svůj projekt.

Technický background

Samozřejmě ideálním projektovým manažerem by byl bývalý vývojář nebo tester. Jelikož tito lidé se vyznají v technickém prostředí, na pozici projektových manažerů by byli schopni svému týmu pomáhat. Ale říká se, že čím více toho PM umí, tím dražší je, a ne každá společnost má možnost zaměstnávat drahého projektového manažera a využívat veškeré jeho znalosti. Kolikrát společnostem stačí, když PM umí pracovat s informacemi, které mu dává technický tým.

4.2 Základní technické pojmy pro IT projektového manažera

Jak již bylo zmíněno v předešlé kapitole, tak ne každý IT PM je bývalý tester nebo vývojář se silnými technickými znalostmi. Pokud ale PM nemá technické základy vůbec, tak je třeba, aby tyto mezery doplnil a dokázal se vyznat v základních technických pojmech. Níže jsou sepsané pojmy, se kterými se obvykle v IT firmách PM setkává na denní bázi. Popisy jsou zde stručně uvedeny a více informací o jednotlivých pojmech lze nalézt ve zde uvedené literatuře.

Application Programming interface (API) – rozhraní pomocí kterého vývojář volá příkazy a funkce programu z jiné aplikace (Lutkevich, 2022).

Application Server – virtuální server, který slouží pro provoz sdílených aplikací třetích stran (Pospíšil, 2021).

Build – proces převodu souborů zdrojového kódu (kompilace) pro vytvoření verze softwarového programu / aplikace (Christensson, 2021).

Continuous Integration (CI) – proces slučování změn vývojového kódu zpět do hlavní vývojové větve (Pittet, 2023).

Continuous Deployment (CD) – automatizované a průběžné nasazování softwarové funkcionality (Pittet, 2023).

Commit – při této operaci dochází k odeslání nejnovější změny (verze) zdrojového kódu do úložiště (Microsoft, 2022a).

Cross-origin Resource Sharing (CORS) – mechanismus, který podporuje zabezpečené požadavky a přenosy dat z vnějších zdrojů, jako například domény, schémata nebo porty. Díky CORS prohlížeč zajišťuje, že útočníci nemohou získat citlivá data pomocí požadavků z různých zdrojů (SuperToken team, 2022).

Database – soubor strukturovaných informací nebo dat, uložených elektronicky v počítačovém systému. Lze s nimi manipulovat pomocí databázového systém (Oracle, 2023a).

Deployment Process – proces zahrnující všechny kroky a aktivity, které jsou nutné ke zpřístupnění softwarového systému uživatelům (SumoLogic, 2023).

Docker – je open source nástroj, který pomáhá vývojářům při vytváření, nasazení nebo spouštění aplikací s pomocí kontejnerů. Kontejnery umožňují vývojáři zabalit aplikaci do jednoho celku včetně knihoven a nasadit ji jako jeden balíček (Bigelow, 2020).

Git – nástroj DevOps používaný pro správu zdrojového kódu. Jedná se o systém správy verzí, který se používá k efektivnímu zpracování projektů všech velikostí. Git se používá ke sledování změn ve zdrojovém kódu (Sheldon, 2023).

Grafical User Interface (GUI) – forma uživatelského rozhraní, která uživatelům umožňuje komunikovat s elektronickými zařízení, jako jsou počítače a smartphony, a to pomocí ikon, nabídek a dalších vizuálních indikátorů nebo grafiky (Rouse, 2021).

Jenkins – automatizační nástroj, který se používá k neustálému vytváření a testování softwarů, což vývojářům usnadňuje integraci změn do projektu. Pomáhá urychlit proces vývoje softwaru pomocí automatizace (Edureka, 2022).

Merge – proces sloužící ke sloučení více nezávislých vývojových linii do jedné větve (Atlassian, 2023).

Microservices – nativní architektonický přístup, ve kterém se jedna aplikace skládá z více volně propojených a nezávisle nastavitelných menších komponent nebo služeb (AWS, 2023).

Pipeline – proces přebírání kódu ze správy verzí a jeho zpřístupnění uživatelům aplikace automatizovaným způsobem. Když tým vývojářů pracuje na projektech nebo funkcích, tak potřebuje spolehlivý a efektivní způsob vytváření, testování a nasazení své práce (Atlassian, 2023a).

Pull Request (PR) – akce v nástroji Git, kdy přispěvatel požádá správce úložiště Git, aby zkontroloval kód, který chce sloučit do projektu (Atlassian, 2023b).

Pull – příkaz, který se používá k načtení a stažení obsahu ze vzdáleného úložiště a okamžité aktualizaci místního úložiště, aby odpovídalo tomuto obsahu (Atlassian, 2023c).

Push – příkaz, který se používá k nahrání obsahu místního úložiště do vzdáleného úložiště (Microsoft, 2022b).

Request – objekt obsahující informace nezbytné pro vyvolání metody na serverové straně. Mezi tyto informace mohou patřit například URL adresa, metoda HTTP, hlavičky a tělo požadavku a další metadata (IBM, 2021).

REST API – rozhraní, sloužící pro předávání dat mezi dvěma, či více aplikacemi (RedHat, 2020).

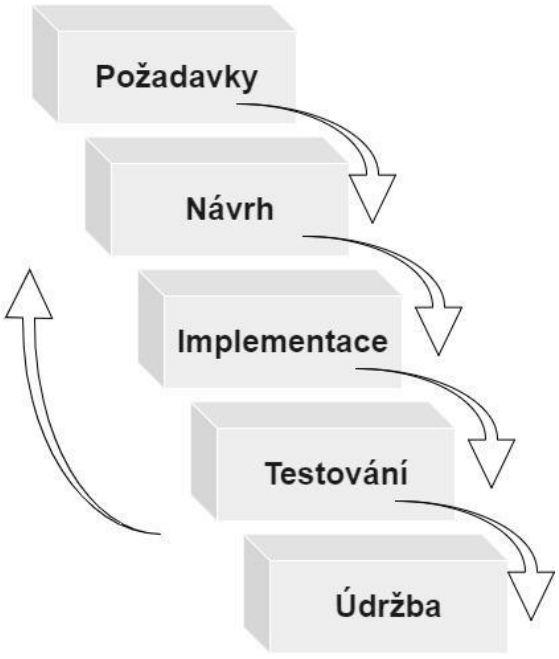
Seznam není zcela kompletní, jelikož existuje nespočet pojmů, které se vyskytují v IT „světě“. Seznam pojmů se může lišit dle firem a dle toho s jakými technologiemi IT firma pracuje. Nicméně tento seznam je základem, který by PM měl znát již při vstupu do řízení IT projektů.

5 Metody řízení projektu

Úspěch projektu při tvorbě softwaru úzce souvisí se zvoleným přístupem vývoje. Agile a Waterfall jsou v současnosti dvě nejpopulárnější SDLC (Systems Development Life Cycle) metodologie vhodné pro vývoj softwaru. Ačkoliv jsou oba modely v mnohých přístupech podobné, v několika ohledech se podstatně liší. V následujících kapitolách jsou tyto dvě metodologie, spolu s jejich silnými a slabými stránkami, popsány.

5.1 Waterfall model

Waterfall je jedna z historicky nejstarších metod projektového řízení. Je to lineární, sekvenční přístup k návrhu projektu, kde se pokrok pohybuje jedním směrem od shora dolů, tedy jako vodopád. Tato metoda je založena na pečlivém plánování, podrobné dokumentaci a následné realizaci. Skládá se z několika samostatných na sebe navazujících fází. Waterfall řízení projektu neumožňuje vrátit se do předchozího kroku. Jedinou možností, jak se lze dostat do předchozích fází, je začít znovu od fáze první (Horry, 2022).



Obr. 3 – Waterfall
Zdroj: upraveno dle Waseem (2022)

Tato metoda byla nejprve používána v softwarovém odvětví. Pro porozumění Waterfall řízení je možné použít následující příklad ze stavebnictví. V momentě, kdy jsou základy stavby hotové, je teprve možné postavit zdi. Až po postavení zdí lze osazovat okna a postavit střechu. Tyto fáze nemohou být zaměněné, jelikož jsou na sebe návazné.

Ze zjevných důvodů je ve Waterfall metodě velmi dbáno na důležitost plánování. Požadavky projektu musí být předem jasně definovány, a každý, kdo se na projektu podílí, s nimi musí být obeznámen. Od počátku projektu musí každý člen týmu znát svou roli včetně zodpovědností, které daná role obnáší. Všechny tyto informace musí být důkladně zdokumentovány a poskytnuty týmu před začátkem projektu (Waseem, 2022).

Důkladná dokumentace je proto prioritou Waterfall metody. Měla by kromě výše zmíněného obsahovat také veškeré kroky vývojového procesu – specifikaci byznys požadavků, popis architektury včetně datových modelů a technické specifikace, dokumentaci zdrojového kódu, testovací scénáře, informace o opravách chyb. Pro zajištění správného směřování všech zúčastněných, je nutné doplňovat dokumentaci v každé fázi projektu. Členové týmu by měli umět pracovat s dokumentací a odkazovat se na ni. O jednotlivých fázích ve Waterfallu pojednávají následující odstavce.

Systémové a softwarové požadavky

V první fázi vývoje softwaru je důležité zdokumentovat všechny požadavky klienta. Je důležité se všemi zainteresovanými stranami projednat podrobnosti projektu (termíny, finanční rozpočet, výběr vhodných technologií a postupů atp.) a všechny tyto získané informace je nutné zdokumentovat. Výsledkem této etapy by mělo být vytvoření specifikace definující všeobecné funkcionální požadavky zákazníka a popis IT infrastruktury klienta.

Návrh softwaru

Vývojový tým přistoupí k vypracování podrobných zadání. Tým prodiskutuje s klientem logiku systému, detaily hotového produktu. Dále se vytvoří popis

funkčnosti (co a jak má fungovat), připraví se prototyp a navrhne se rozložení uživatelského rozhraní tzv. Grafical User Interface (GUI). Následně se začne jednat o rozpočtu, určí se pracnost v člověkodnech (mandays, MDs), a sestaví se tým. Výsledkem této přípravné fáze je rozsáhlá dokumentace, avšak není ještě zcela vyřešena otázka implementace a veškerá technologická omezení.

Implementace

V této fázi se obvykle zpočátku začne řešit otázka, jak přesně bude vývoj probíhat, jaké nástroje bude tým používat, jaké programovací jazyky nebo vybavení budou využívat, pokud to nebylo vydefinováno již v předchozí fázi. Dokumentace sestavená v předchozích fázích se rozšiřuje o implementační detaily a začíná samotný vývoj daného řešení, kde vzniká první GUI produktu. Třetí fáze zabírá většinou nejvíce času projektu – programátoři píší kód, designéři kreslí design a editoři píší texty. V případě Waterfall přístupu by veškeré práce měly probíhat přesně dle zadání a měl by vznikat tlak na dodržení termínů, funkčních požadavků a odpovídající kvality.

Verifikace a validace – testování

Produkt je připraven k prvnímu použití, a proto začíná fáze testování. Obvykle v této fázi začínají vycházet najevo problémy a je třeba je odstranit. Pokud jsou v kódu kritické chyby, je potřeba je ihned opravit, což může trvat i několik týdnů. V tento moment lze vidět nevýhody Waterfall přístupu, jelikož testování neprobíhalo soustavně v průběhu vývoje, ale až po dokončení všech funkcionalit, a tak je zde větší riziko náročnějších a komplexnějších oprav. Kdyby byly použity flexibilní metody, kritickým chybám by se dalo předejít dříve a díky tomu ušetřit čas, finance i lidské zdroje. Je zcela možné, že se ve fázi testování může zjistit, že se projektu věnovalo spoustu času a financí, ale i přesto není produkt funkční. Poté, co je produkt otestován a chyby jsou opraveny, je produkt uveden do provozu.

Údržba

K tomu, aby se vyloučily další problémy, je potřeba po nasazení softwaru do provozu jej určitou dobu pozorovat, zda vše korektně funguje, a v případě identifikace problému, jej ihned odstranit. Po dohodě s klientem je sestaven technický tým, který má na starosti podporu a údržbu softwaru (Horry, 2022).

Každý proces řízení podléhá souboru zásad. Při startu nového projektu a zvolení vhodného modelu pro řízení projektu, je potřeba zjistit, zda principy Waterfall řízení odpovídají potřebám projektu.

Zde jsou tři základní principy, kterými se řídí Waterfall model.

Sekvenční struktura – model rozděluje projekt do sekvenčních fází. Do další fáze projektu lze přejít až poté, co je dokončena ta aktuální. To znamená, že není prostor pro změnu či úpravy výstupů z této fáze po jejím dokončení. Jediný způsob, jak se vrátit, je začít znovu.

Minimální zapojení zákazníka – model vyžaduje minimální interakci se zákazníkem. Je to z toho důvodu, že vývoj začíná až poté, co jsou definovány požadavky a cíle zákazníka.

Rozsáhlá dokumentace – tato metoda zahrnuje podrobnou dokumentaci všech požadavků procesů vývoje a konečného výsledku. Vzhledem k tomu, že se zákazníkem během projektu probíhá minimální nebo žádná komunikace, tak je nutné předem zdokumentovat každý důležitý detail (např. jak by mělo tlačítko vypadat, co se provede při kliknutí na něj atd.)

Z výše popsaných fází a principů je tedy potřeba vyzdvihnout hlavní pravidla Waterfall metody, kterými je třeba se řídit v případě zvolení této metody pro řízení projektu:

1. Specifikace požadavků a technická dokumentace je velmi důležitá.
2. Každá další etapa by měla začít až po skončení předchozí etapy.
3. Nelze vynechat žádnou etapu.
4. Je téměř nemožné se vrátit k předchozí etapě (např. za účelem provedení úpravy.)
5. V případě, že se požadavky změnily, je nezbytné upravit zadání a začít znovu od první etapy.
6. Chyby z vývoje se opravují až během fáze testování, nikoliv průběžně.

7. Klient se nemusí účastnit celého vývojového procesu, jeho účast je zásadní při úvodu a konci projektu (při počátečnímu sepisování a ladění požadavků a předávce hotového produktu).

Jsou to zcela jasná pravidla, ale je zapotřebí na začátku projektu vše důkladně promyslet a naplánovat, jinak hrozí, že plynulý přesun z jedné fáze do druhé bude komplikovaný nebo také nereálný (Waseem, 2022).

Silné a slabé stránky Waterfall metody

Každá metoda vývoje projektu má své klady a zápory, proto je pro některé projekty více vhodná než pro jiné.

Výhodou vývoje stylem Waterfall je možnost rozdělení a kontroly fází. Spolu s časovou osou aktivit pro každou etapu vývoje lze rozvrhnout harmonogram tak, aby produkt mohl plynule přecházet z jedné etapy do druhé. Každá etapa projektu probíhá v přesně určeném pořadí.

Na základě praxe a odborných konzultací s kolegy ve firmě dodavatele autorka mezi další výhody modelu Waterfall řadí následující:

- postupné zpracování projektových fází (jedna po druhé),
- snadná organizace úkolů v každé fázi,
- exaktně definované milníky a časové osy rozvoje projektu,
- přesně stanovené nacenění projektu (díky fixnímu harmonogramu),
- podrobně zdokumentovaný proces a výsledky,
- vhodnost pro projekty s malým rozsahem a jasně definovanými požadavky.

Nevýhodou modelu Waterfall je neposkytnutí času pro zásahy či změny, se kterými se na začátku vývoje nepočítalo. Když se projekt nachází ve fázi testování, je velmi náročné vrátit se a změnit něco, co nebylo zváženo a zdokumentováno ve fázi vývoje.

Hlavní nevýhody modelu jsou následující:

- zvýšení rizik a nejistoty v dodání očekávané hodnoty,
- nemožnost průběžné adaptace na měnící se byznys požadavky,

- nemožnost identifikace technologických problémů v rané fázi (před finální integrací jednotlivých částí),
- obtížně měřitelný pokrok v jednotlivých fázích,
- hrozba zastavení projektu při výskytu zásadní změny v průběhu životního cyklu projektu,
- nevhodnost modelu pro složité projekty (McCormick a kol., 2012).

5.2 Agilní model

Agilní metoda neboli Agile je jedním v současnosti z nejpopulárnějších přístupů projektového řízení na světě. Tato metoda je užitečná v případech, kdy je využíván inkrementální a iterativní postup, což znamená, že jsou požadavky zadávány průběžně a řešení postupuje plynule díky dobře nastavené spolupráci zákazníka a vývojového týmu. Původně byl Agile vytvořen za účelem vývoje IT produktů v reakci na nedokonalosti Waterfall modelu, jehož pracovní postupy se nemohly dostatečně rychle přizpůsobit neustále se měnícímu trhu.

