EKONOMICKÁ UNIVERZITA V BRATISLAVE
FAKULTA HOSPODÁRSKEJ INFORMATIKY

Evidenčné číslo: 103004/I/2021/421000161994

INTEGRÁCIA APLIKÁCIÍ V IT PODNIKU

Diplomová práca

2021 **Bc. Adam Chlebovec**

EKONOMICKÁ UNIVERZITA V BRATISLAVE
FAKULTA HOSPODÁRSKEJ INFORMATIKY

INTEGRÁCIA APLIKÁCIÍ V IT PODNIKU

Diplomová práca

Študijný program:	Informačný manažment
Študijný odbor:	Ekonómia a manažment
Školiace pracovisko:	Katedra aplikovanej informatiky
Vedúci záverečnej práce:	Ing. Igor Bandurič, PhD.

Bratislava 2021 **Bc. Adam Chlebovec**

Ekonomická univerzita v Bratislave
Fakulta hospodárskej informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta:	Bc. Adam Chlebovec
Študijný program:	informačný manažment (Jednoodborové štúdium, inžiniersky II. st., denná forma)
Študijný odbor:	ekonómia a manažment
Typ záverečnej práce:	Inžinierska záverečná práca
Jazyk záverečnej práce:	slovenský
Sekundárny jazyk:	anglický

Názov: Integrácia aplikácií v IT podniku

Anotácia: Moderné technológie a aplikácie sú kľúčový faktor pri tímovej práci a riadení projektov v IT sektore. Súčasným trendom je agilné riadenie projektov ktoré sprevádza potreba efektívnej komunikácie a plánovania. Jira je momentálne lídrom na trhu v oblasti IT projektového riadenia. Cieľom práce je integrovať Jira platformu do spoločnosti, ktorá využíva čisto Microsoft technológie za víziou zefektívnenia pracovnej činnosti, komunikácie a reportovania.

Vedúci:	Ing. Igor Bandurič, PhD.
Oponent:	Ing. Igor Košťál, PhD.
Katedra:	KAI FHI - Katedra aplikovanej informatiky FHI
Vedúci katedry:	Ing. Mgr. Peter Schmidt, PhD.
Dátum zadania:	13.11.2019

Dátum schválenia: 13.11.2019

Ing. Mgr. Peter Schmidt, PhD.
vedúci katedry

Čestné vyhlásenie

Čestne vyhlasujem, že som samostatne vypracoval záverečnú diplomovú prácu na základe zadania a že som uviedol kompletný zoznam použitej literatúry.

V Bratislave, dňa 1. 5. 2021

.....

Bc. Adam Chlebovec

Poďakovanie

Moje poďakovanie patrí Ing. Igorovi Banduričovi, Phd., za cenné rady a usmernenie pri práci ako aj možnosť pracovať na tejto téme. Tak isto by som chcel poďakovať spoločnosti Millennium a Ing. Martinovi Kozlovskému za pomoc a podporu pri praktickej časti.

Abstrakt

CHLEBOVEC, Adam: *Integrácia aplikácií v IT podniku*. – Ekonomická Univerzita v Bratislave. Fakulta hospodárskej informatiky; katedra aplikovanej informatiky FHI.- Vedúci záverečnej práce: Ing. Igor Bandurič, PhD. – Bratislava: FHI EU, 2021, 76 s.

Cieľom práce je integrácia vybraných Microsoft aplikácií s Jira platformou v oblasti projektového riadenia v rámci IT podniku. Práca je rozdelená do 3 hlavných kapitol a podkapitol. Obsahuje 5 tabuliek a 19 obrázkov. Prvá kapitola je venovaná súčasnej problematike. V tejto kapitole sú vymedzené pojmy ako aplikačná architektúra, integrácia, IT servis manažment, proces životného cyklu softvéru a informácie o podniku a využívaných aplikáciách. V ďalšej časti sa charakterizuje cieľ práce, metodika a metódy skúmania. Záverečná časť práce sa venuje výsledkom práce. Opisuje postupy výberu, analýzy a konfigurácie integračných nástrojov ako aj následné testovanie, zobrazenie a zhodnotenie výslednej funkcionality riešenia. Výsledkom práce je teda integrácia Microsoft aplikácií (MS Teams, Git, Azure DevOps Boards a MS exchange) s Jira platformou. Výslednú funkcionalitu sme zhodnotili a vyslovili závery.

Kľúčové slová:

Integrácia, Microsoft, Atlassian, Jira, MS Teams, Azure DevOps Server, Git

Abstract

CHLEBOVEC, Adam: *Aplications integration within IT company* – University of Economics in Bratislava. Faculty of Economic Informatics; Department of Applied Informatics. – Thesis supervisor: Ing. Igor Bandurič, PhD. – Bratislava: FHI EU, 2021, 76 p.