Základem Agilní metody je Agile Manifest, který se skládá ze čtyř základních hodnot (Agile Alliance, 2022):

1. Jednotlivci a interakce před procesy a nástroji – dávat přednost komunikaci mezi členy týmu před používáním příliš striktních procesů a nástrojů. Komunikace a dobrá spolupráce v týmu může vést k lepším výsledkům, a to i díky příjemnému prostředí, ve kterém se tým cítí podporován a motivován ke spolupráci.
2. Fungující software před vyčerpávající dokumentací – tým by měl svou energii zaměřovat na tvorbu funkčního softwaru namísto vytváření podrobných a často zastaralých dokumentací. Vývojáři by měli pracovat na tvorbě funkčních prototypů, které slouží jako základ pro další iteraci, místo vytváření podobných dokumentů, které mohou brzdit agilní vývoj a přinášet neefektivitu.
3. Spolupráce se zákazníkem před vyjednáváním o smlouvě – tým by měl dostávat průběžnou zpětnou vazbu od zákazníka, aby lépe mohl porozumět jeho potřebám a požadavkům. Díky tomuto přístupu tým může rychle

reagovat na změny v požadavcích a být schopen lépe se přizpůsobit potřebám zákazníka.

4. Reagování na změny místo striktního dodržování plánu – vývoj softwaru má tendenci být flexibilní a projektový tým by tak měl být schopen během vývojového procesu rychle reagovat na změny namísto striktního dodržování plánu. Manifesto klade důraz na flexibilitu a otevřenost – klíčové prvky úspěšného vývoje softwaru.

Agile nemá stanovené předpisy pro realizaci projektů. Je pouze souborem principů, které zajišťují důsledný vývoj projektu rozdělující celý cyklus na krátké etapy. V momentě, kdy jsou všechny etapy dokončeny, je projekt považován za úspěšně realizovaný.

Na základě těchto principů byly vyvinuty pracovní rámce jako je Scrum, Kanban, Lean a další. Tato diplomová práce se zabývá pouze prvními dvěma zmíněnými rámci – Scrumem a Kanbanem, které jsou nejvíce využívány v praxi autorky ve firmě dodavatele.

Scrum

Scrum je metoda, která byla vyvinuta primárně pro IT projekty a je založena na sprintech. Sprint je iterativní časový rámec, ve kterém je dosaženo dílčího cíle, který byl stanoven pro daný časový úsek. Nejčastěji je sprint nastavován na období 14 dní. Cílem Scrumu je vyvíjet, dodávat a podporovat komplexní projekty prostřednictvím neustále spolupráce mezi klientem a vývojáři, čímž je odpovědnost převedena na stranu klienta vystupujícího v roli soustavného kontrolora. Scrum se liší od jiných metodologií tím, že má specifické role, artefakty a události, které jsou popsány v kapitole níže.

Role

Pro pochopení procesu Scrum je důležité znát role, které se v procesu vyskytují.

Product Owner – je vlastníkem produktu. Má na starosti vydefinovat cíl a vize projektu. Product Owner stanovuje priority, rozhoduje, jaká funkcionalita má přednost a začne se na ni v rámci sprintu pracovat. Měl by být dle potřeby týmu

k dispozici, ale oproti Scrum Masteru nemá tak blízký kontakt s týmem. Neměl by týmu přikazovat, co má dělat a čím má začít. Jeho úkolem je pouze určit prioritu jednotlivých funkcionalit, a tým by měl následně pracovat samostatně na základě přiřazených priorit. Role PO, vyžaduje komunikativního člověka se silnou znalostí produktu a procesů (Šochová a Kunce, 2014).

Vývojový tým – je skupina IT specialistů z různých oblastí s potřebnými dovednostmi vyvinout požadovaný produkt. Může se skládat z programátorů, testerů, UI designerů, architektů, analytiků, DevOps specialistů, síťařů, databázistů a dalších. Tým je řízen PM/Scrum mastrem a vývojovým leaderem, kteří se snaží o dosažení maximální efektivity, vytíženosti jednotlivých kapacit a dodržení všech náležitostí (kvalita, termín, technologie, finance). Jednotliví členové týmu by měli být schopni si svou práci efektivně organizovat sami, ale vždy by nad nimi měl někdo dohlížet na postup a kvalitu jejich práce.

Velikost týmu by měla být dostatečná na to, aby mohl tým provádět práci v požadovaném objemu a kvalitě a zároveň měl všechny potřebné dovednosti. Nesmí být ale příliš velký, aby se zbytečně nezvyšovalo množství interakcí mezi jednotlivými členy týmu nebo nedošlo k neefektivitě využívání pracovních zdrojů. Proto bývá doporučená velikost týmu 5-9 členů, jak se uvádí Rubin (2012) a Schwaber a Sutherland (2020).

Scrum Master – osoba, která pomáhá všem zúčastněným v projektu pochopit hodnoty a principy Scrumu. Dále se stará o to, aby byl Scrum efektivní a fungoval. Má na starosti jeho dodržování, ale také má právo inicializovat změnu, která pomůže zefektivnit některý z procesů. Scrum Master by měl být komunikativní, vnímavý a v případě potřeby by měl utlumovat konflikty v týmu. Osoba v této roli dělá vše pro to, aby tým byl produktivní a samoorganizační (Šochová a Kunce, 2014).

Artefakty a události

Product Backlog – seznam funkčních požadavků na produkt, ze kterého se průběžně vybírají položky do Sprint Backlogu.

Sprint Backlog – seznam úkolů a požadavků, které je potřeba dokončit během dalšího sprintu. Sprint Backlog má být naplněn dle kapacity týmu.

Sprint Review – neformální setkání konané na konci každého sprintu, kde tým vyhodnotí práci během sprintu, ověří se, zda se naplnily cíle sprintu a které aktivity se případně posunou do dalšího sprintu.

Story Points – měrná jednotka, která se používá pro odhad náročnosti implementace úkolů.

Definition of Done (DoD) – přesná specifikace požadovaného výstupu práce. Na konci sprintu se práce nepovažuje za hotovou, pokud vyvinutá funkcionalita softwaru nesplňuje DoD. Tato definice dodává úkolům transparentnost tím, že poskytuje všem členům sdílené pochopení toho, jak by výsledek práce měl vypadat a zda byla práce dokončena v rámci sprintu. Pokud se na Sprint Review rozhodne, že není splněno DoD u některé funkcionality, je potřeba tuto funkcionalitu přesunout do dalšího sprintu, ve kterém bude dokončená.

Increment – funkční přírůstek produktu, který vzniká po ukončení sprintu. Přírůstek musí splňovat kritéria DoD, která byla definovaná na začátku sprintu.

Sprint – iterativní časový rámec, ve kterém je dosaženo dílčího cíle, který byl stanoven pro daný časový úsek (Sprint). Termíny sprintu by neměly přesahovat jeden kalendářní měsíc a jsou konzistentní během celého vývoje. Nejčastěji je sprint nastavován na období 14 dnů, ale to záleží na granularitě vyvíjených funkcionalit. Čím je větší náročnost jedné funkcionality, tím delší sprint se nastavuje.

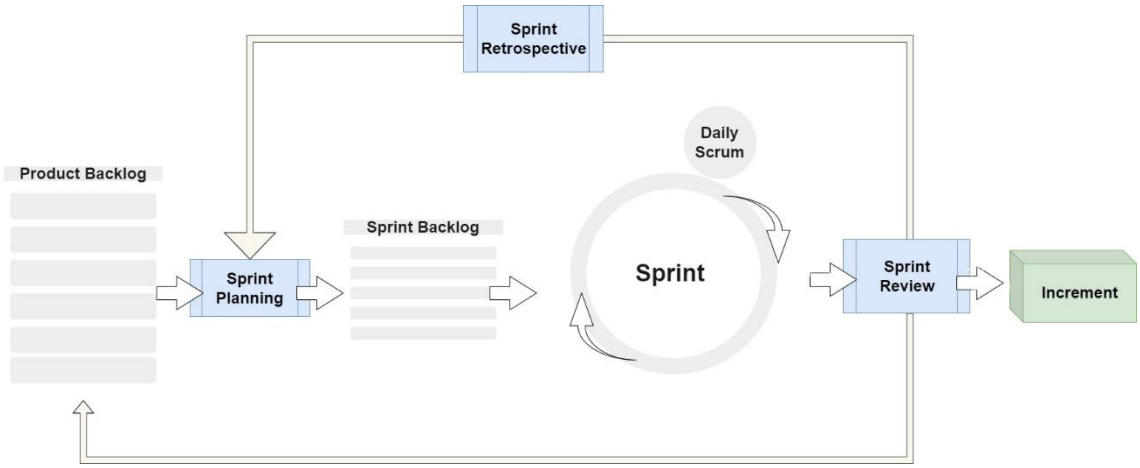
Daily Scrum/Stand-up meeting (SU) - krátké setkání konané ve stejnou dobu každý den sprintu, kde se diskutuje o tom, co se udělalo od poslední schůzky, a co se plánuje udělat do další schůzky. Doba trvání schůzky se určuje dle velikosti týmu. Kolik lidí se zúčastní schůzky, tolik minut by měla trvat.

Sprint Planning – schůzka na začátku sprintu, během které se tým s Product Ownerem domluví, jaký je cíl nadcházejícího sprintu. Při tomto setkání se představují nové funkcionality, které byly vytvořeny v průběhu uplynulého sprintu. Dále se přidávají z Backlogu další úkoly do nového/plánovaného sprintu na kterých se bude v následujícím období (Sprintu) pracovat a do Backlogu se mohou rovněž přidávat nové požadavky klienta.

Sprint Retrospective – schůzka, která poskytuje příležitost ke kontrole a přizpůsobení procesů. Tým, Scrum Master a někdy i Product Owner zde diskutují o tom, co probíhalo dobře v minulosti co se týče komunikace, vztahů, procesů

a nástrojů. Diskutují rovněž o problémech, které se objevily během sprintu a snaží se najít způsob, jak zlepšit nedokonalé procesy uvnitř rámce Scrumu a předejít těmto problémům (Schwaber a Sutherland, 2020).

Je důležité si pamatovat, že se podle Scrum metody produkt nevyvíjí kompletně najednou, ale po malých funkčních částech, viz obrázek níže.



Obr. 4 – Schéma Scrum
Zdroj: upraveno dle Scrum (2023)

Scrum metoda se velmi často používá v případě, kdy se vyvíjí nový produkt, u kterého je velmi obtížné definovat finální výstup/podobu či se požadavky na produkt mohou v čase měnit v závislosti na byznysu. Proto je pro takový vývoj vhodná implementace po částech, kdy se v průběhu "dolaďuje" zadání.

Kanban

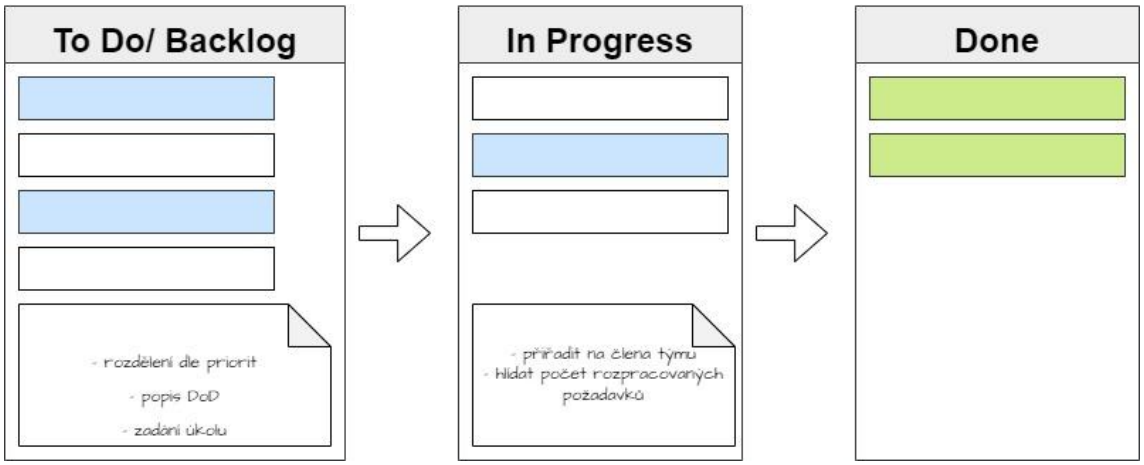
Kanban je agilní metoda založená na vizualizaci správy pracovních postupů. Díky postupnému vývoji produktu metoda Kanban umožňuje získat kvalitní výsledek stejně jako tomu je u Scrumu.

Kanban pomáhá vizualizovat pracovní postupy, omezuje počet rozpracovaných úkolů a rychle přesouvá práci od plánování k dokončení skrze tři základní stavy – Backlog, In progress, Done. Zjednodušené schéma procesu Kanban je znázorněno na obrázku (Obr. 5). Samozřejmě je možné dle potřeby přidávat další stavy, ale i zde mnohdy platí pravidlo, že čím méně, tím lépe. Tato metoda je nejvhodnější pro tým, který zpracovává velké množství požadavků, které se liší velikostí a prioritou dokončení. Velmi používaný je pro maintenance týmy, které se starají o údržbu

softwaru, a kde vzhledem k povaze práce nejde snadno zafixovat práci na pravidelné iterace.

Úkoly v Kanbanu je možné kdykoliv zastavit, pokud k tomu existují důvody, např. změna priorit či odbavení jiných urgentnějších, naléhavějších úkolů či takových, jejichž vyřešení přinese vyšší přidanou hodnotu či ziskovost společnosti. Zjednodušené schéma procesu Kanbanu je znázorněno na Obr. 5.

Oproti Scrumu Kanban nedefinuje žádné role ani meetingy, nicméně v praxi se často lze setkat s tím, že tým má pravidelné Stand-upy (SU) a občas i retrospektivy (viz Artefakty a události). Kanban navíc oproti Scrumu nelpí na pravidelných dodávkách stabilních verzi produktu, ale na eliminaci rozdělané práce.



Obr. 5 – Kanban tabule
Zdroj: upraveno dle Atlassian (2023d)

Používání Kanban tabule je velmi oblíbené i v účetním a finančním sektoru, kde se používá především pro vizualizaci všech úkolů. V této oblasti se pracovníci zaměřují na splnění daného úkolu, a nikoliv většího implementačního celku jako to bývá u vývoje softwaru.

Silné a slabé stránky Agilní metody

Agilní vývoj umožňuje provádět změny v každé fázi projektu, přizpůsobit projekt požadavkům Product Ownera, a co nejrychleji dodat produkt na trh. Dále poskytuje možnost okamžitě zahájit vývoj, jakmile je dohodnut obchodní model, celková strategie a funkcionality, a to aniž by byla sestavena vyčerpávající

dokumentace postihující veškeré specifika výsledného produktu. Vývojový tým a Product Owner mají také možnost probírat aktuální aktivity, problémy či budoucí změny každý den, což je z pohledu obou stran velmi žádoucí a efektivní. Vše výše zmíněné je považováno za silné stránky agilních metod.

Na základě praxe a odborných konzultací s kolegy ve firmě dodavatele autorka mezi další výhody agilních metod řadí následující:

- rychlé a efektivní rozhodování, zapracování změn,
- rychlá (průběžná) identifikace a odstranění chyb,
- minimalizace rizika nenaplnění očekávání a představ klienta,
- transparentnost a průběžná informovanost vlastníka produktu o stavu a výsledcích vývoje,
- vyšší efektivita vývoje díky blízké spolupráci vývojového týmu a klienta,
- použitelnost produktu či jeho části již v průběhu vývoje,
- včasná reakce na vývoj trhu a konkurenci.

Přes všechny své pozitivní vlastnosti nezaručují agilní metody vždy dobré výsledky. Pro docílení vysoké efektivity je zapotřebí umět tyto metody správně používat a zvolit vhodnou metodu pro daný projekt na základě jeho specifik.

Agilní metody mohou mít několik nevýhod, které plynou z flexibility této metody a projevují se zejména, dochází-li k mnoha změnám a velkým zásahům do specifikace požadavků a zadání projektu. Mohou mít dopad, jak na klienta, tak na vývojový tým dodavatele. Mezi takové nevýhody patří následující:

- nemožnost přesného odhadu celkových nákladů na začátku projektu a složité plánování finančních zdrojů,
- riziko nedokončení všech původně požadovaných funkcionalit za daného rozpočtu a potenciální snížení celkové kvality konečného produktu,
- možnost potřeby zásadního navýšení rozpočtu,
- riziko prodloužení harmonogramu projektu,
- demotivace vývojového týmu,
- malá pozornost věnovaná tvorbě dokumentace.

Časové omezení délky každého sprintu může být rovněž příčinou nižší kvality v dodávce jednotlivých funkcionalit, proto je důležité dohlížet na kvalitu výsledných řešení a řídit se Definition of Done (DoD).

6 Výběr a aplikace metody pro vybraný projekt z praxe

Cílem praktické části diplomové práce je demonstrovat proces výběru vhodné metody pro řízení projektu a popsat projektové procesy, které byly součástí vývoje konkrétního produktu. Tento proces je simulován na příkladu projektového požadavku.

V první části je popsán a následně analyzován projektový požadavek, který byl vybrán z praxe autorky. Tento projektový požadavek řešila dodavatelská společnost, ve které autorka pracuje na pozici projektového manažera. Jedná se o středně velkou IT firmu čítající cca 100 zaměstnanců, která působí převážně na českém a slovenském trhu. Jejími klienty jsou zejména společnosti z finančního sektoru a dalších služeb jako logistika, maloobchod atd. Jedním z dlouholetých klientů dodavatele je také zadavatelská společnost, která se specializuje na poskytování operativního leasingu vozidel svým zákazníkům, a fungujícím ve více než 40 zemích po celém světě. Zadavatelská společnost (dále "klient") se velmi snaží o spokojenost svých zákazníků, a proto dbá na uživatelskou přívětivost všech svých IT platforem. Jednou z takových platforem je webová aplikace pro online správu vozového parku a souvisejících aktivit (dále "portál"). Tato platforma svým uživatelům usnadňuje vyhledávání informací o stavu vozu a případných povinnostech vyplývajících z leasingu vozidla a souvisejících služeb. V rámci webové aplikace je uživateli umožněn výběr, konfigurace a objednání nového vozu.

Klientův požadavek na dodavatele vychází z rozhodnutí modernizovat svůj webový portál s využitím nejnovějších technologií. Cílem dodávky je poskytnout konečným zákazníkům tu nejlepší dostupnou úroveň služeb odpovídající aktuálním trendům a zvýšení konkurenceschopnosti podniku klienta. Z důvodu rozsáhlosti portálu a nutnosti jeho celkové modernizace, která by trvala přibližně 1-2 roky při dané kapacitě IT týmu a finančním možnostem klienta, byl zvolen postupný přístup k "přepisu" (aktualizaci) jednotlivých menu záložek. To znamená, že modernizace portálu probíhá postupně, jedna záložka po druhé, dokud nebude celý portál kompletně aktualizován. V této diplomové práci je pozornost věnována modernizaci úvodní stránky aplikace, tzv. nástěnky, do nové technologie.

Nástěnka, nebo také dashboard, je nástroj pro vizualizaci dat, který slouží k rychlému a snadnému získání přehledu o klíčových informacích. Nástěnka obvykle zobrazuje data v podobě grafů, tabulek, ukazatelů nebo dalších vizuálních prvků, které pomáhají konečným uživatelům pochopit informace, v případě tohoto projektu např. o stavu údajů ve smlouvách, pojistných událostech, spotřebě pohonných hmot nebo o zadaných požadavcích na klienta a notifikacích k důležitým událostem (expirace STK, vrácení vozidla atd.).

Nástěnka by měla být jednoduchá a snadno čitelná, aby uživatelé mohli rychle porozumět informacím, které jsou prezentovány. Je důležité pečlivě vybrat klíčové metriky a data, která se na nástěnce zobrazí, aby uživatelé mohli snadno vyhodnotit důležité informace a provést informovaná rozhodnutí. Zadání projektu

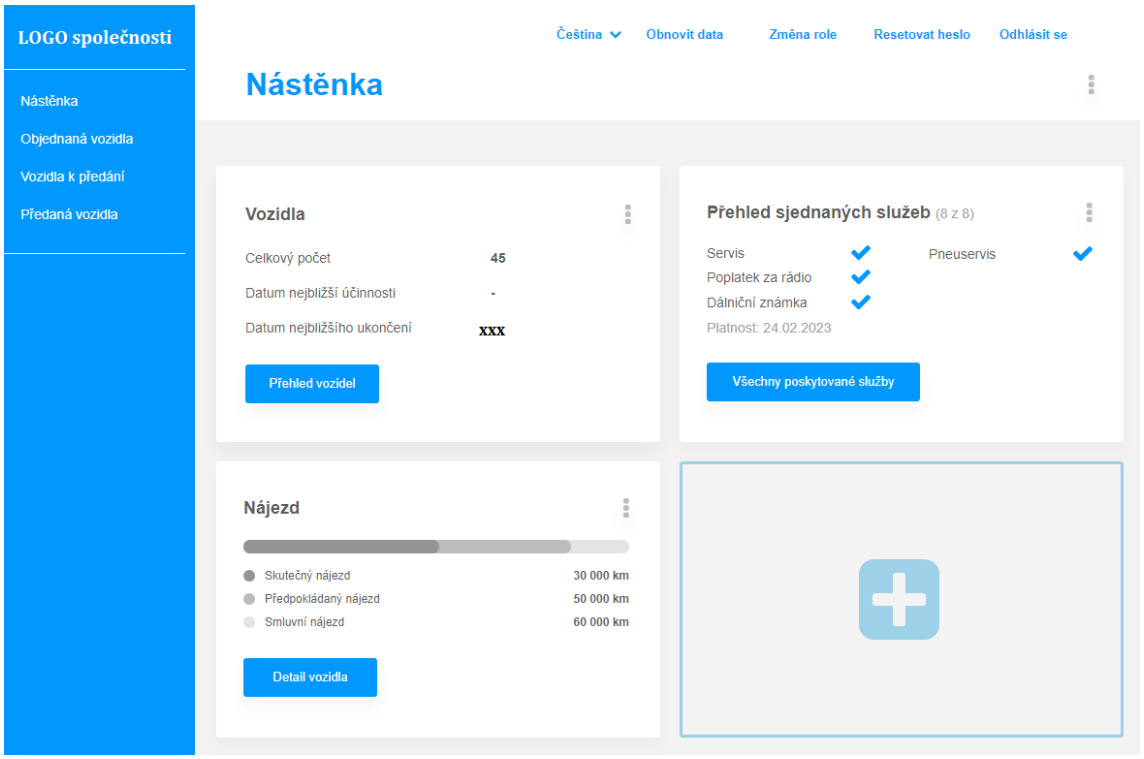
Klient si uvědomuje nízkou úroveň uživatelské přívětivosti aktuální formy nástěnky, a proto bylo dodavateli v rámci projektu modernizace portálu zadáno několik hlavních požadavků shrnutých na následujících řádcích.

Prvním požadavkem je dynamičnost nástěnky, která zajistí uživateli možnost přemísťovat jednotlivé boxy (části obrazovky) v rámci nástěnky a určovat jejich pozici. Tato funkcionalita se nazývá „drag and drop“.

Druhým požadavkem klienta je tzv. uživatelská volba. Díky ní si budou uživatelé moci sami vybrat, které boxy chtějí mít na hlavní stránce zobrazené a které nikoliv, podle toho, jak jsou pro ně jednotlivé boxy a informace v nich obsažené relevantní a důležité.

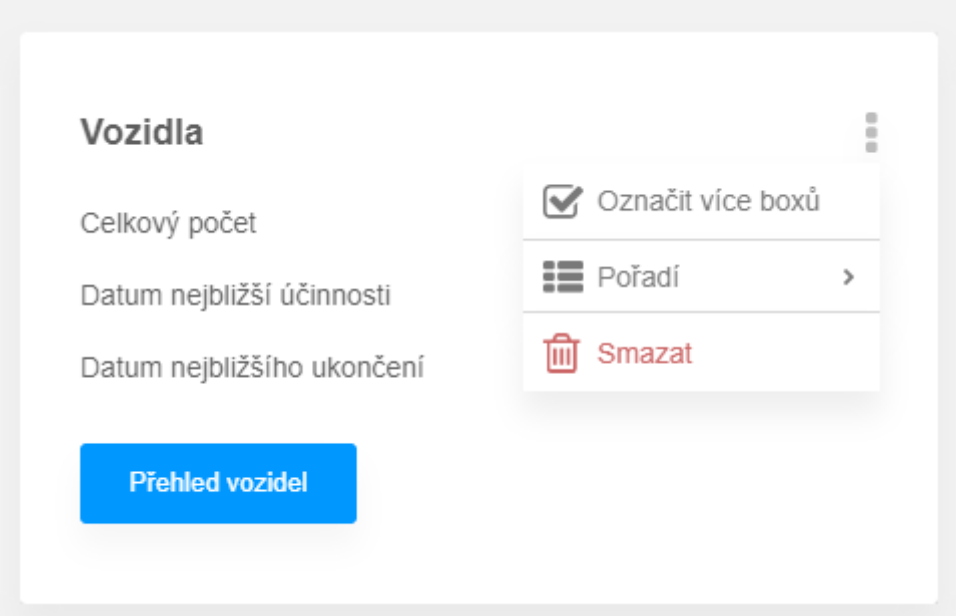
Třetím požadavkem klienta na modernizaci hlavní stránky portálu je sjednocení všech boxů nástěnky do jednoho typologického boxu. Pokud by se do nového řešení měla implementovat původní forma nástěnky umožňující hned několik způsobů zobrazení informací, a tedy vytvoření každého boxu zvlášť, znamenalo by to zvýšení pracnosti vývoje a s tím související zvýšení finanční náročnosti projektu. Rozhodnutí zvolit jeden typologický box je tedy finančně výhodnější a představuje nejjednodušší způsob, jak předejít komplexitě vývoje. Na Obr. 7 lze v každém z boxů nástěnky vidět různé typy reprezentace dat a informací. Pro uživatele nepředstavuje odlišnost jednotlivých prvků nic zásadního, dopad na náročnost a pracnost vývoje je však výrazná. Všechny nově vznikající boxy jsou tedy vytvářeny pouze skrze jeden typologický box, který je znázorněn na Obr. 17. Jeden typologický box znamená,

že se všechny informace v jednotlivých boxech budou zobrazovat stejně čili bez fajfek a posuvníků, ale pouze ve dvou sloupcích jako to je v boxu „Vozidla“ na Obr.6.

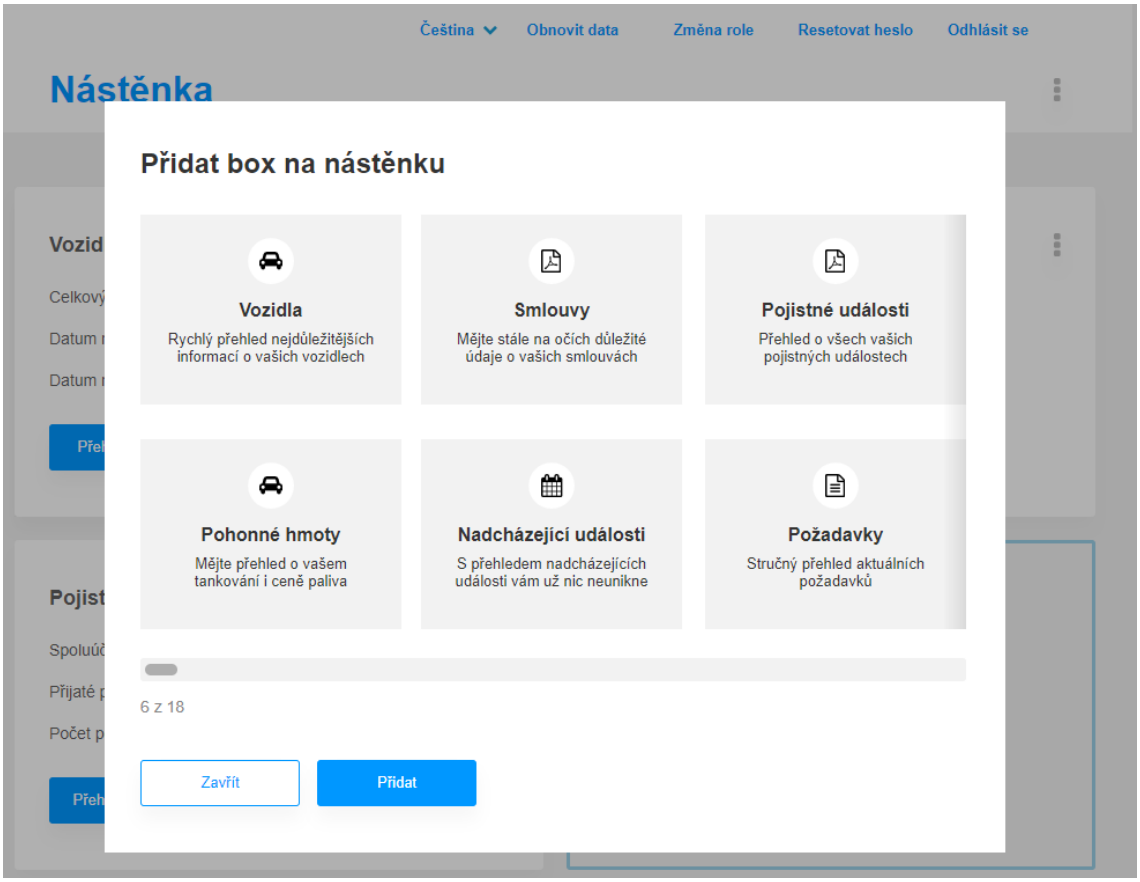


Obr. 6 – Nástěnka starého portálu
Zdroj: snímek obrazovky ze starého portálu

Obrázek výše znázorňuje některé z možností, jak manipulovat s boxy, které jsou uživatelům poskytnuty. V momentě, kdy uživatel rozklikne jednotlivé komponenty, zobrazí se mu možnost výběru další akce. Například po kliknutí na symbol se třemi tečkami se uživateli zobrazí nabídka akcí pro daný box, tj. možnost jeho smazání, změna pořadí boxů nebo označení více boxů pro současnou manipulaci s nimi (viz Obr. 8). Prázdný box s ikonou “plus” umožňuje uživateli přidat další boxy, které si může zobrazit na nástěnce (viz Obr. 9).



Obr. 7 – Nabídka ve starém portálu
Zdroj: snímek obrazovky ze starého portálu



Obr. 8 – Přidání boxů ve starém portálu
Zdroj: snímek obrazovky ze starého portálu

Již v původní verzi portálu je práce s nástěnkou relativně snadná a intuitivní. Kvůli potřebě aktualizace samotné technologie, mimo jiné např. z bezpečnostních důvodů, je však nutné portál celkově modernizovat. Tato modernizace umožňuje další návazné inovace jako například optimalizace designu ovládacích prvků, což přinese uživatelům ještě jednodušší a efektivnější způsob práce s portálem. Konkrétním příkladem je již zmíněná funkcionalita "drag and drop", která uživateli umožní přesouvat jednotlivé boxy mezi sebou (vybráním a posunem pomocí myši na určené místo). Tato funkce nahradí ruční zadávání pořadí boxů na nástěnce.

Při zadání projektu klient nejdříve definoval hlavní tři požadavky pro modernizaci nástěnky portálu, které jsou popsány na začátku kapitoly. Poté je na realizačním týmů dodavatele, aby připravil návrhy jejich naplnění. Mezi hlavní kroky, které v této fázi probíhají, patří:

1. analýza zadání (např. odhalení negativních scénářů),
2. vytvoření UI/UX návrhu,
3. odhad pracnosti vývoje jednotlivých komponent aplikace,
4. rozdělení práce do jednotlivých tasků v nástroji Jira,

a nakonec zahájení samotného vývoje.

Pro potřeby diplomové práce zaměřené na výběr metod používaných v dodavatelské firmě a na řízení projektů byla vybrána první část projektu modernizace portálu – nástěnka, která již byla úspěšně dokončena. Tento výběr slouží k demonstraci průběhu projektových procesů a použitých nástrojů (Jira, Confluence). Při výběru vhodné metody a následně i metody je však důležité myslet na zvolené postupy nejen pro vývoj této části, ale i pro celý projekt modernizace portálu.

Je nutné pečlivě zvažovat, jaké metody budou nejvhodnější pro dosažení cílů projektu, a to s ohledem na jeho specifika a požadavky. Zvolené metody by měly být vždy přizpůsobeny konkrétnímu projektu a integrovány do celkového projektového procesu. Koordinace jednotlivých aktivit a procesů v projektu je klíčová pro úspěšné dosažení cílů a splnění požadavků zainteresovaných stran.

6.1 Porovnání dvou metod

Pro zvolení vhodné metody pro konkrétní projekt od klienta dodavatelské firmy je potřeba nejdříve porovnat dvě vybrané metody a určit, která bude lépe odpovídat charakteristikám projektu. Pro tyto účely autorka diplomové práce na základě rozdílností metod vydefinovaných testerem Thomasem Hamiltonem (Hamilton, 2023) a projektovou manažerkou Alisou Kornevovou (Korneva, 2020) sestavila strukturovanou tabulku doplněnou o vlastní zkušenosti a zkušenosti kolegů z praxe. Tato tabulka autorce jako začínající/juniorní projektové manažerce pomohla identifikovat klíčová specifika projektu a získat lepší přehled o tom, jak vybrat vhodnou metodu pro jeho úspěšné řízení.