The goal of thesis is integration of chosen Microsoft applications with Jira platform in area of project management in IT company. Thesis is devised into three main chapters and sub-chapters. It contains 5 tables and 19 pictures. First chapter is devoted to the current state of the problem. This chapter defines terms as application architecture, integration, IT service management, software lifecycle and information about company and used applications. The next chapters define goals, work methodology and research methods. The final chapter contains results of thesis. It describe process of selection, analysis and configuration of integration tools as well as testing, presenting and summarizing of final integrated functionality. The result of thesis is integration of Microsoft applications (MS Teams, Git, Azure DevOps Boards, MS exchange) and Jira platform. We summed up final functionality and shared our conclusion.

Key words:

Integration, Microsoft, Atlassian, Jira, MS Teams, Azure DevOps Server, Git

OBSAH

ÚVOD

Moderné technológie a aplikácie sú kľúčový faktor pri tímovej práci a riadení projektov nie len v IT sektore. Súčasným trendom je agilné riadenie projektov ktoré sprevádza potreba efektívnej komunikácie a plánovania. Práve pre tento spôsob riadenia projektov je Jira jedným z lídrov na trhu. Pri riadení projektov avšak nestačí využívať iba jednu aplikáciu ale zvyčajne napríklad k Jire treba pridať napríklad ešte nástroje na komunikáciu, spoluprácu či ďalšie aplikácie na zlepšenie a zefektívnenie potrebných procesov.

V práci sa najprv oboznámime s aplikačnou architektúrou, aké typy aplikácií sa v rámci aplikačnej architektúry vyskytujú a v čom je ich výhoda. Keďže sa v práci venujeme integrovaní aplikácií je dobré vedieť, čo takéto aplikácie dokážu. Pri budovaní aplikačnej architektúry sú neodmysliteľnou súčasťou ich vzájomné integrácie, ktoré jednotlivé aplikácie spájajú spríjemňujú a zefektívňujú tak ich využívanie.

Pri uvažovaní o integrácií podnikových aplikácií je potrebné porozumieť častiam a obsahu podnikových procesov a dát. Taktiež je potrebné poznať ako sú jednotlivé procesy automatizované (prípadne neautomatizované) ako aj dôležitosť jednotlivých podnikových procesov. S jednotlivými procesmi sa oboznámime v kapitole o životnom cykle softvéru ako celku a následne o procesoch IT Servis manažmente a jednotlivých kľúčových procesov, ktoré je potrebné mať efektívne nastavenie pre čo najvyššie uspokojenie zákazníka a zároveň si zachovať ziskovosť.

Cieľom práce je integrovať Jira platformu do spoločnosti, ktorá využíva čisto Microsoft technológie za víziou zefektívnenia pracovnej činnosti, komunikácie a reportovania. Pre uskutočnenie tejto integrácie si najprv predstavíme jednotlivé aplikácie a ich využitie v rámci podniku, následne si stanovíme jednotlivé ciele, ktoré chceme integráciou dosiahnuť. Proces integrovania potom môžeme rozdeliť do troch častí. V prvej časti vyberieme nástroj (aplikáciu), ktorá nám bude integráciu zabezpečovať. Tieto aplikácie vyberieme na základe požiadaviek na funkcionalitu a taktiež s čo možno najnižšími nákladmi pre podnik. Po výbere integračnej aplikácie túto aplikáciu nainštalujeme, a na základe analýzy požiadaviek nastavíme jednotlivé parametre. Po úspešnej inštalácií a konfigurácií integráciu manuálne otestujeme na základe očakávaných používateľských scenárov.

Výsledkom sú teda integrované aplikácie v rámci podnikovej aplikačnej infraštruktúry. Výsledky následne opíšeme a zhodnotíme ich výhody, nevýhody a navrhneme ich optimálne využitie. Výsledky ako aj celá práca sa snaží ukazovať využitie nielen pre konkrétnu firmu nakoľko nezachádza do detailov procesov.

1 SÚČASNÝ STAV RIEŠENEJ PROBLEMATIKY

Problematika úspešnej integrácie podnikových aplikácií je téma, riešená v každom podniku, ktorý podstupuje proces digitalizácie. Zavádzanie nových moderných aplikácií na zvýšenie konkurencieschopnosti a efektivity do podnikov je v dnešnej dobe nevyhnutnosť. V tejto kapitole sa oboznámime s pojmami ako aplikačná architektúra podniku, integrácia, oboznámime sa s typmi procesov, ktoré využívajú integrácie v IT sektore a zároveň si predstavíme aplikácie, ktoré budeme integrovať a ich význam.