Tabulka 1 – Rozdíl mezi Agile a Waterfall metodou

Charakteristiky projektu	*Agile*	*Waterfall*
Vývojový cyklus	Životní cyklus vývoje projektu rozdělen do sprintů.	Proces vývoje je rozdělen do omezených po sobě jdoucích fází.
Flexibilita	Požadavky se v průběhu mohou měnit.	Po zahájení projektu není možné měnit požadavky, pouze přes změnové požadavky.
Velikost týmu	Menší samoorganizační tým 5-9 členů.	Týmová koordinace a synchronizace je velmi omezená – pouze při přechodu z jedné fáze do druhé.
Komunikace v týmu	Každodenní Stand-up a přímá komunikace.	Je možné komunikovat pomocí dokumentace, například komentářů v dokumentu. Nicméně, pro lepší koordinaci projektu a předcházení nedorozumění, je vhodné nastavit pravidelné schůzky s projekčním týmem. Tyto schůzky by měly být efektivní a plánovány tak, aby řešily aktuální otázky a problémy. Měly by být průběžně monitorovány a upravovány podle potřeb projektu a vývoje jeho fází.
Kontrola výsledku	Po každém sprintu.	Ve fázi testování. V praxi je možné provádět kontrolu i během fáze návrhu softwaru ve formě dry run testování, kdy si tester při procházení zadání vytváří testovací scénáře, a tím představuje tzv. “druhé oči”

		analytikovi. Tento typ testování probíhá při psaní analýzy ve fázi vývoje.
Definování projektu	Detaily se mohou upřesnit během vývoje.	Podrobný popis je potřeba mít už při zahájení projektu, kdy se sepíší požadavky a definuje projekt. Po tomto kroku se projekt pevně řídí podle termínů a rozsahu projektu.
Rozpočet	Peníze a čas nemají většinou pevné limity.	Pevná cena, která je dohodnutá předem.
Dokumentace	Stručná.	Rozsáhlá -> při zahájení projektu klient definuje, jak rozsáhlá má dokumentace být.
Rozsah projektu	Mění se v průběhu.	Definuje se na začátku projektu a dále se nemění, pouze v případě definování změnových požadavků.
Termín dodání produktu	Po dokončení sprintu vzniká fungující produkt -> termín závisí na požadavcích klienta.	Po analýze je možné termín dodání přesně odhadnout.
Dostupnost pro zákazníka	Zákazník může předkládat připomínky během vývoje.	Přítomnost zákazníka je nezbytná pouze na začátku a konci projektu.
Projektový manažer	Členové v týmu jsou nahraditelní, a díky tomu pracují rychleji. Přítomnost PM není příliš potřeba, protože projekt je řízen celým týmem. Dle praxe je ale tento přístup jako ideál vystřižený z teorie. Tým může být takto definován v případě, že se skládá ze seniorních vývojářů.	Proces je vždy přímočarý, PM hraje zásadní roli v každé fázi projektu.
Release produktu	Nové funkcionality vytvořené během sprintů se periodicky dostávají mezi uživatele.	Release po dokončení celého projektu v co nejlepší kvalitě. V praxi není běžné, že by byl hned první release úspěšný, na což je potřeba brát ohled a být již dopředu s klientem domluven na případných změnách postupu.

Zdroj: vlastní zpracování

Dva výše zmíněné rozdílné pohledy na řízení projektu mohou posloužit ke snazšímu rozhodnutí, kterou z metod řízení projektu využít pro startující projekt. Je nutné podotknout, že k úspěšnému řízení vývoje softwaru ne vždy stačí zvolit

pouze jednu metodu a řídit se jí v průběhu celého procesu. V případě rozsáhlého projektu, který je vyvíjen po delší časový úsek, je možnost rychle měnit aktuálně používanou metodu řízení v závislosti na měnících se okolnostech projektu celkem důležitá.

Změna vybrané metody řízení projektu na jeho začátku za jinou může lépe pomoci vyřešit problém, se kterým se tým potýká. Proto by měl být PM schopen odhadnout správný okamžik v průběhu trvání projektu, kdy je vhodné využít jinou metodu řízení. Detailnější popis charakteristik projektu, sloužících pro výběr vhodné metody řízení projektu, je představen v samostatné kapitole níže.

V praxi jsou agilní metody řízení projektu vhodné do prostředí vývoje webů a modulárních aplikací, kdy je možné aplikace uvádět do provozu postupně po jednotlivých částech. Waterfall je vhodné využít v případě komplexních projektů, kde může změna představovat velký zásah do vývojového řešení. Jako příklad lze uvést datové sklady a jejich stavbu.

6.2 Výběr vhodné metody

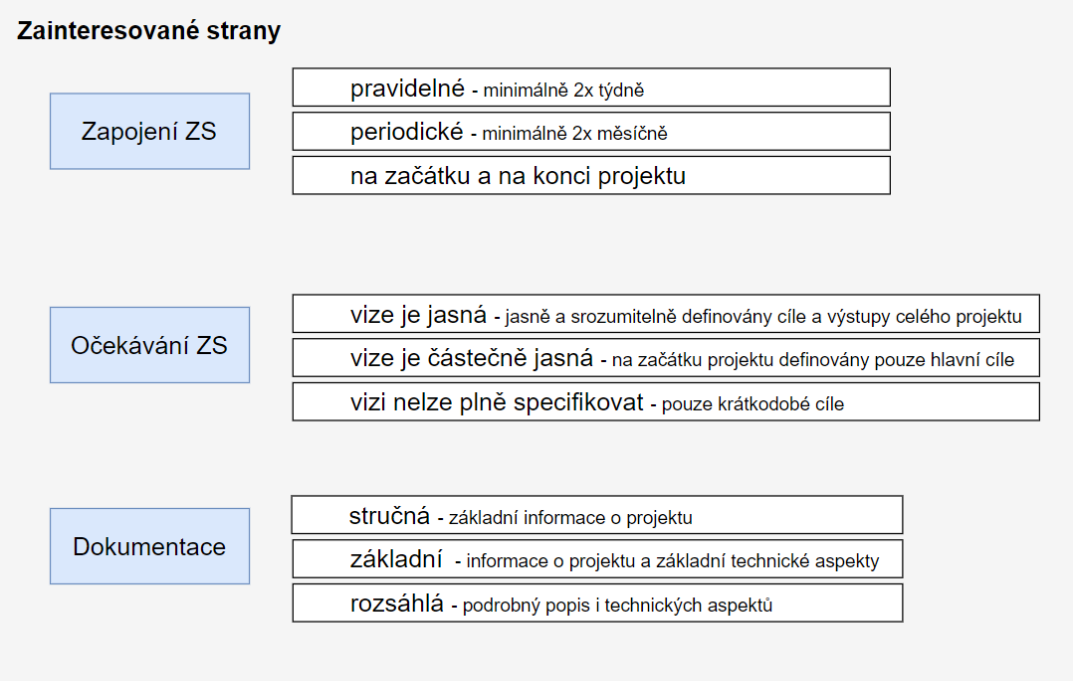
Aby byl IT projekt úspěšný, je klíčové zvolit vhodnou metodu jeho řízení. Při výběru správného rámce, který projektu nejvíce vyhovuje, je nutné postupovat pečlivě. Autorka na základně literatury (Torosyan a Tulkina, 2020) vytvořila tabulku (viz níže), která pomáhá projektovým manažerům vybrat vhodnou metodu řízení projektu. Je nutné vzít na vědomí, že tato tabulka a proces výběru vhodné metody v této diplomové práci probíhá v kontextu praxe autorky ve firmě dodavatele. Jedná se o středně velkou IT firmu čítající cca 100 zaměstnanců, která působí převážně na českém a slovenském trhu. Jejími klienty jsou zejména společnosti z finančního sektoru a dalších služeb jako logistika, maloobchod atd.

Tabulka zahrnuje základní kritéria výběru a jejich indikátory. Hlavními kritérii jsou IT projekt, zainteresované strany, produkt, tým a organizace. Indikátory jsou další faktory, které blíže specifikují charakteristiku projektu a pomáhají tak rozhodnout jakou metodu zvolit.

Zainteresované strany

Při výběru metody je důležité brát v potaz požadavky klienta. Ve skutečnosti je to nejdůležitější atribut pro spokojenost zákazníka s produktem. V případě, kdy dodavatel nemá informace o představě zákazníka o produktu a neví tedy, jakým směrem se ve vývoji vydat, je obtížné zákazníkova očekávání naplnit. Dodavatel může vystupovat v roli konzultanta, pokud si zákazník není jistý tím, jaký produkt přesně potřebuje. Dodavatel může pomoci zákazníkovi zjistit, jakého výsledku je možné dosáhnout a jakou přidanou hodnotu mu produkt přinese. Příkladem může být zákazníkem nespecifikovaná modernizace softwaru, kdy dodavatel, jako konzultant, prozkoumá trh a navrhne řešení, které by bylo pro zákazníka užitečné a odpovídalo jeho potřebám.

Je důležité si stanovit, zda má zákazník od začátku přesnou představu o tom, co potřebuje, nebo zda bude v průběhu vývoje povoleno zadání upřesňovat. Prvnímu případu, kdy má zákazník přesnou představu o konečném výsledku, tedy produktu, odpovídají strukturované metody, např. Waterfall. U druhého případu, kdy zákazník neví, co přesně požaduje a počítá se s průběžnými změnami rozsahu projektu, je vhodné vybrat flexibilnější metodu řízení. Na začátku spolupráce je nutné stanovit i rozsah dokumentace pro klienta. Otázkou je, zda klient požaduje rozsáhlejší dokumentaci, jako je tomu v metodě Waterfall, nebo stručnou jako v Agile. Stručná dokumentace IT projektu by měla obsahovat základní informace o projektu, jako jsou jméno projektu, popis účelu a cílů projektu, seznam funkcí a požadavků na projekt. Také by měla obsahovat popis projektového týmu a rolí jeho členů. Rozsáhlá dokumentace IT projektu by měla obsahovat podrobný popis technických aspektů projektu, jako jsou architektura systému, specifikace rozhraní, podrobný popis jednotlivých funkcí a komponent projektu, popis datových struktur a návrh databáze. Dále by měla obsahovat plány a postupy pro testování, řízení rizik, řízení verzí a správu konfigurace. Základní dokumentace by měla být vyváženým kompromisem mezi stručnou a rozsáhlou dokumentací, která poskytuje důležité informace o projektu včetně základních technických aspektů.



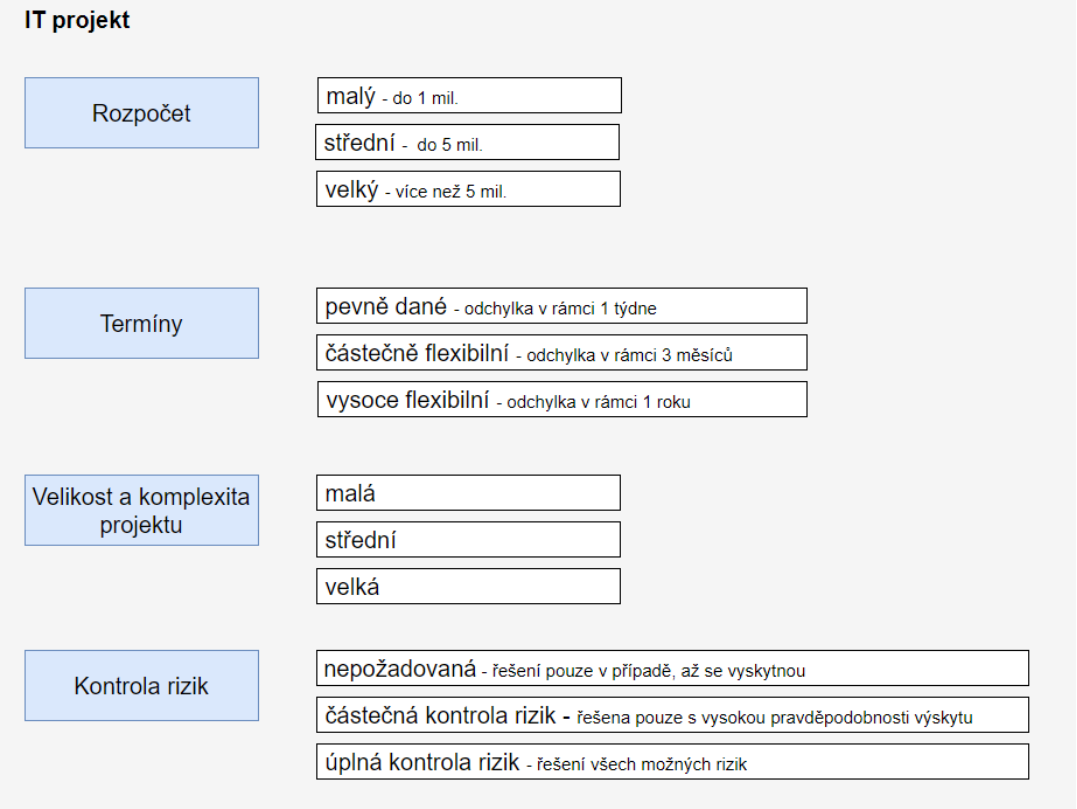
Obr. 9 – Zainteresované strany
Zdroj: vlastní zpracování

Kromě očekávání klienta je třeba dbát na jeho požadavek aktivně se zapojovat do spolupráce s dodavatelem, a na základě toho zvolit vhodnou metodu řízení projektu. V případě, kdy klient nevyžaduje úzkou spolupráci s dodavatelem, jeho zájem je směřován pouze k uspokojujícímu konečnému produktu a ne k procesu vývoje, bude vyhovující metodou Waterfall.

IT projekt

V každém projektu jsou stanoveny termíny rozdělující projekt na fáze a finanční rozpočet určující složení týmu. Dva zmíněné indikátory mají tedy přímý vliv na výběr metody řízení projektu. Termíny mohou být buď flexibilní, nebo pevně dané.

Na výběr metody řízení projektu má vliv i náročnost a komplexnost požadavku klienta. Pro požadavek vyžadující velký objem práce, a tedy i rozsáhlý tým, je vhodný spíše Waterfall model řízení nebo hybrid čili kombinace Waterfall a Agile modelů.

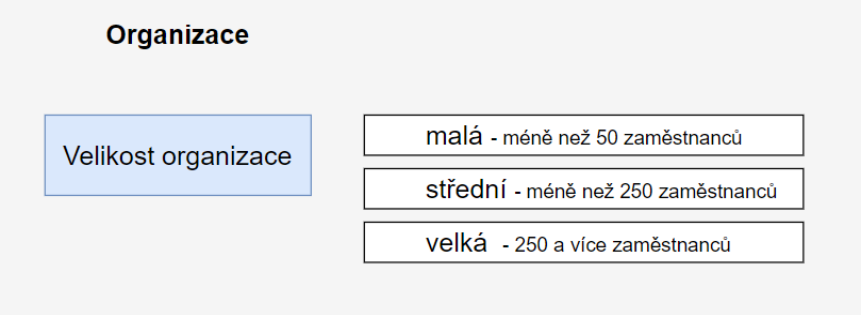


Obr. 10 – IT projekt
Zdroj: vlastní zpracování

Při výběru metody řízení IT projektu je nutné vzít v úvahu možná rizika. Rizika se zpravidla odvíjí od složitosti IT projektu, finančního rozpočtu a očekávaného výsledku.

Organizace

Při výběru vhodné metody řízení IT projektu je důležité zohlednit nejen samotný projekt, ale také velikost a životní fázi organizace. Velikost organizace se obvykle rozděluje na malé, střední a velké firmy. Každá z těchto kategorií má své specifické požadavky na řízení projektů a vhodnou metodu. Například malé firmy mohou preferovat agilní metody, které jsou flexibilní a umožňují rychlou reakci na změny, zatímco velké firmy často potřebují více formální a strukturované přístupy, jako je například Waterfall. Životní fáze organizace může také ovlivnit výběr metody, například start-upy často potřebují rychlé a agilní řešení, zatímco zavedené firmy s pevnou strukturou preferují metody, které zahrnují více plánování a koordinaci.

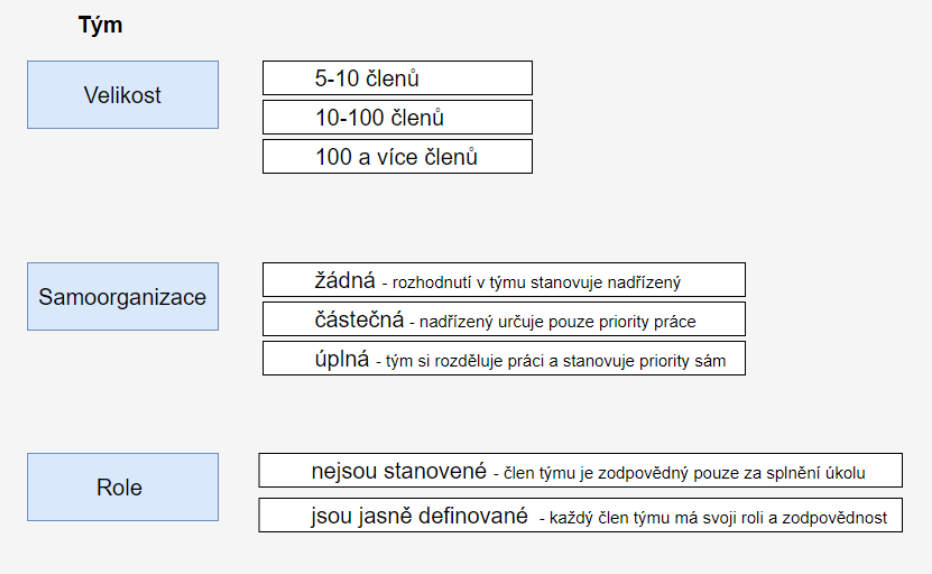


Obr. 11 – Organizace
Zdroj: vlastní zpracování

Tým

Metoda zvolená pro řízení projektu je plánem projektu, který je představován týmu. Jedná se o plán toho, kdy a co bude vyvíjeno. Při startu vývoje je třeba tým se zvolenou metodou seznámit. Každý jednotlivec musí být informován o požadavcích, které jsou na něho kladeny. Bez znalosti obsahu vybrané metody bude komplikované dosáhnout úspěšného výsledku.

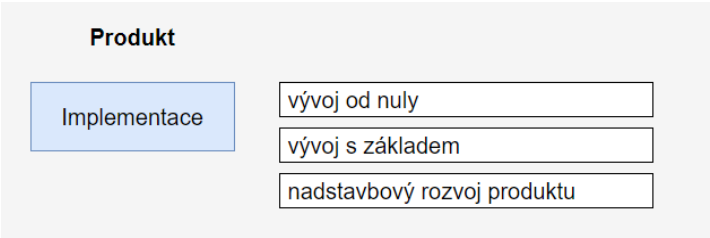
Nejdříve je nutné naučit tým s daným modelem pracovat, což může ovlivnit termíny dokončení projektu. Při výběru metody je nutné analyzovat sestavu týmu, vzít v potaz počet členů a jejich silné i slabé stránky. Pro tým, který je dobře sladěný a snadno spolupracuje, se hodí metoda Agile. Strukturovanému a motivovanému týmu vyhovuje metoda Scrum. V případě omezeného počtu lidských zdrojů a nevymezených rolí se využívá metoda Kanban.



Obr. 12 – Tým
Zdroj: vlastní zpracování

Produkt

Posledním zmíněným kritériem výběru metody řízení projektu je produkt, u kterého se rozhoduje na základě typu vývoje. Vývoj od nuly čili od přípravy prostředí po release znamená, že je vše prováděno v rámci jednoho projektu. Vývoj se základem – jedná se o pokračování ve vývoji na již existujícím minimálním základu softwaru (spolupráce více dodavatelů). Nadstavbový rozvoj produktu – produkt, který již existuje a je v provozu a který klient požaduje vylepšit, rozšířit nebo modernizovat.



Obr. 13 – Produkt
Zdroj: vlastní zpracování

Níže sestavená tabulka má za cíl pomoci PM při výběru vhodné metody řízení projektu. Průnik jednotlivých charakteristik udává jasnou představu o tom, která metoda řízení projektu je pro daný projekt vhodná.

Tabulka 2 – Výběr vhodné metody

Kritéria	Indikátory	Waterfall	Agile	
			Scrum	Kanban
Zainteresované strany	Zapojení ZS	• Na začátku a na konci • Periodické (klientova volba)	• Pravidelně ✓	• Pravidelně ✓
	Očekávání ZS	• Vize je jasná	• Vizi nelze plně specifikovat ✓ (pouze v rámci sprintu = cíl sprintu)	• Vize je částečně jasná
	Dokumentace	• Rozsáhlá • Základní (klientova volba) ✓	• Stručná • Základní (klientova volba) ✓	• Stručná • Základní (klientova volba) ✓
IT projekt	Rozpočet	• Velký • Střední ✓ • Malý	• Velký • Střední ✓ • Malý	• Střední ✓ • Malý
	Termíny	• Pevně dané	• Částečně flexibilní ✓	• Vysoce flexibilní
	Velikost a komplexita	• Malá • Střední ✓ • Složitá	• Malá • Střední ✓	• Malá
	Kontrola rizik	• Částečná ✓ • Úplná	• Částečná ✓ • Úplná	• Nepožadovaná • Částečná ✓ • Úplná
Organizace	Velikost organizace	• Střední ✓ • Velká	• Malá • Střední ✓	• Malá • Střední ✓
Tým	Velikost	• 10-100 členů • od 100 členů	• 5-10 členů ✓	• 5-10 členů ✓
	Samoorganizace	• Žádná	• Úplná ✓ • Částečná	• Úplná ✓
	Role	• Ano ✓	• Ano ✓	• Ne
Produkt	Implementace	• Vývoj od nuly • Vývoj se základem • Nadstavbový rozvoj ✓	• Vývoj od nuly • Vývoj se základem • Nadstavbový rozvoj ✓	• Nadstavbový rozvoj ✓
		Σ7	Σ12	Σ8

Zdroj: vlastní zpracování

Výběr odpovídající hodnoty k ukázkovému projektu je znázorněn v tabulce pomocí znaku „✓“. Suma těchto hodnot vybraných k jednotlivým metodám je uvedena na konci tabulky ve formátu „∑x“.

6.3 Shrnutí okolností a výsledek výběru metody pro ukázkový projekt

Jak již bylo v této práci uvedeno, demonstrace výběru metody probíhá zejména na nových požadavcích na část projektu “Nástěnka”, avšak pro účely výběru vhodné metody je potřeba brát v úvahu celkovou modernizaci portálu, která by měla trvat odhadem 1 až 2 roky. Klient, pro kterého se projekt realizuje, je středně velká firma, která má zájem o pravidelné zapojení do vývojového procesu a klade si za cíl mít modernizovaný a uživatelsky přívětivý portál.

Vize projektu zahrnuje nejen vytvoření nového grafického návrhu portálu, ale také potřebu technické modernizace celého portálu. Je tedy třeba brát v úvahu, že projekt bude trvat delší dobu a bude potřebovat pečlivé plánování a koordinaci.

Podrobná dokumentace není pro klienta prioritou, ale zato klade důraz především na pravidelné poskytování nejdůležitějších informací, které jsou nezbytné pro dokončení projektu. Z tohoto důvodu se projektová dokumentace, která je ukládána na Confluence, zaměřuje především na přehledné a srozumitelné vysvětlení klíčových aspektů projektu, které klientovi umožní porozumět tomu, co se děje, a jak projekt postupuje, na místo snahy o komplexní popis každého kroku projektu.

Ke kritériu IT projekt je důležité zmínit, že projekt má omezené finanční a časové limity, přestože tyto limity nemusí být striktně dodrženy. S klientem je dohodnutá možná odchylka +- 15 % od prvotní dohody při začátku realizace. Komplexita projektu je poměrně vysoká, protože je portál integrován s několika dalšími systémy klienta a třetích stran, a to vyžaduje průběžné analýzy dopadu na integraci s těmito systémy. Zanedbání tohoto aspektu může vést k rizikům dodržení termínu a rozpočtu projektu.

Tým, který na projektu pracuje, se skládá z šesti členů – tří vývojářů, jednoho Scrum Mastera, jednoho Product Ownera a jednoho testera, který je zodpovědný

za kvalitu dodávky projektu. Tento tým spolupracuje již tři roky, je velmi autonomní, organizovaný a schopen efektivně řešit dílčí vývojové problémy.

Na základě analýzy jednotlivých kritérií a indikátorů (výběr znazorněn ✓) z tabulky vychází, že pro dosažení maximální efektivity řízení projektu nejlépe vyhovuje metoda Scrum.

6.4 Průběh projektu

Tato kapitola se zaměřuje na popis průběhu ukázkového projektu, který byl realizován za použití agilních metod. Jsou zde popsány jednotlivé kroky, které byly provedeny od zadání projektu až po jeho dokončení. Nejdříve je pozornost věnována sestavení Scrum týmu, poté definici backlogu, rozdělení a odhadování pracnosti úkolů ve sprintu. Následně je popsán průběh agilních ceremonií, jako jsou Stand-upy, plánování, grooming a retrospektiva. Nakonec je shrnut výsledek projektu včetně doby trvání a výsledné celkové pracnosti.

Vytvoření Scrum týmu

Samotný proces vytváření Scrum týmu začal výběrem potřebných členů týmu na základě objednané kapacity týmu klientem. Ta je objednávána vždy na tři měsíce předem podle toho, jaké projekty jsou aktuálně rozpracovány. Technická náročnost vývoje Nástěnky byla rovněž brána v potaz při výběru týmu, aby byla zajištěna odbornost a potřebné zkušenosti pro úspěšné dokončení projektu.

Na základě dlouhodobé spolupráce mezi daným klientem a dodavatelskou firmou bylo rozhodnuto o složení týmu, jehož základ byl identický jako u předchozích společných projektů. Vybraní členové týmu totiž klienta již dobře znají a vědí, jaký způsob spolupráce a komunikace je pro něj nejvhodnější.

Velmi důležitou roli při tomto rozhodnutí hrál fakt, že klient si během předchozích úspěšných projektů vybudoval se členy týmu vztah založený na vzájemné důvěře.

Zejména na začátku nové spolupráce (či spolupráce s novým realizačním týmem) mohou mít klienti pochyby a nemusí svým dodavatelům softwaru plně důvěřovat, například pokud jde o množství vykázaného času stráveného prací. Vybudovaná důvěra tedy hraje roli nejen v tomto ohledu, ale i při samotných

odhadech pracnosti vývoje, nebo při přijímání doporučení ohledně některých technických řešení. Takovýto přístup k výběru členů týmu lze považovat za vhodný.

Dalším krokem byla jasná definice rolí a odpovědností jednotlivých členů týmu. Za řízení projektového procesu a zajištění dodržování principů a praktik Scrumu ostatními členy týmu byl zodpovědný Scrum Master. Role Product Ownera byla zodpovědná za definování cílů projektu, vytvoření Product Backlogu a určení priorit jednotlivých úkolů. Poslední část týmu byla tvořena členy vývojového týmu, který se skládal z backend a frontend vývojářů, testera a rolí jako je DevOps specialista a UX/UI designer.

Z důvodu modernizace aplikace a její architektury – tedy převedení do nové technologie a oddělení backendové a frontendové části aplikace, bylo nutné do původního vývojového týmu zapojit také frontendového JavaScript vývojáře. Aby mohlo dojít také k vizuální proměně Nástěnky portálu, byl do projektu nově zapojen také UX/UI specialista.

Výběr Product Ownera

Scrum Master a Product Owner jsou klíčové role v metodologii Scrum, které mají významný vliv na úspěšnost projektu. Tyto role byly v projektu Nástěnka nezbytné pro správnou implementaci a provozování scrum procesů a principů, které jsou základem této agilní metody. Bez těchto dvou rolí by tým nebyl schopen plně využít potenciálu této metody a dosáhnout stanovených cílů.

Product Owner byl zvolen na straně klienta, protože měl hluboké znalosti potřeb ze strany klienta a také specifickou vizi produktu, který se vyvíjí. To znamená, že byl schopen lépe pochopit potřeby koncového uživatele a byl schopen reprezentovat tyto potřeby tak, aby je pochopil tým vývojářů. Pokud by byl Product Owner zvolen na straně dodavatele, mohl by být ovlivněn interními prioritami a mohl by mít tendenci repriotizovat požadavky klienta tak, aby byly lépe přizpůsobeny jeho vlastním zájmům. Navíc by musel provést detailní analýzu fungování klientova byznysu a celé workflow navnímat, což je velmi nerentabilní z pohledu klienta i dodavatelské IT společnosti. Zapojení Product Ownera na straně klienta tak umožnilo lépe rozvrhovat IT požadavky na projekt a určovat jednotlivé

priority dílčích úkolů, což vedlo ke zvýšení efektivity spolupráce a lepším výsledkům v rámci projektu.

Během projektu Nástěnka se od Product Ownera vyžadovalo, aby měl důkladné znalosti o produktu a o potřebách zákazníka, a také aby byl schopen definovat a prioritizovat požadavky na produkt. Zároveň měl být schopen pracovat se zainteresovanými stranami a přijímat rozhodnutí ohledně toho, co mělo být v rámci projektu vyvinuto a kdy. Během plánování sprintu spolupracoval s týmem na určení cílů pro daný sprint a vybíral položky z Product Backlogu, které měly být v nadcházejícím sprintu dokončeny.

Product Owner měl také za úkol se ujišťovat, že tým chápe, co se od něj očekává, a že má přístup ke všem potřebným informacím. Dále poskytoval zpětnou vazbu od uživatelů produktu, což umožnilo realizačnímu týmu přizpůsobit svou práci tak, aby co nejlépe naplnila potřeby a přání zadavatele. V rámci role Product Ownera bylo potřeba, aby byl flexibilní a schopný navnímávat nové požadavky ze strany koncových uživatelů produktu. Tato schopnost byla zásadní pro úspěšné dokončení projektu v rámci Scrum metody, která se vyznačuje velkou flexibilitou.

Výběr Scrum Mastera

Přestože bývá role Scrum Mastera většinou zastoupena buď ze strany IT dodavatele nebo klienta, měla by být nezávislá. To znamená, že tento člen týmu by neměl být příliš “blízko” k jedné straně, aby mohl být objektivní a spravedlivý při řízení projektu.

V ukázkovém projektu analyzovaném v této diplomové práci se role Scrum Mastera ve většině kompetencí překrývá s rolí PM a na projektu Nástěnky byl alokován ze strany dodavatele, a to z několika důvodů. Prvním důvodem byla nižší pravděpodobnost, že bude mít emocionální vztah k produktu nebo klientovi. Druhým důvodem byla větší důvěra klienta, jelikož dodavatelská firma dodržovala procesy a postupy a neprosazovala své vlastní zájmy. A třetí důvodem bylo to, že dodavatelská firma měla větší zkušenosti s agilní metodou a věděla, jak je takový projekt potřeba řídit.

Scrum Master byl v podstatě koordinátorem týmu, který sloužil jako spojka mezi Product Ownerem a týmem. Jeho úkolem bylo zajistit, aby tým dodržoval

pravidla Scrumu, a byl během vývoje produktu efektivní. Scrum Master pracoval na odstraňování překážek, které bránily týmu při plnění cílů projektu, nebo komunikaci mezi členy týmu a pomáhal tak týmu udržovat tempo. Dále sledoval pokrok projektu a průběžně informoval Product Ownera a vývojový tým o stavu projektu tak, aby měli přehled o aktuálním plnění plánu. Product Owner tak stanovoval, jaké úkoly se budou řešit během sprintu, a Scrum Master plánoval a stanovoval termíny dokončení těchto úkolů.

Definování Product Backlogu

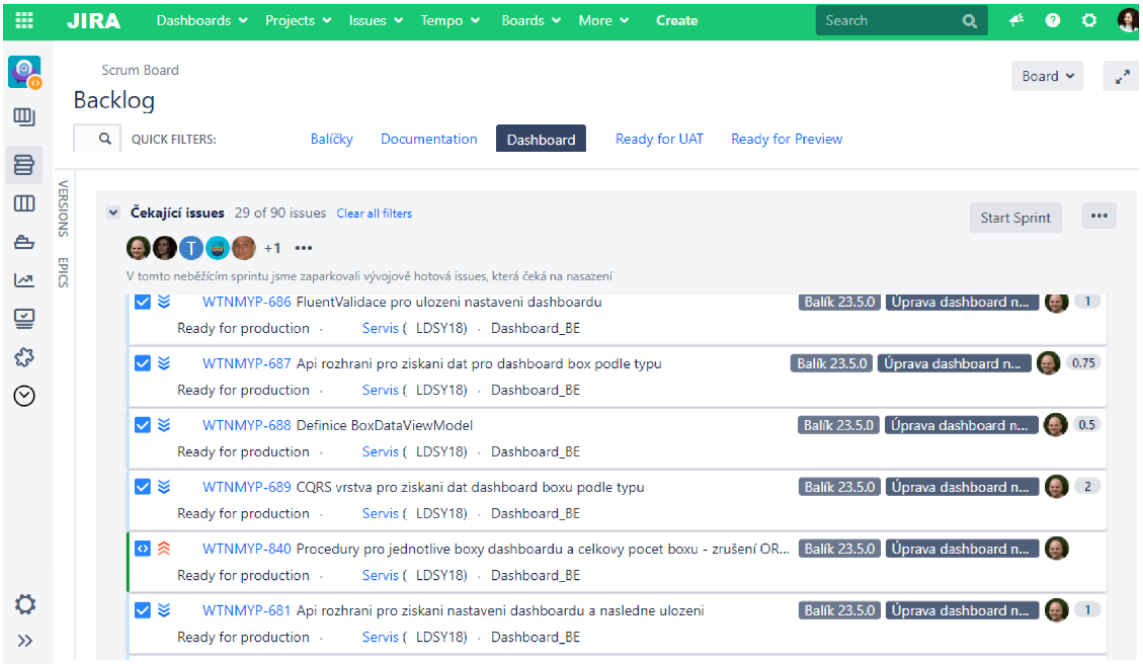
Během projektu byl vytvořen seznam požadavků na vývoj produktu (backlog). Tyto požadavky měly být realizovány během projektu, a v průběhu vývoje byly prioritizovány dle jejich významu pro splnění cílů projektu. Backlog byl tedy dynamický a v čase se postupně měnil, doplňoval, aktualizoval, a to na základě nových informací, priorit či zpětné vazby od Product Ownera. Tým a Product Owner spolu pravidelně komunikovali ohledně požadavků a úkolů v Product Backlogu, aby se navzájem ujistili, že postupují v souladu s cíli projektu, a že se vyhýbají potenciálním problémům.