1.1 Aplikačná architektúra, intranet extranet

Pri aplikačnej architektúre sú sledovanými komponentami informačného systému jednotlivé softvérové aplikácie. Počet aplikácií je závislý od veľkosti a konkrétnych potrieb podniku. Malým podnikom zvyčajne stačí zopár jednoduchých aplikácií, no čím väčší podnik, tým väčšie sú jeho potreby ako aj na počet, tak aj na robustnosť jednotlivých aplikácií (Napríklad aplikácie na podporu výroby, personalistiky, účtovníctva, na evidenciu a podporu vzťahov so zákazníkmi, riadenie projektov či komunikáciu). [1]

Podľa spôsobu zaobstarania rozlišujeme tri základné typy aplikácií: [1]

- individuálny aplikačný softvér (IASW)
- typový aplikačný softvér (TASW)
- voľný softvér (open-source /OSS)

V prípade **IASW** sa príslušná aplikácia vytvára na mieru presne podľa potrieb a požiadaviek daného podniku, nainštaluje sa na technologickú platformu využívanú v podniku a následne sa prevádzkuje (obyčajne aj s poskytovaním dodatočného servisu vývojárskou spoločnosťou v forme aktualizácií a opráv). Vývoju takejto aplikácie predchádza dôkladná analýza podnikových procesov a funkcií, tak aby aplikácia poskytovala všetku potrebnú funkcionalitu (Životný cyklus softvéru si ešte rozoberieme ďalej v tejto práci). Výhoda IASW je, že rozsahom a hĺbkou spracovania poskytuje funkcionalitu, ktorú môže jej výrobca prispôsobiť presným potrebám podniku, jeho zámerom, obchodným stratégiám, procesom a funkciám. Nakoľko podnik, pre ktorý je takáto aplikácia vytvorená je vytvorená špeciálne pre daný podnik, ktorý je jediný používateľ danej aplikácie môže mu toto riešenie priniesť značnú konkurenčnú výhodu. Hlavnou nevýhodou je, že IASW je podstatne drahší a vývoj je časovo

náročnejší ako zaobstaranie aplikácie formou TASW. Z porovnania výhod a nevýhod vyplýva, že IASW nie je vhodnou voľbou pre vysoko štandardizované oblasti ako sú napríklad e-mail, účtovníctvo, mzdové účtovníctvo a podobne. Hodí sa skôr pre funkcionalitu, ktoré je individuálna. [1]

V prípade **TASW** je aplikácia vytvorená a ďalej rozvíjaná špecializovaným výrobcom, ktorý sa zoberá zovšeobecňovaním požiadaviek určitej triedy cieľových používateľov (podnikov). Nejedná sa teda o aplikácie, ktoré by boli vytvorené na mieru konkrétnemu podniku, ale o aplikácie, ktoré sú určené vo všeobecnosti pre podniky patriace do určitej oblasti a teda pre určitý typ podnikov. Z pohľadu softvérovej firmy, ktorá vyvíja aplikáciu, bývajú celkové náklady na vývoj TASW zvyčajne podstatne vyššie ako náklady na vývoj IASW. Náklady pre licencie na používanie TASW však pre podniky bývajú omnoho nižšie oproti nákladom na IASW. Pre používateľa TASW sú taktiež nižšie aj časové nároky nakoľko je aplikácia už hotová a po miernom doladení a integrovaní s ostatnými podnikovými aplikáciami môže byť aplikácia zavedená do prevádzky. Výhodou je aj fakt, že výrobcovia TASW budujú aplikácie na základe „best practices“, teda praxou overené postupy a skúsenosti úspešných firiem. Ak teda podnik prispôsobí svoje podnikové procesy TASW, môže naplno využívať osvedčené praktiky lídrov na trhu v danej oblasti. Reorganizácia interných podnikových procesov na obraz best practices využitých v TASW však môže byť značne zložitá a zdĺhavá. Ďalším problémom pri TASW je aj to, že spravidla obsahuje množstvo funkcionality, ktorú príslušný podnik nevyužije. Z analýzy výhod a nevýhod vyplýva, že toto riešenie je vhodné najmä pre podporu vysoko štandardizovaných procesov. [1]

TASW sa zvyčajne distribuuje od výrobcu k zákazníkovi prostredníctvom kanálu tvoreného distribútormi, dealermi a systémovými integrátormi. Ešte pred uvedením softvéru do prevádzky sa však zvyčajne realizuje: [1]

- **lokalizácia** – softvér a jeho funkcionalita sa upravuje podľa platnej legislatívy, národného jazyka a kultúrnych tradícií v príslušnej geografickej lokalite.
- **customizácia –** po dokončení lokalizácie dochádza k ďalším úpravám softvérového produktu podľa špecifických požiadaviek zákazníka. Hlavným cieľom customizácie je prispôsobenie TASW procesom a potrebám konkrétneho podniku. Dochádza napr. k úprave dátových štruktúr (aby boli ukladané správne údaje v správnom formáte tak,

ako to podnik potrebuje), k úprave vzhľadu obrazoviek, presného tvaru vstupných zostáv, formulárov a podobne.

- **systémová integrácia –** pri systémovej integrácií ide o proces prepojenia aplikácie typu TASW s ostatnými aplikáciami, ktorá sú súčasťou informačného systému v danom podniku, z dôvodu zabezpečenia prístupu ku spoločným údajom (pripadne údaje predstavujúce výstup z jednej aplikácie sú vstupom do inej) a vzájomného volania funkcionality (jedna aplikácia volá funkcionalitu inej aplikácie).
- **doprogramovanie softvéru –** ide o zmeny zdrojového kódu aplikácie podľa individuálnych potrieb podniku. Rozdiel medzi doprogramovaním a customizáciou spočíva v tom, že pri customizácii nedochádza k zmene zdrojového kódu aplikácie. Ta už totiž vopred ponúka viaceré voliteľné možnosti rozličných nastavení, z ktorých si zákazník môže vybrať (za pomoci nastavení parametrov aplikácie) a teda nie je potrebné meniť zdrojový kód.