Product Backlog obsahoval všechny úkoly, které musel tým dokončit, aby vytvořil plně funkční produkt. Každý úkol v backlogu měl svůj popis, odhadovanou dobu implementace, prioritu a zodpovědného člena týmu za jeho realizaci (viz Obr. 14). Popis úkolu musel být dostatečně detailní, aby členové týmu mohli úkol plně pochopit a dokázali ho dokončit.

Product Backlog byl vytvořen na základě počátečního zadání od Product Ownera, který specifikoval, co je potřeba splnit v rámci projektu. Tyto požadavky byly následně předány UI/UX designerovi, který navrhl nový vzhled a funkce nástěnky. Následně se sešel vývojový tým a provedl základní předimplementační analýzu nástěnky. Tato analýza sloužila jako výchozí bod pro posouzení vývojových prací. V případě, že by realizační tým během analýzy narazil na nějaké chybějící oblasti v logice fungování aplikace, se kterými UI/UX návrh nepočítal, bylo by potřeba doplnit zadání a následně upravit také grafický návrh.

Když se tým ujistil, že je předimplementační návrh řešení kompletní, začal se zabývat sepsáním funkcí, které mají tvořit základní stavební bloky produktu.

Tyto funkce byly sepsány v podobě stručných popisů, které se nazývají User Stories. Jedná se o popis požadavků z pohledu uživatele s určitou rolí v daném systému, který má požadovanou operaci provádět. Tyto User Stories byly napsány tak, aby byly srozumitelné pro všechny členy týmu včetně vývojářů, testera, Product Ownera a Scrum Mastera. Při tvorbě takových popisů se kladl důraz na to, aby byly krátké, konkrétní, a měly jasně definovaný cíl.



Obr. 14 – Backlog v Jira
Zdroj: snímek obrazovky z nástroje Jira

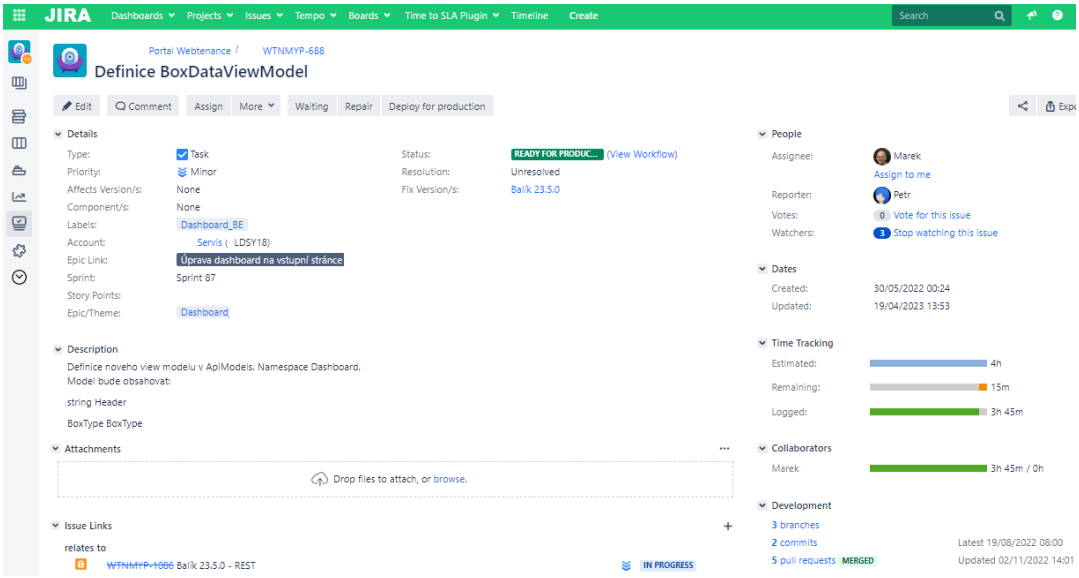
Grooming

Pravidelná grooming schůzka probíhala jednou za 14 dní po dobu 2 hodin. Setkával se na ní tým s Product Ownerem, a společně pracovali na aktualizaci, zpřesňování a detailizaci úkolů v Product Backlogu. Cílem této schůzky bylo zajistit, že Product Backlog obsahuje dostatečně podrobné a přesně definované úkoly, které se mohou naplánovat do sprintu. U každého úkolu měl být doplněn popis, cíl, priorita a odhady časové náročnosti na jejich dokončení.

Kromě pravidelného groomingu, který probíhal každých 14 dní, se občas vyskytla potřeba svolat i neplánovaný (ad hoc) grooming na dovyjasnění některých procesů vývoje a zajištění odstranění překážek, které se při vývoji vyskytly.

Běžně každá grooming schůzka probíhala dle následujících kroků:

1. Tým společně s Product Ownerem provedli kontrolu aktuálního stavu Product Backlogu, tedy zhodnocení toho, jaká práce již byla provedena, a jakou zbývá dokončit v následujícím sprintu.
2. Tým s Product Ownerem společně prošli jednotlivé úkoly v Product Backlogu a diskutovali o tom, které funkcionality jsou na řadě k vývoji. Společně vybrali jednotlivé User Stories s nejvyšší prioritou, které se následně rozdělily na dílčí úkoly neboli Tasky. U těch byly poté dospecifikovány další potřebné detaily.
3. Tasky s nejvyšší prioritou se posunuly na začátek Product Backlogu, a tým tak mohl přistoupit k určení odhadu pracnosti vývoje. Dle Scrum metody by se měl používat Planning poker, kdy každý člen týmu dostane sadu karet, na kterých jsou čísla reprezentující pracnost vývoje, a to buď v hodinách, dnech nebo Story Pointech. V praxi byla zvolena jednotka hodiny, jelikož klient neměl zájem určovat komplexnost jednotlivých úkolů a vystačil si pouze s časovým odhadem toho, jak dlouho zabere vývoj jednotlivých úkolů. Poté, co byl úkol popsán a bylo jasné, co je potřeba pro jeho dokončení, přistoupilo se k hlasování, kdy každý člen týmu současně ukázal kartu s odpovídajícím odhadem časové náročnosti. Následně, co byly karty vyloženy a odhady odhaleny, členové týmu začali diskutovat o rozdílech mezi odhady a snažili se najít kompromis, či vysvětlit svůj pohled na věc, pokud se odhady od sebe příliš lišily. Společně dohodnutý odhad byl poté evidován do nástroje Jira (viz Obr. 15 v sekci Time Tracking).
4. Takto připravené úkoly byly zařazeny na začátek Product Backlogu, a během Sprint Planningu byly vybrány a přidány do Sprint Backlogu. Poté byly přiřazeny jednotlivým členům týmu a naplánovány ke zpracování do konkrétního sprintu.



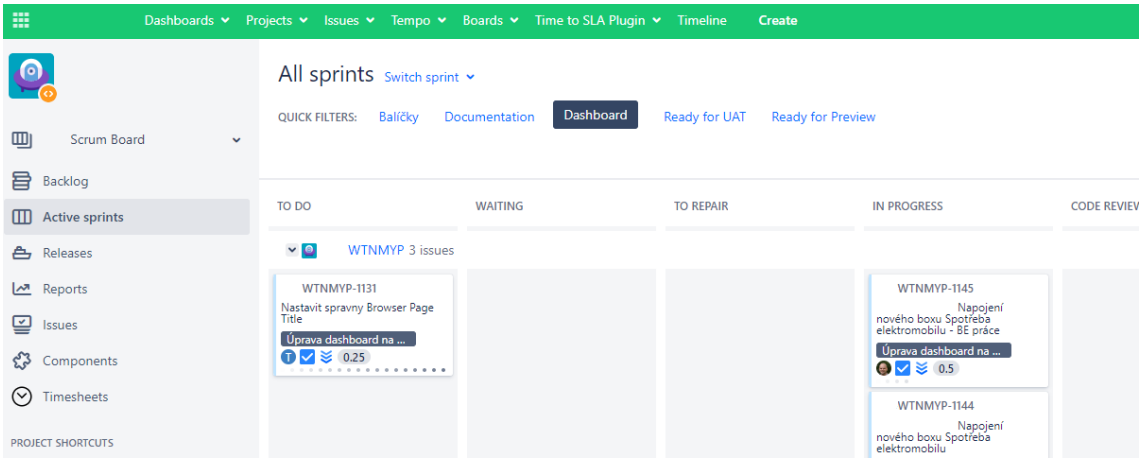
Obr. 15 – Task v Jira
Zdroj: snímek obrazovky z nástroje Jira

Díky groomingu tým dokázal lépe porozumět tomu, co klient očekává od výsledného produktu. Tým mohl diskutovat o detailech požadavků, klást otázky a případně navrhovat alternativní řešení. Zároveň to byla možnost přezkoumat úkoly ze Sprint Backlogu a upřesnit je tak, aby se do sprintu plánovaly pouze úkoly, které obsahují veškeré potřebné informace. Během groomingu měl tým také prostor na identifikaci závislosti mezi úkoly a rizik spojených s jejich implementací. Ty pak bylo možné minimalizovat nebo alespoň vědomě zohlednit při plánování prací.

Planning a definování sprintu

Planning probíhal pravidelně každých 14 dní, vždy jeden den před zahájením nového sprintu, a trval přibližně 2 hodiny. Na schůzce se sešel tým, aby se společně s Product Ownerem naplánovalo, jaké požadavky se budou realizovat v novém sprintu. Na začátek Product Owner představil Sprint Backlog a prošel s týmem požadavky a jejich priority. Tým poté zkontroloval, zda dává pořadí jednotlivých kroků po sobě smysl. Bylo totiž potřeba myslet na to, že frontendové práce mohou začít až poté co byl připraven backend, čili aby frontendový vývojář měl připravený základ, na kterém mohl začít vyvíjet svou část. Na základě odhadů, které byly domluveny na groomingu, se tým rozhodl, jaké úkoly z Product Backlogu půjdou

do sprintu a jednotlivé úkoly si mezi sebou členy týmu rozdělili. Po dokončení Sprint Planningu tým věděl, jaké úkoly je potřeba splnit, a kdo na nich bude pracovat.



Obr. 16 – Sprint v Jira
Zdroj: snímek obrazovky z nástroje Jira

Jak pro tým dodavatele, tak pro Product Ownera ze strany klienta bylo důležité, aby Sprint Planning probíhal otevřeně a v transparentní atmosféře. To proto, aby obě strany věděly, jaký výstup mohou na konci sprintu očekávat.

Na konci každého sprintu se provádělo Sprint Review, jehož účelem bylo prezentovat výsledky týmu, zhodnotit, čeho bylo dosaženo a získat zpětnou vazbu od zainteresovaných stran. Tato schůzka byla interaktivní a každý člen týmu prezentoval své výsledky pomocí ukázek zpracovaného úkolu nebo funkcionality, což umožňovalo zúčastněným stranám lépe vizualizovat to, co bylo dokončeno, v jaké kvalitě, a jak to funguje. Na Sprint Review by měli být pozváni všichni zainteresovaní včetně klienta, koncových uživatelů a další týmů, které byly spojeny s vývojem produktu. Nicméně v praxi se zúčastňoval jen Product Owner a jeho nadřízený ze strany klienta.

Během Sprint Review tým nejen prezentoval výsledky a odpovídal na otázky zainteresovaných stran, ale také zdůrazňoval neúspěchy a jejich příčiny, a rovněž ukazoval, jakou hodnotu přinesly dokončené úkoly. Tato schůzka byla důležitá, protože umožňovala týmu získat zpětnou vazbu od zainteresovaných stran, díky čemuž spolupráce mezi týmem a zákazníkem byla více transparentní. Na základě

zpětné vazby mohl tým také upravovat Product Backlog a plánovat další sprinty tak, aby mohl co nejefektivněji pracovat a poskytovat co největší hodnotu pro uživatele.

Daily Scrum/Stand-up meeting

Stand-up (SU), bylo krátké každodenní setkání týmu a sloužilo k tomu, aby se tým mohl pravidelně informovat o tom, jak se posouvají práce na projektu, a zda se vyskytl problém, který blokuje vývoj aplikace.

SU probíhal následovně:

1. Scrum Master zahájil setkání předáním informací, které daný den potřeboval týmu sdělit.
2. Každý člen týmu následně sdělil ostatním novinky o tom, na čem pracoval minulý den, jaké problémy řešil, a jaké aktivity má v plánu pro následující den. Tým se snažil být otevřen kritice a připraven na spolupráci při řešení problémů.
3. Pokud byly během SU oznámeny nějaké problémy, tým společně hledal nejlepší způsob, jak je vyřešit.

Cílem tedy bylo, aby případné překážky byly odstraněny a práce mohla co nejdříve pokračovat. Celý SU byl vždy krátký, rychlý a účelný, a každý člen týmu měl obvykle k dispozici 1-2 minuty na prezentaci jejich informací. Zúčastnění měli po tomto setkání jasno o tom, co musí sami udělat, nebo jak mohou naopak pomoci ostatním členům týmu.

Retrospektiva

Během vývoje projektu Nástěnka probíhala retrospektiva 2x do měsíce. Během ní se zhodnotila práce za poslední období (od poslední retrospektivy), identifikovaly se oblasti, které byly úspěšné, a případně se hledala další místa pro zlepšení procesů.

Retrospektiva se skládala z několika kroků. Začalo to tím, že každý člen týmu odpověděl na otázky “Co se nám povedlo/nepovedlo?”, “Co bychom mohli zlepšit?”, a koho by chtěl pochválit a za co. Následně byl vytvořen seznam s úspěšnými body

toho, co se povedlo, a co se tým naučil, a také seznam oblastí, které vyžadují pozornost do budoucna, aby se problémy neopakovaly.

Po sepsání seznamu s úspěchy a neúspěchy začal tým identifikovat jejich příčiny, tedy proč se něco povedlo a něco nikoliv, a snažil se co nejpřesněji definovat důvody, které stály za těmito úspěchy a neúspěchy. To umožnilo získat lepší pochopení toho, co fungovalo a co ne, a umožnilo týmu soustředit se na oblasti, které potřebovaly zlepšení. Dále se vybralo několik hlavních bodů k řešení, a tým sestavil preventivní opatření, aby se v budoucnu podařilo dosáhnout lepších výsledků. Jednotlivé kroky se zadaly do Sprint Backlogu jako úkol, a přiřadil se k nim řešitel, aby bylo zajištěno, že tyto kroky budou opravdu řešeny.

Retrospektiva se ukončovala tím, že se shrnul výsledek diskuzí a akční kroky. Tato schůzka umožňovala týmu reflektovat jejich práci, procesy a nalézt možnosti pro zlepšení. Vedla tým ke zvýšení kvality a efektivity práce a k podpoře transparentnosti vývoje projektu.

Na závěr je třeba zdůraznit důležitost toho, aby celý tým bral retrospektivu vážně a pravidelně se jí zúčastňoval. Následně se taková schůzka může stát součástí přirozeného zlepšení procesů.

Výsledek projektu Nástěnka

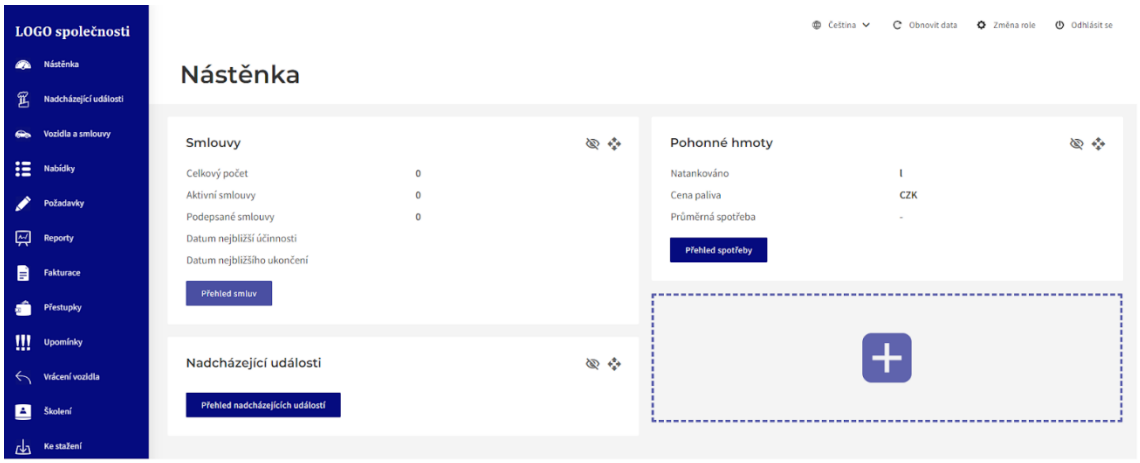
Ukázkový projekt vývoje nové nástěnky portálu byl realizován pro zákazníka, který oslovil dodavatele s poptávkou na vývoj určité sady funkčních požadavků, a také s vizí na vylepšení grafického vzhledu aktuální nástěnky.