Po uvedení systému do prevádzky by aplikácia na báze TASW mala podporovať aj možnosti *personalizácie*. Personalizácia je vlastnosť TASW, ktorá umožňuje individuálnym používateľom vykonávať menšie úpravy vzhľadu a zmeny správania používateľského rozhrania aplikácie. Napríklad zmenu používateľského jazyka, zmenu spôsobu zobrazenia údajov, zmenu rozmiestnenia grafických prvkov a podobne. Tieto zmeny sú platné a viditeľné iba pre daného užívateľa. Personalizácia sa tak isto ako customizácia zakladá iba na zmenách dovolených v rámci aplikácie a nie je nutné meniť zdrojový kód aplikácie. Personalizácia sa môže v niektorých prípadoch vykonávať aj automatizovane na základe analýzy a histórie správania konkrétneho užívateľa. Najznámejším príkladom je napríklad jav, kde sa používateľovi po čase automaticky začnú v popredí objavovať ikony, ktoré využíva najčastejšie. [1]

Pre vývojársku firmu vyvíjajúcu TASW je veľmi dôležité vedieť, aké zmeny ich zákazníci požadujú, aby mohla zapracovať prípadné zlepšenia do novších verzii svojich produktov. Systémový integrátori preto tvorcovi zmeny vykonané u jednotlivých zákazníkov nahlasujú. Systémový integrátor môže byť aj externá firma nezávislá od tvorcu informačného systému, prípadne sa o systémovou integráciou môže zaoberať priamo IT oddelenie podniku, ktorý aplikáciu používa. V týchto prípadoch tvorcovia TASW aplikácií využívajú rôzne súťaže v progresívnom využití aplikácie prípadne majú vlastné portály na komunikáciu zo zákazníkmi,

ktorý môžu požiadať o pridanie alebo upravenie funkcionality. Na základe popularity potom návrhy s najväčšou podporou komunity používateľov bývajú zapracované do nových verzií. [1]

Pokiaľ podnik pokrýva istú funkcionalitu informačného systému za pomoci TASW namiesto IASW, potom môžeme identifikovať nasledujúce odlišnosti: [1]

- ako sme spomínali, TASW je vybudovaný na základe best practices a teda všeobecne akceptovaných predstáv o fungovaní podnikových procesov. Ak má však podnik svoje vlastné špecifické procesy, potom bude TASW s veľkou pravdepodobnosťou v rozpore s množstvom vnútorných smerníc a noriem v tomto podniku
- TASW vznikajú na základe zovšeobecnenia požiadaviek veľkého počtu zákazníkov v určitom odvetví a preto netreba požiadavky na aplikáciu formulovať tak podrobne ako pri IASW
- V prípade, že je v informačnom systéme podniku viacero TASW aplikácii, hrozí nebezpečenstvo duplicitnej funkcionality, pretože funkcionalita rozličných TASW aplikácií sa môže čiastočne prekrývať
- TASW je zvyčajne prevádzkovateľný na viacerých platformách, kým IASW je zvyčajne spravidla vyvíjaný pre jednu konkrétnu platformu používanú v podniku, pre ktorý sa aplikácia vyvíja. Ak by sa podnik po určitom čase rozhodol platformu zmeniť, predstavovalo by to značné komplikácie, prípadne úplné prepracovanie aplikácie.
- TASW nie je potrebné testovať do takej miery ako IASW nakoľko ide o preverenú funkcionalitu

Open-Source softvér je špecifický typ TASW, od ktorého sa odlišuje nasledovne: [1]

- Open-source zvyčajne nevyvíja komerčná firma s cieľom zárobku ale softvérová komunita dobrovoľníkov, ktorí nemajú nárok na odmenu. Zvyčajne si výšku platby za produkt môže určiť používateľ sám.
- Zdrojové kódy k aplikáciám sú voľne dostupné a každý, kto má záujem si ich môže stiahnuť a prispôsobiť podľa seba
- Vo väčšine prípadov je síce zdrojový kód k dispozícii zdarma, no nesmie sa predávať tretím stranám. Závisí to však od podmienok v licenčnej zmluve
- Nakoľko tvorcom open-source softvéru nie je väčšinou žiadna konkrétna firma, nebýva v dostatočnej miere zaručená kvalita aplikácie, nemusia byť opravované jej chyby

a výrobcovia nenesú žiadnu zodpovednosť za prípadné škody, ktoré môže používanie aplikácie spôsobiť.