Doba trvání projektu byla 2,5 měsíce, začal v únoru 2023 a byl úspěšně dokončen v polovině dubna 2023. Do projektu byly zapojeni 4 vývojáři, 1 tester a 1 UX/UI designer. Celkový odpracovaný čas na projektu činil 300 hodin, zahrnující operativu, základní analýzu a samotný vývoj. Původně byla pracnost vývoje odhadnuta na 200 hodin, avšak v kapitole 6.6 jsou popsány důvody, které vedly ke navýšení pracnosti vývoje, mezi tyto důvody patřila zastupitelnost, nedostatečná analýza a nedodržení součinnosti ze strany klienta.

Průběh projektu byl poznamenán několika faktory, které měly vliv na jeho trvání. Jednak klient řešil paralelně několik dalších projektů, z nich některé měly vyšší prioritu, zároveň bylo nutné počkat na zapracování úpravy v systému třetí

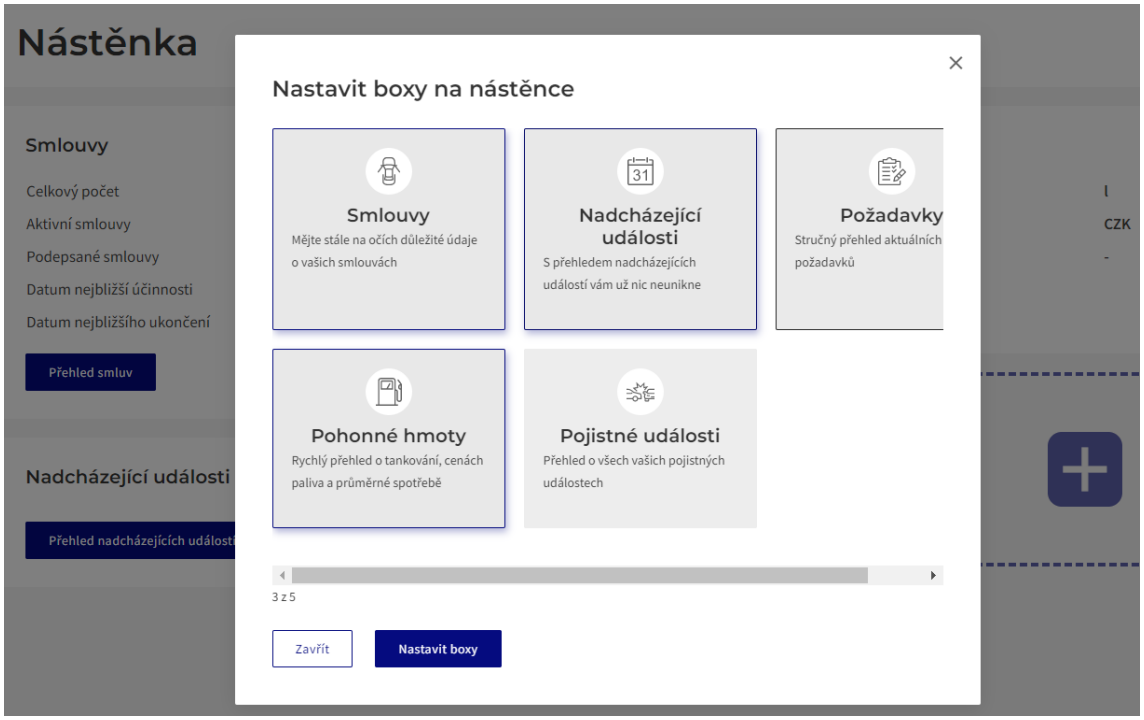
strany (úprava procedur v databázi), která byla na projekt nástěnky navázána, což také přispělo ke zdržení dokončení projektu.

Původní tři požadavky (dynamičnost, možnost uživatelské volby, sjednocení všech boxů nástěnky do jednoho typologického boxu) byly doplněny o další nové požadavky. Jednalo se o animace při přidání a odebírání boxů na plochu, či vizuální vylepšení, aby nástěnka vypadala moderněji. Také byly vytvořeny nové grafické ikony, které mohou být v budoucnu využity v celém portálu, a došlo k celkové změně vizuálního stylu portálu, jak lze vidět na Obr. 17 a Obr. 18. Po změně barevného schématu na nové nástěnce bylo nezbytné dočasně upravit i zbytek portálu, aby barvy byly konzistentní. Díky tomu působí celý portál novějším dojmem, i když z hlediska logiky portálu byla zatím modernizovaná jen jedna záložka (Nástěnka).

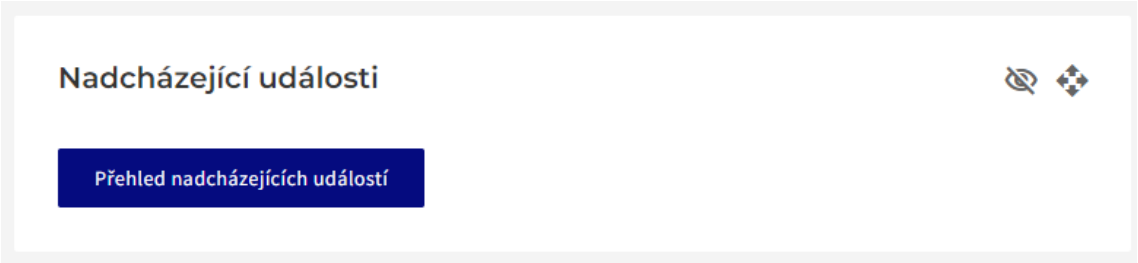


Obr. 17 – Nástěnka nového portálu
Zdroj: snímek obrazovky z nového portálu

Projekt nové nástěnky nyní splňuje jak funkční požadavky zákazníka, tak i požadavek na nový vizuální styl a další vylepšení. Vizualizace nové nástěnky je uvedena na Obr. 17. Přestože byl projekt náročný, zákazník byl s výsledkem spokojen, a uvedení produktu do provozu (produkční release) je plánováno na květen 2023.



Obr. 18 – Přidání boxů v novém portálu
Zdroj: snímek obrazovky z nového portálu



Obr. 19 – Nabídka v novém portálu
Zdroj: snímek obrazovky z nového portálu

6.5 Analýza a řešení problémů na ukázkovém projektu

V životním cyklu každého projektu dochází k více či méně očekávaným rizikovým situacím, které mohou negativně, v některých případech i pozitivně, ovlivnit kvalitu, cenu či termín dodání finálního produktu.

Úkolem projektového manažera je včas tato rizika rozpoznat, popsat, patřičně je komunikovat a snažit se minimalizovat jejich negativní dopad.

Zmiňované modelové situace vycházejí z reálných rizik identifikovaných a mitigovaných v průběhu popisovaného projektu v diplomové práci.

Jedním z rizikových faktorů každého projektu je absence řádné analýzy implementovaného řešení. V případě nedostatečné analýzy hrozí riziko neočekávané změny rozsahu projektu a s tím spojené riziko vícenákladů a průtahů dodávky finálního řešení.

V případě implementace jakéhokoliv komplexního požadavku je naprosto zásadní provést detailní předimplementační analýzu. Díky tomu se dá předejít velkému množství rizik hrozících v případě absence této přípravy. V praxi se lze setkat s případy, kdy je tým tlačen rozsah předimplementační analýzy výrazně zúžit. Děje se tak například z důvodu časového presu či omezených finančních prostředků. V každém z těchto případů je vždy nutné detailně vydefinovat vzniklá rizika a včas a důrazně je komunikovat s klientem.

V případě projektu popisovaného v diplomové práci se s podobným tlakem ze strany klienta tým setkával. Týmu byla schválena pouze minimální dotace na předimplementační analýzu, což v případě komplikovanějších změn vedlo nejčastěji ke komplikacím v podobě změn rozsahu požadovaného řešení ať již z důvodu nedomyšleného byznys požadavku, nebo z důvodu technických překážek, které by standardní podrobná analýza včas odhalila. To následně vedlo k navyšování operativy spojené s řešením těchto rizik a vyjednáváním o změně rozsahu projektu. Během těchto jednání klesala utilizace týmu, jelikož se čekalo na aktualizaci pokynů. Jedním z důsledků bylo zahození neimplementované části funkcionality a bylo nutné přeplánování celého sprintu a definování nového. Všechny tyto popsané dopady vedly logicky k průtahům s dodávkou daného požadavku, a především k výraznému navýšení finálních nákladů. V konečném výsledku tyto vícenáklady výrazně převýšily úspory spojené s levnější analýzou. To následně vedlo ke konfliktům s klientem, který nechtěl tyto náklady akceptovat. Časté opakování takových situací může zapříčinit frustraci celého týmu.

Následující modelová situace popisuje dopad rizika způsobeného nedodržením smluvených součinností ze strany klienta a nedostatečnou předimplementační analýzou. Vzniklé komplikace měly ve výsledku dopad na rozsah, cenu i termín dodání požadavku/projektu.

V rámci definice rozsahu projektu bylo oběma stranami odsouhlaseno řešení jednoho z byznys požadavků implementací nástroje poskytovaného třetí stranou.

Klient tento postup požadoval z interních důvodů a zavázal se dodat všechny nutné podklady a součinnost pro realizaci tohoto požadavku. Před začátkem implementace obdržel tým podklady pro analýzu proveditelnosti schváleného řešení. Byla požadována pouze obecná analýza technické použitelnosti vybraného nástroje. Analýzu dostupnosti potřebných dat provedl klient samostatně. Na tomto základě byly definovány odhady pracnosti a termín realizace. V průběhu implementace tým narazil na několik závažných problémů silně ovlivňující dodávku. Data z požadovaného nástroje byla dostupná pouze pro povolené IP adresy a jejich poskytovatel reagoval jen velmi sporadicky. To způsobilo průtahy s implementací jednotlivých funkcionalit portálu. Zároveň se navyšovaly náklady na operativu spojenou s komunikací. Riziko bylo eskalováno na Product Ownera, který zjednal nápravu rychlosti komunikace mezi třetí stranou a týmem.

V souvislosti se zmíněným nástrojem vznikla další neočekávaná překážka. Data, která pomocí tohoto nástroje měla být zobrazována na frontendu vyvíjené aplikace, neobsahovala velkou část potřebných dat definovaných rozsahem požadavku. Po konzultaci s poskytovatelem implementovaného nástroje bylo zjištěno, že požadovaná data nebude možné poskytnout. Riziko bylo opět komunikováno na Product Ownera. Vzhledem k důležitosti chybějících dat bylo rozhodnuto přistoupit k náhradnímu řešení. Po konzultaci s technickým týmem klienta bylo zjištěno, že chybějící data je možné získat z existující databáze, ovšem neexistovala žádná služba, která by umožnila tato data zpřístupnit pro frontend aplikace. Tým navrhl vystavění chybějící služby a připravil odhady náročnosti na její implementaci, které enormním způsobem navýšily náklady na dodávku řešení. S Product Ownerem bylo nutné projednat změnu rozsahu dodávky a její negativní dopad na cenu i termín dodání. Vzhledem ke kritičnosti byznys požadavků, kterých se chybějící data týkala, byla změna rozsahu projektu schválena. Na tomto modelovém příkladu je patrné, že úspory na řádné přípravě projektu mohou způsobit nečekané komplikace.

Dalším rizikovým faktorem je zastupitelnost členů týmu. Vždy je třeba počítat s tím, že některý člen týmu může být na určitou dobu z různých důvodů nedostupný. V případě neplánované nedostupnosti může takový výpadek lidských zdrojů vést k průtahům s dodávkou jednotlivých požadavků.

Řešením je zajištění zástupu pro danou roli. V takovém případě je však nutné počítat s tím, že nemusí být dodrženy původní odhady spojené s prací, kterou dočasně přebírá zastupující člen týmu. Nového člena je nutné uvést do kontextu projektu a daných požadavků, které má převzít.

Na ukázkovém projektu v diplomové práci se v takových případech Scrum Master setkal s nepochopením nedodržených odhadů. Následné diskuze zbytečně dále zvyšovaly operativu na projektu. Proto je nutné vždy zvážit, zda negativa spojená dočasným pozastavením některých prací jsou natolik závažná, že je vhodnější jít cestou dočasného zástupu spojeného s náklady popsanými výše.

7 Závěry a doporučení

V práci byly analyzovány různé metody řízení IT projektu a na základě dostupné literatury a nabytých znalostí z praxe byl připraven doporučený postup pro výběr optimální metody. Jedním z hlavních přínosů práce je doporučení pro samotný postup, který je prezentován v podobě tabulky. Tabulka umožňuje snadné a efektivní orientování v postupu výběru metody řízení IT projektu a reflektuje tak potřeby cílové skupiny, kterou jsou začínající IT projektoví manažeři.

V doporučeném postupu je rozlišeno několik základních kritérií projektu a indikátory naznačující, která metoda by mohla být pro projekt s podobnou charakteristikou vhodná. Výběr vhodné metody pro řízení IT projektu dle tohoto vlastního postupu je demonstrován na příkladu projektu Nástěnka z praxe.

Na základě analýzy zadání projektu, a po identifikaci jednotlivých kritérií a indikátorů, bylo zřejmé, že pro dosažení maximální efektivity řízení projektu by měla vyhovovat nejlépe metoda Scrum. Během realizace samotného projektu se ukázalo, že zvolená metoda byla správnou volbou. Bylo totiž nezbytné flexibilně reagovat na požadavky klienta, jako např. dokreslení grafiky či změny a upřesnění zadání v průběhu projektu, včetně konzultace a analýzy integrovaných systémů třetí strany. Díky zvolené metodě mohl klient změny zpracovávat průběžně, místo čekání na dokončení a akceptaci projektu. Výhodou pro dodavatelskou firmu bylo, že tato metoda lépe kryje riziko navýšení pracnosti, i když ani v tomto ohledu nemusí být stoprocentně účinná, jak popisuje kapitola řešení problémů.

Ne všechny oblasti řízení projektu v praxi přesně odpovídaly metodice Scrum. V kapitole průběh projektu autorka popisuje, jak byla metoda upravena pro potřeby klienta v určitých oblastech. Příkladem je využívání Story Pointů, které agilní metody doporučují pro odhadování komplexity úkolů, a jejíž postup je jasně popsaný. Na ukázkovém projektu byl však redukován na odhad čisté pracnosti na základě požadavku klienta a omezení času strávených na schůzkách nad těmito odhady. Efektivita tohoto rozhodnutí není v práci zkoumána.

V praxi je stále častěji používán hybridní přístup, který kombinuje prvky agilních a Waterfall metod, a snaží se využít výhod obou metod a může být tak

v některých případech nejvíce účinný. Umožňuje totiž flexibilitu a adaptabilitu agilního přístupu, zatímco zachovává strukturu a plánování Waterfall metodiky.

Projektový management v oblasti IT je dynamický a neustále se rozvíjející se obor, který musí reagovat na rychlý technologický pokrok. Jedním z hlavních trendů, který bude do budoucna promlouvat do řízení projektů je bezesporu automatizace a využití umělé inteligence. To umožní projektovým manažerům soustředit se na strategické aspekty projektů a snížit operativu na manuálních a repetitivních činnostech. Je velmi pravděpodobné, že nové metody a techniky povedou k ještě vyšší efektivitě a úspoře nákladů. Tyto nové metody mohou využívat například analýzy velkých dat, které pomáhají s přesnějším plánováním a řízením rizik.

Limity této diplomové práce jsou dány využitím metod a praxe využívané v dodavatelské firmě a jejími specifiky - např. velikost firmy, obvyklý rozsah IT projektů nebo realizačních týmů. Proto je další výzkum v oblasti výběru vhodné metody pro řízení IT projektu žádoucí. Doporučený postup by bylo možné rozšířit jak do hloubky, tak do šířky. Z hlediska prohloubení by bylo možné vytvořit rozhodovací strom, který by byl podložen daty z praxe. V rámci rozšíření je možné zahrnout další metody (Lean, PMBOK, apod.) či rozpracovat postup pro jiná odvětví mimo obor IT.