Výhodou open-source softvéru sú nulové, prípadne dobrovoľné náklady na nákup licencie. Nevýhodou na druhú stranu je fakt, že aplikácie tohto typu nemajú tak prepracovanú funkcionalitu ako TASW či IASW. Ďalšou nevýhodou je fakt, že tieto aplikácie nebývajú do takej miery parametrizovateľné ako TASW a teda sa nepodporujú takú širokú customizáciu a personalizáciu. Väčšina úprav sa teda musí riešiť priamo zmenami zdrojového kódu. Napriek tomu obľúbenosť open-source softvérov stúpa. [1]

Pri návrhu aplikačnej architektúry informačného systému treba dôsledne analyzovať podnikové funkcie a procesy a identifikovať všetky oblasti, ktoré má tento systém podporovať. Potom je potrebné začleniť do tohto systému také aplikácie, ktoré budú podporovať všetku požadovanú funkcionalitu a zároveň budú poskytovať len minimum funkcionality navyše nakoľko nemá zmysel platiť za niečo, čo podnik nevyužije. Jednou z chýb pri vytváraní aplikačnej architektúry môže byť zaobstaranie aplikácií, ktorých funkcionalita sa navzájom prekrýva. Z pohľadu podniku nemá zmysel platiť za tú istú funkcionalitu viackrát. Navyše sa môžu stávať nezrovnalosti pri používaní tej ktorej aplikácie. Duplicitnej funkcionalite môže predísť dôkladná analýza aplikácií dostupných na trhu (TASW, OSS), prípadne tvorba vlastnej IASW aplikácie. Pri zostavení informačného systému výlučne z aplikácii typu TASW sa duplicite pravdepodobne vyhnúť nedá. Pri návrhu informačného systému je cieľom vyskladať ho možno z čo najmenšieho počtu čo najkvalitnejších aplikácií. Veľký počet aplikácií totižto navyšuje náklady a komplikuje systémovú integráciu ako aj správu a údržbu systému ako celku. Z hľadiska požiadaviek na systémovú integráciu sú dôležite najmä tieto aspekty: [1]

- Ukladanie údajov systematicky tak, aby sa predišlo duplicite dátových objektov v databáze a to aj v prípade, ak s nimi pracujú viaceré aplikácie. Ak by sa stalo, že napríklad údaje o zákazníkoch by boli uložené viackrát, mohlo by dôjsť k tomu, že by sa tieto údaje od seba odlišovali. Potom by sa museli riešiť problémy so zisťovaním, ktorý z údajov je správny.
- Aplikácie by nemali sprístupňovať svoju funkcionalitu len prostredníctvom používateľského rozhrania ale aj prostredníctvom aplikačného programového rozhrania

(Application Program interface – API). Vďaka tomu ich budú môcť využívať nielen koncoví používatelia systému ale aj iné aplikácie.

- Používateľské rozhrania aplikácií by mali byť navzájom podobné tak, aby to používateľovi čo najviac uľahčovalo prácu.

Na základe dnešných trendov, sa z viacerých dôvodov sa typovo aplikačný softvér častokrát vyvíja ako webová aplikácia prístupná cez prehliadač. Takéto aplikácie sú potom prístupné v rámci podniku (alebo iného podniku) za využitia intranetu (extranetu).

Intranet je množina internetových technológií v rámci lokálnej počítačovej siete (LAN) patriacej určitému subjektu (napríklad firma, banka, škola a podobne), ktorá je prístupná len z pracovných staníc patriacich do tejto siete. Intranet je postavený na využívaní približne rovnakých technológií ako internet, pracuje na princípe klient-server architektúry a od vonkajšieho sveta je chránený prostredníctvom kombinácie hardvérových a softvérových prostriedkov. Intranet je pod správou jedného konkrétneho subjektu, ktorý ho vlastní a je využívaný členmi tohto subjektu. Preto treba všetkým užívateľom, ktorý majú do intranetu prístup zodpovedajúce prístupové práva a následne dôsledne overovať ich identitu pri každom pokuse o prístup do systému. Proces overenia identity používateľa do určitého systému sa označuje pojmom *autentifikácia*. Proces stanovenia prístupových práv pre jednotlivých používateľov alebo skupiny používateľov na základe ich spoločných charakteristík sa nazýva *autorizácia.* [2]

Intranet teda predstavuje využitie internetových technológii na vybudovanie vnútropodnikového informačného systému podniku. V podnikovom prostredí zvyčajne obsahuje najmä tieto základné informácie: [2]

- Aktuálne informácie o podniku
- Informácie o cieľoch a stratégií podniku
- Informácie o riadení a nadväznosti podnikových pracovných postupov (workflow)
- Informácie o aktuálnych cenách, zásobách, doprave a podobne
- Informácie o zamestnancoch, pracovných pozíciách, voľných pracovných pozíciách a organizačnej štruktúre podniku
- Informácie o zamestnaneckých benefitoch, teambuildinogoch a podobne
- Záznamy vnútropodnikovej komunikácie vykonanej pomocou intranetu

- Podniková báza znalostí a postupov
- Pokyny, usmernenia pre zamestnancov

Intranet okrem vyššie uvedených informácií môže poskytovať aj jednotný prístupový bod pre autentifikovaný a autorizovaný prístup k rozličným podnikovým aplikáciám (napríklad CRM, ERP, SCM a podobne). Samozrejme v takomto prípade je potrebná integrácia. Používateľ sa tak nepotrebuje prihlasovať do každej aplikácie samostatne, pretože postačuje jeho prihlásenie do intranetu. [2]

Extranet môžeme chápať v dvoch významoch: [2]

1. Rozšírenie intranetu, ktorý patrí konkrétnemu subjektu, tak, aby do neho mali prístup a iné externé subjekty. Veľmi často ide o rozšírenie intranetu určitej firmy tak, aby doňho mohli pristupovať aj obchodný partneri – dodávatelia, zákazníci, distribútori a podobne. Extranet naďalej patrí iba jednému subjektu, rozdiel je teda iba ten, že do extranetu majú prístup viaceré subjekty. Toto chápanie extranetu spomíname s obsahom tejto práce.
2. Prepojenie niekoľkých intranetov prostredníctvom bezpečných komunikačných kanálov napríklad technológiou virtuálnych privátnych sietí. V tomto prípade nie je vlastníkom intranetu jeden subjekt. Každý z týchto subjektov vlastní jeden svoj intranet a prepojenia medzi nimi sa vybudujú na základe interných dohôd a vzájomnej spolupráce

1.2 Integrácia podnikových aplikácií

Integrácia aplikácií je proces, ktorý umožňuje jednotlivým aplikáciám - každá je navrhnutá pre svoj vlastný konkrétny účel - pracovať navzájom. Zlúčením a optimalizáciou údajov a pracovných tokov medzi viacerými softvérovými aplikáciami môžu organizácie dosiahnuť integrácie, ktoré modernizujú ich infraštruktúry a podporujú agilné obchodné operácie. Integrácia aplikácií pomáha preklenúť priepasť medzi existujúcimi miestnymi systémami a rýchlo sa rozvíjajúcimi cloudovými podnikovými aplikáciami. Prostredníctvom hladko prepojených procesov a výmeny údajov umožňuje integrácia aplikácií podnikom organizovať rôzne funkcie v rámci celej ich infraštruktúry, čo podnikom umožňuje efektívnejšie a efektívnejšie fungovanie. [3]

Pri uvažovaní o integrácií podnikových aplikácií je potrebné porozumieť častiam a obsahu podnikových procesov a dát. Taktiež je potrebné poznať ako sú jednotlivé procesy automatizované (prípadne neautomatizované) ako aj dôležitosť jednotlivých podnikových procesov. V závislosti od podniku môžu tieto procesy vyžadovať množstvo času a energie. Veľa podnikov vyhľadáva nové metódy na podporu procesov a sleduje takzvané best pracitices. V skratke, podnik mus rozumieť jak jednotlivým procesom tak aj dátam. Podnik sa teda musí rozhodnúť ktoré procesy a dáta potrebuje integrovať. Tento proces rozhodovania môžeme rozdeliť do niekoľkých časti: [4]

- Dátový level
- Level aplikačného rozhrania
- Level používateľského rozhrania
- Metodický level

Dátový level je proces, technika a technológia presúvania dát medzi dátovými úložiskami. Toto môže byť opísané ako extrahovanie dát z jednej databázy a následné pripísanie dát do druhej databázy, pričom môžeme dáta nejakým spôsobom upraviť. Výhoda dátového levelu integrácie podnikových aplikácií je cena využitia tohto prístupu. Nakoľko nezasahujeme do samotnej aplikácie a nemeníme zdrojový kód šetríme náklady na vývoj, testovanie a nasádzanie aplikácie. Navyše, technológie na prenos dát medzi databázami ako aj preformátovanie informácií je pomerne finančne nenáročné.

Level aplikačného rozhrania integrácie podnikových aplikácií odkazuje na využitie rozhraní poskytnutom iným aplikáciám. Vývojári využívajú tieto rozhrania pre prístup k podnikovým procesom ako aj informáciám. Využitím týchto rozhraní vývojári vedia spojiť viaceré aplikácie dohromady tak, aby vedeli zdieľať informácie aj podnikovú logiku. Jediné obmedzenia pre vývojárov sú možnosti a funkcie, ktoré poskytuje aplikačné rozhranie. Tento druh integrácie je veľmi populárny pri moderných a robustných aplikáciách ako napríklad SAP. Pre integráciu takýchto systémov, musíme využiť tieto rozhrania na prístup k procesom a dátam, extrahovať informáciu, preložiť ju do formátu pochopiteľného pre cieľovú aplikáciu a túto informáciu uložiť.