8 Seznam zkratek a převzatých odborných výrazů

Akční kroky – konkrétní kroky nebo úkoly, které je třeba provést k dosažení určitého cíle nebo řešení problému

Backend – část webové aplikace nebo softwaru, která zpracovává data, pracuje s databází a logickými aplikacemi; tato část není viditelná uživateli, ale zajišťuje správnou funkčnost celé aplikace

Background – základ znalostí

Business Analyst (BA) – byznys analytik

Development & Operations (DevOps) – přístup, který je založen na sadě nástrojů a postupů urychlující dodávku softwaru kombinací a automatizací vývojových a IT provozních týmu

Dry Run testování – testování softwaru, které se provádí, aby se zjistilo, jak se bude systém chovat v případě poruchy

Framework – sada nástrojů a modulů, které lze znovu použít pro různé projekty

Frontend – část webové aplikace nebo softwaru, která se zabývá tvorbou uživatelského rozhraní (GUI) a interakcí webového produktu s uživatelem

Fix Time Fix Price (FTFP) – způsob řízení projektu, který zaručuje pevný rozpočet projektu bez ohledu na čas a náklady

Funkce Drag and Drop – posouvání komponent v rámci grafického rozhraní pomocí funkce uchopení elementu a přesunutí na jinou pozici

Graphic User Interface (GUI) – uživatelské grafické rozhraní

IT – informační technologie

IT Operations (ITOps) – IT Operations tým

Know-how – souhrn odborných znalostí/dovedností

Layouty – rozmístění prvků na webových stránkách

Lesson Learned – ponaučení ze zkušenosti z minulých činnosti

Maintenance – údržba systému (zlepšení výkonu, oprava chyb)

Manday (MD) – člověkoden – jeden den práce jednoho pracovníka

Project Manager (PM) – projektový manažer

Release – vypublikování IT služeb, softwaru nebo aplikace k veřejnému používání

Risk Log – dokument, který se používá k identifikaci potenciálních rizik na projektu

Stakeholder – zainteresovaná strana

Task – reprezentace jednotky práce nebo úkolu v nástroji Jira

User Experience (UX) – aspekty interakce uživatele s produktem, včetně jeho vzhledu, použitelnosti, intuitivity a spokojenosti uživatele

User Interface (UI) – označení vizuálního interaktivního prvku na webové stránce, v počítačovém programu či mobilní aplikaci (např. tlačítka, menu, formuláře, nebo dialogová okna)

Workflow – termín pro popis postupu, procesů nebo série kroků, které jsou prováděny k efektivnímu dosažení určitého cíle

Změnové požadavky – požadavek na změnu oproti původnímu zadání

9 Seznam použité literatury

[1] AGILE ALLIANCE, 2022. What is the Agile Manifesto? Agile Alliance [online]. ©2001-2022 Agile Manifesto Authors [cit. 2023-04-15]. Dostupné z: https://www.agilealliance.org/agile101/the-agile-manifesto/

[2] ATLASSIAN, 2023. Git Merge [online]. ©2023 Atlassian [cit. 2023-01-13]. Dostupné z: https://www.atlassian.com/git/tutorials/using-branches/git-merge

[3] ATLASSIAN, 2023a. Continuous Delivery Pipeline 101 | Atlassian . Collaboration software for software, IT and business teams [online].©2023 Atlassian [cit. 13.01.2023]. Dostupné z: https://www.atlassian.com/continuous-delivery/principles/pipeline

[4] ATLASSIAN, 2023b. Collaboration software for software, IT and business teams [online]. ©2023Atlassian [cit. 15.04.2023]. Dostupné z: https://www.atlassian.com/git/tutorials/making-a-pull-request

[5] ATLASSIAN, 2023c. Making a Pull Request. Atlassian.com [online]. ©2023 Atlassian [cit. 2023-04-08]. Dostupné z: https://www.atlassian.com/git/tutorials/making-a-pull-request

[6] ATLASSIAN, 2023d. From to-do to done with Jira kanban boards [online]. ©2023 Atlassian [cit. 2023-05-02]. Dostupné z: https://www.atlassian.com/software/jira/features/kanban-boards

[7] AWS, 2023. What are Microservices? | AWS. Cloud Computing Services - Amazon Web Services (AWS) [online]. ©2023, Amazon Web Services, Inc. or its affiliates. All rights reserved. [cit. 13.01.2023]. Dostupné z: https://aws.amazon.com/microservices/

[8] BIGELOW, Stephen J.,2020. Docker. Techtarget [online]. Březen 2020 [cit. 2023-04-08]. Dostupné z: https://www.techtarget.com/searchitoperations/definition/Docker

[9] DOLEŽAL, Jan. a kol., 2016. Projektový management: Komplexně, prakticky a podle světových standardů [online] Praha 7: Grada Publishing, 2016 [cit. 2022-08-07]. ISBN 978-80-271-9066-9. Dostupné z: https://books.google.cz/books?id=70c-DAAAQBAJ&printsec=frontcover&dq=definice+projektov%C3%A9ho+managementu&hl=cs&sa=X&ved=2ahUKEwjQ1-zVw7T5AhUjJcUKHTfdDRAQ6AF6BAgHEAI#v=onepage&q=definice%20projektov%C3%A9ho%20managementu&f=false

[10] EDUREKA, 2022. What is Jenkins? | Jenkins For Continuous Integration | Edureka. Instructor-Led Online Training with 24X7 Lifetime Support | Edureka, 2022 [online]. ©2023 Brain4ce Education Solutions Pvt. Ltd. All rights Reserved. [cit. 13.01.2023]. Dostupné z: https://www.edureka.co/blog/what-is-jenkins/

[11] HAMILTON, Thomas, 2023. Agile vs Waterfall – Difference Between Methodologies. Guru99 [online]. Březen 2023 [cit. 2023-01-21]. Dostupné z: https://www.guru99.com/waterfall-vs-agile.html

[12] HOORY, Leeron a BOTTORFF Cassie, 2022. What Is Waterfall Methodology? Here's How It Can Help Your Project Management Strategy. Forbes Advisor [online]. Forbes

Media, Mar 2022 [cit. 2022-09-04]. Dostupné z: https://www.forbes.com/advisor/business/what-is-waterfall-methodology/

[13] CHARVAT, Jason, 2023. Project management methodologies [online]. Canada: John Wiley, 2003 [cit. 2022-12-18]. Dostupné z: https://books.google.cz/books?hl=cs&lr=&id=Pd5AnH3ZA1AC&oi=fnd&pg=PR9&dq=waterfall+methodology+project+management&ots=Q9DRLmzdOH&sig=L9T4w0y0t4gpVOLMZ0a1rbTADeU&redir_esc=y#v=onepage&q&f=false

[14] CHRISTENSSON, Per, 2021. Build. Techterms.com [online]. Říjen 2021 [cit. 2023-04-08]. Dostupné z: https://techterms.com/definition/build

[15] IBM, 2021. HTTP requests. IBM [online]. IBM Corporation, 2021 [cit. 2023-04-08]. Dostupné z: https://www.ibm.com/docs/en/cics-ts/5.3?topic=protocol-http-requests

[16] KAREŠOVÁ, Agáta, 2022. Plán projektu a návrhy na jeho zefektivnění [online]. Plzeň, 2022 [cit. 2022-12-18]. Dostupné z: https://dspace5.zcu.cz/bitstream/11025/47905/1/BP_Karesova.pdf. Bakalářská práce. ZÁPADOČESKÁ UNIVERZITA V PLZNI. Vedoucí práce Ing. Radim Špicarov.

[17] KORNEVA, Alisa, 2020. Waterfall или Agile: как выбрать методологию управления проектом?. Azoft [online]. Srpen 2020 [cit. 2023-01-21]. Dostupné z: https://www.azoft.ru/blog/waterfall-agile/

[18] LUCKY HUNTER, 2023. ПОДБОР IT ПЕРСОНАЛА: МЕНЕДЖЕРЫ В IT. КАК НАЙТИ IT СПЕЦИАЛИСТОВ ДЛЯ ВАШЕЙ КОМПАНИИ? Lucky Hunter Blog [online]. © 2023 Lucky Hunter [cit. 2023-04-19]. Dostupné z: https://blog.luckyhunter.io/menejiry-v-it

[19] LUTKEVICH, Ben, 2022. Application programming interface (API). Techopedia [online]. Techopedia, Prosinec 2022 [cit. 2023-04-08]. Dostupné z: https://www.techtarget.com/searchapparchitecture/definition/application-program-interface-API

[20] McCORMICK, Mike, 2012. Waterfall vs. Agile Methodology [online]. Září 2012[cit. 2022-12-18]. Dostupné z: http://www.mccormickpcs.com/images/Waterfall_vs_Agile_Methodology.pdf

[21] MICROSOFT, 2022a. Save your work with commits. Microsoft [online]. 2022 [cit. 2023-04-08]. Dostupné z: https://learn.microsoft.com/en-us/azure/devops/repos/git/commits?view=tfs-2018&tabs=visual-studio-2022

[22] MICROSOFT, 2022b. Share code with push. Microsoft [online]. 2022 [cit. 2023-04-08]. Dostupné z: https://learn.microsoft.com/en-us/azure/devops/repos/git/pushing?view=azure-devops&tabs=visual-studio-2022

[23] NĚMEC, Vladimír, 2002. Projektový management. Praha Grada Publishing, 2002, ISBN 80-247-0392-0

[24] ORACLE, 2023a. What Is a Database. Oracle. Cloud Applications and Cloud Platform [online]. ©2023 Oracle [cit. 13.01.2023]. Dostupné z: https://www.oracle.com/database/what-is-database/

[25] PERENS, Algis, 2011. E-курс для подготовки к экзамену по квалификации специалиста информационной технологии: Управление проектами. EUCIP [online]. Таллинн: Институт вычислительной техники Таллиннского технического университета., 2011 [cit. 2023-04-07]. Dostupné z: https://eopearhiiv.edu.ee/e-kursused/eucip/juhtimine_vk/513_____.html

[26] PITTET, Sten, 2023. Continuous integration vs. delivery vs. deployment. Atlassian [online]. ©2023 Atlassian [cit. 2023-01-13]. Dostupné z: https://www.atlassian.com/continuous-delivery/principles/continuous-integration-vs-delivery-vs-deployment

[27] POSPÍŠIL, Vojtěch, 2021. Co jsou a k čemu slouží aplikační servery? Magazín ZonerCloud [online]. ZONER, 2021 [cit. 2023-01-13]. Dostupné z: https://www.zonercloud.cz/magazin/co-jsou-a-k-cemu-slouzi-aplikacni-servery

[28] REDHAT, 2020. What is a REST API? [online]. ©2023 Red Hat, Inc. [cit. 13.01.2023]. Dostupné z: https://www.redhat.com/en/topics/api/what-is-a-rest-api

[29] ROUSE, Margaret, 2021. What is a Graphical User Interface (GUI)? Definition from Techopedia. Techopedia: Educating IT Professionals To Make Smarter Decisions [online]. Květen 2021 [cit. 13.01.2023]. Dostupné z: https://www.techopedia.com/definition/5435/graphical-user-interface-gui

[30] RUBIN, Kenny, 2012. Essential Scrum: Practical Guide to the Most Popular Agile Process. Addison Wesley, 2012. ISBN 9780137043293.

[31] SCRUM, 2023. What is Scrum? | Scrum.org. Home | Scrum.org [online]. ©2023 Guidelines [cit. 14.01.2023]. Dostupné z: https://www.scrum.org/resources/what-is-scrum

[32] SERVICENOW, 2023. What is IT operations (ITOps)? [online]. ©2023 ServiceNow. All rights reserved. [cit. 2023-02-25]. Dostupné z: https://www.servicenow.com/products/it-operations-management/what-is-it-operations.html

[33] SHELDON, Robert, 2023. What is Git?. Tech Target - IT Operation [online]. TechTarget, 2023 [cit. 2023-04-08]. Dostupné z: https://www.techtarget.com/searchitoperations/definition/Git

[34] SCHWABER, Ken a SUTHERLAND Jeff. 2020. The Scrum Guide [online]. Listopad 2020 [cit. 2022-12-18]. Dostupné z: https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-US.pdf

[35] SOLONTAI, Viktoriia, 2022. Кто такой менеджер IT-проектов. Tproger [online]. Ренессанс Кредит, 2022 [cit. 2023-01-08]. Dostupné z: https://tproger.ru/articles/kto-takoj-menedzher-it-proektov/

[36] SUMOLOGIC, 2023. Software deployment - definition & overview | Sumo Logic. Cloud Log Management, Monitoring, SIEM Tools | Sumo Logic [online]. ©2023 Sumo Logic [cit. 13.01.2023]. Dostupné z: https://www.sumologic.com/glossary/software-deployment/

[37] SUPERTOKEN TEAM, 2022. What is Cross Origin Resource Sharing (CORS)? SuperToken [online]. ©2022 SuperTokens [cit. 2023-01-13]. Dostupné z: https://supertokens.com/blog/what-is-cross-origin-resource-sharing

[38] SVOZILOVÁ, Alena, 2011. Projektový management [online]. Druhé vydání. Praha 7: Grada Publishing, 2011 [cit. 2022-08-07]. ISBN 978-80-247-7428-2. Dostupné z: https://books.google.cz/books?id=zAJbAgAAQBAJ&pg=PA81&dq=definice+projektov%C3%A9ho+managementu&hl=cs&sa=X&ved=2ahUKEwjQ1-zVw7T5AhUjJcUKHTfdDRAQ6AF6BAgJEAI#v=onepage&q&f=false

[39] ŠEBEK, Václav, 2014. Plánování a řízení projektů. SlideServe [online]. © 2022 SlideServe, 2014 [cit. 2022-11-25]. Dostupné z: https://www.slideserve.com/alyssa-harrington/9-pl-nov-n-a-zen-projekt

[40] ŠOCHOVÁ, Zuzana a KUNCE Eduard, 2014. Agilní metody řízení projektů. Brno, 2014: Computer Press. ISBN 978-80-261-4194-6.

[41] TOROSYAN, Elena a TULKINA Anastasiia, 2020. Критерии выбора методологии управления IT-проектами. Петербургский экономический журнал [online]. 2020, 99-107 [cit. 2023-04-07]. Dostupné z: https://cyberleninka.ru/article/n/kriterii-vybora-metodologii-upravleniya-it-proektami

[42] WASEEM, Ahad, 2022. Waterfall Methodology: History, Principles, Stages & More. Management Library [online]. Copyright Paradise Media, 2022 [cit. 2022-09-04]. Dostupné z: https://managementhelp.org/waterfall-methodology

[43] АБЛИЦОВА, Аня, 2021. Как стать проджект-менеджером в IT. Laba [online]. ©2015-2023 Laba All Rights Reserved [cit. 2023-01-08]. Dostupné z: https://l-a-b-a.com/blog/2587-kak-stat-prodzhekt-menedzherom-v-it?utm_source=google&utm_medium=cpc&utm_campaign=blog-dsa&utm_content=search_DSA&gclid=CjwKCAiAqt-dBhBcEiwATw-ggJujohmFxntg4sy1RyDr7M7lRnxrSlJ56TpudShpRLjRF6izz-WuJBoCA0gQAvD_BwE

Univerzita Hradec Králové
Fakulta informatiky a managementu

Zadání diplomové práce

Autor:	**Bc. Mariya Nagornyak**
Studium:	I2000088
Studijní program:	N0688A140001 Informační management
Studijní obor:	Informační management
Název diplomové práce:	**Metody řízení IT projektu**
Název diplomové práce AJ:	Methods of IT Project Management

Cíl, metody, literatura, předpoklady:

Cílem práce je porovnat vybrané metody pro řízení IT projektů a na případové studii ukázat, jak se používají v praxi.

Struktura diplomové práce:

1. Úvod

2. Teoretická část (popis vybraných metod)

3. Praktická část (použití vybrané metody v praxi na reálnem IT projektu)

4. Závěr a doporučení

Knihy:

DOLEŽAL, Jan. *Projektový management podle IPMA*. 2. vydání. Praha: Grada Publishing, 2012. ISBN 978-80-247-4275-5.

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

DOLEŽAL, Jan. *Projektový management*. Praha: Grada Publishing, a.s., 2016. ISBN 978-80-247-5620-2.

Zahraniční zdroje:

Databáze Web of Science

- KITTLAUS, Hans-Bernd. *Software Product Management* [online]. 2nd Edition. Rheinbreitbach, Germany: Springer, 2017 [cit. 2022-10-22]. ISBN ISBN 978-3-662-65116-2. Dostupné z: https://bibis.ir/science-books/management/2022/Software%20Product%20Management%20The%20ISPMA%20by%20Hans-Bernd%20Kittlaus_bibis.ir.pdf
- GUSTAVSSON, Tomas. Team Performance in Large-Scale Agile Software Development. *Lecture Notes in Information Systems and Organisation* [online]. Springer, 2022, (55) [cit. 2022-10-22]. Dostupné z: https://link.springer.com/chapter/10.1007/978-3-030-95354-6_14
- ROGGENBACH, Markus a kol. Formal Methods. *Formal Methods for Software Engineering* [online]. Springer, 2022, [cit. 2022-10-22]. Dostupné z: https://link.springer.com/chapter/10.1007/978-3-030-38800-3_1
- AMETA, Upasana a kol. Scrum Framework Based on Agile Methodology in Software Development and Management. *Studies in Autonomic, Data-driven and Industrial Computing* [online]. Springer, 2021 [cit. 2022-10-22]. Dostupné z: https://link.springer.com/chapter/10.1007/978-981-16-3915-9_28

Scopus

- NEPOMUCENO, Vilmar Santos. Decision support system to project software management. *2013 IEEE International Conference on Systems, Man, and Cybernetics* [online]. 2013, 964 - 969 [cit. 2022-10-22]. ISSN 978-076955154-8. Dostupné z: https://www.scopus.com/record/display.uri?eid=2-s2.0-84893567582&origin=resultslist&sort=plf-f&src=s&sid=0b92096f6ae9c0cbf6b1ac628400a7ad&sot=a&sdt=a&sl=16&s=agile+metodology&relpos=2&citeCnt=3&searchTerm=

Zadávající pracoviště:	Katedra informačních technologií, Fakulta informatiky a managementu
Vedoucí práce:	Ing. Tereza Otčenášková, BA, Ph.D.
Oponent:	Ing. Petr Buchbauer
Datum zadání závěrečné práce:	15.10.2021