Metodický model integrácie podnikových aplikácií je zdieľanie podnikovej logiky, ktorá existuje naprieč podnikom. Napríklad, metóda aktualizácie záznamu o zákazníkovi môže byť

prístupná z rôzneho počtu aplikácií a aplikácie môžu pristupovať k metódam inej aplikácie. Mechanizmus zdieľania metód medzi aplikáciami v rátane distribuovaných objektov, aplikačných serverov a nástrojov umožňuje jednoduché vytvorenie novej aplikácie, ktorá je kombináciou dvoch a viacerých iných aplikácií.

Level používateľského rozhrania integrácie podnikových aplikácií je veľmi jednoduchý a užitočný. Aplikácie môžu byť zoskupené a ich používateľské rozhrania použité ako spoločný bod integrácie – tento prístup je známy ako integrácia používateľského rozhrania na báze proxy. Existuje aj iný prístup, ktorý je založený na skriptovaní. Niektorý ľudia považujú tento prístup za nestabilný a hoci to nie je preferované, integrácia na úrovni používateľského rozhrania by sa mala používať v prípadoch, keď nemôžeme ľahko alebo priamo získať prístup k databáze alebo keď je obchodná logika vložená priamo do používateľského rozhrania. Napríklad ak aplikácia neposkytuje dátové úložisko ani verejné API. Mnoho aplikácií typu klient/server vkladá obchodnú logiku do klienta. Prístup k údajom a ich zachovanie v takýchto prípadoch je možné dosiahnuť iba integráciou na úrovni používateľského rozhrania. [4]

Okrem definovania vzájomne zrozumiteľného jazyka, s ktorým môžu aplikácie komunikovať sa využívajú aj integrácie aplikácií aj kombináciou dvoch špecifických typov správ na prenos a príjem relevantných informácií medzi zdrojmi údajov a aplikáciami: [5]

- Synchrónna výmena
- Asynchrónna výmena

Synchrónna výmena sa využíva tak, že aplikácia odošle požiadavku inej aplikácií a nemôže pokračovať v ďalších funkciách, kým nedostane odpoveď. Pripojenie vytvorené touto metódou výmeny zostáva otvorené po dobu výmeny údajov a spolieha sa na middleware, aby sa predišlo zbytočným oneskoreniam v dôsledku chýb a časových limitov. Napríklad webový server poskytujúci obsah stránky nakupujúcim online pomocou prehľadávača v reálnom čase je synchrónna výmena používajúca protokol HTTP (Hypertext Transfer Protocol). Ďalším príkladom by mohlo byť opakovanie zákazníckeho servisu kliknutím na „volanie“ pre konkrétny kontakt v rámci systému riadenia zdrojov zákazníka (CRM) a kontaktné informácie zákazníka, ktoré sa doručujú komunikačnej sade na automatické vytáčanie. Synchrónna výmena je ideálna na sekvenčné spracovanie, ale jej závislosť od potvrdených odpovedí môže spomaliť

pracovný tok, ak je middleware na zadnom konci neoptimálny a vyžaduje častý časový limit alebo zlyhá z dôvodu nedostatočnej odpovede.

Asynchrónna výmena sa nespolieha na potvrdenú odpoveď, aby mohla pokračovať v činnosti. Spojenia sa ukončia, akonáhle sa pošlú požiadavky, a operácie je možné vykonávať paralelne a v akomkoľvek poradí. Správa API je robustnejšia a vo výsledku je táto metóda škálovateľnejšia, rýchlejšia a všestrannejšia ako synchrónna výmena. Metódy integrácie asynchrónnej výmeny používajú niekoľko rôznych typov správ:

- *Spätné volanie* je asynchrónna variácia požiadavky na odpoveď. Aplikácia odosielajúca žiadosti o informácie takpovediac „nedrží líniu; namiesto toho nastaví konkrétnu akciu nazývanú spätné volanie, ktorá spracuje každú odoslanú odpoveď a podľa potreby pokračuje v ďalších operáciách. To urýchľuje a zefektívňuje automatizované úlohy, ako je napríklad jedna aplikácia, ktorá získava informácie z viacerých ďalších aplikácií, zhromažďuje a analyzuje prijaté informácie (triedi ich podľa závislostí) a po prijatí všetkých požadovaných informácií generuje správu.
- *„Fire and Forget“* je jednosmerná metóda asynchrónnej výmeny. Žiadajúca aplikácia odošle jednu (alebo viac pravdepodobných) požiadaviek do ďalších aplikácií a pokračuje bez prijatia potvrdenia. Ak napríklad predajný personál používa CRM riešenie na zadanie nových informácií o zákazníkovi, môže CRM automaticky aktualizovať ERP systém o nové kontaktné údaje a naopak.
- *Smerovanie správ* je variácia Fire and Forget, ktorá integruje middleware na koordináciu požiadaviek z viacerých aplikácií poskytovaných jednej prijímajúcej aplikácii na spracovanie. Pravidlá smerovania a rozdelenia sa vytvárajú tak, aby sa zabezpečilo, že údaje sa spracúvajú v správnom poradí, aby sa dokončil ďalší krok v každom procese (napr. Generovanie správ alebo vytváranie faktúr pre zákazníkov pomocou informácií získaných z viacerých odlišných systémov).
- *Publikovanie a prihlásenie na odber* je inverzia metódy smerovania správ. Prijímajúce aplikácie vytvárajú pravidlá pre typy informácií, ktoré chcú dostávať a podľa potreby využíva middleware na získanie potrebných informácií z rôznych zdrojov. Napríklad korporácia využívajúca centralizované riešenie ERP by mohla tento systém napájať z CRM, fakturačného softvéru, finančných nástrojov a súborov na analýzu veľkých dát

atď. Táto metóda umožňuje, aby rôzne aplikácie slúžili ako odosielatelia aj prijímatelia, čím sa vytvára zložitá, ale výkonná sieť vzájomne prepojených aplikácií poskytujúcich údaje do jedného používateľského rozhrania s najvyššou možnou kvalitou a rýchlosťou údajov.

1.3 Životný cyklus softvéru

Životný cyklus vývoja softvéru dodržiava dôležité fázy, ktoré sú nevyhnutné pre vývojárov, ako napríklad plánovanie, analýza, návrh a implementácia. Pre vývoj životného cyklu softvéru bolo vyvinutých mnoho modelov ako napríklad:

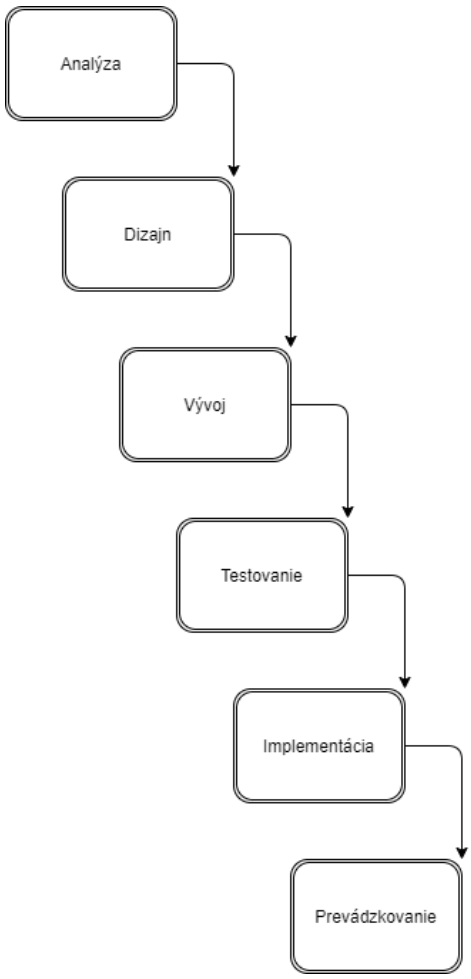
- Vodopádový
- Špirála
- V-model
- Rýchle prototypy
- Prírastkové/iteračné

Vodopádový model je sekvenčný model. Najstarší a najznámejší model, ktorý zobrazuje postupnosť etáp, v ktorých výstup každého stupňa sa stáva vstupom pre ďalší. V-Model demonštruje vzťahy medzi každou fázou životného cyklu vývoja a s ním spojenú fázu testovania. Agilný vývoj softvéru je skupina softvérovej vývojovej metodiky založenej na iteráciách a postupnom vývoji, kde sa požiadavky a riešenia vyvíjajú prostredníctvom spolupráce medzi samo-organizujúcimi sa a viacfunkčnými tímami. [6]

1.3.1 Vodopádový model

- Vodopádový model predstavuje postupný vývojový model
- Požiadavky by mali byť vždy jasné a ukončené pred prechodom do ďalšej fázy
- Testovanie prebieha až potom, ako je ukončená fáza vývoja/programovania
- Každá fáza vývoja prebieha bez akéhokoľvek prekrývania
- Každá fáza obsahuje harmonogram úloh, ktoré sa musia dokončiť v určitom (naplánovanom) časovom období
- Na konci každej fázy prebehne dokumentácia
- Vo vodopádovom modeli musí byť každý proces zastavený predtým ako sa začne ďalší. Tu však nastáva problém, čo má napríklad robiť testovací tím predtým, než sa dokončí

programovanie. Vďaka tomu sa vo vodopádovom modeli odhaľujú chyby až veľmi neskoro.



Obrázok 1 Vodopádový model životného cyklu softvéru

Požiadavka zadaná klientom by mala byť jasná predtým začíname ďalšiu fázu životného cyklu vývoja, pretože v modeli vodopádu by mala byť fáza požiadavky ukončená pred začatím fázy návrhu. Ďalšie zmeny v požiadavkách nebudú zohľadnené. [6]

Výhody

- Požiadavka je jasná pred začiatkom vývoja.
- Každá fáza je dokončená v stanovenom časovom období potom prejde do ďalšej fázy.
- Ako jeho lineárny model sa dá ľahko implementovať.

- Množstvo zdrojov potrebných na realizáciu tohto cieľa model sú minimálne.
- Každá fáza sa riadi náležitou dokumentáciou kvalita rozvoja.

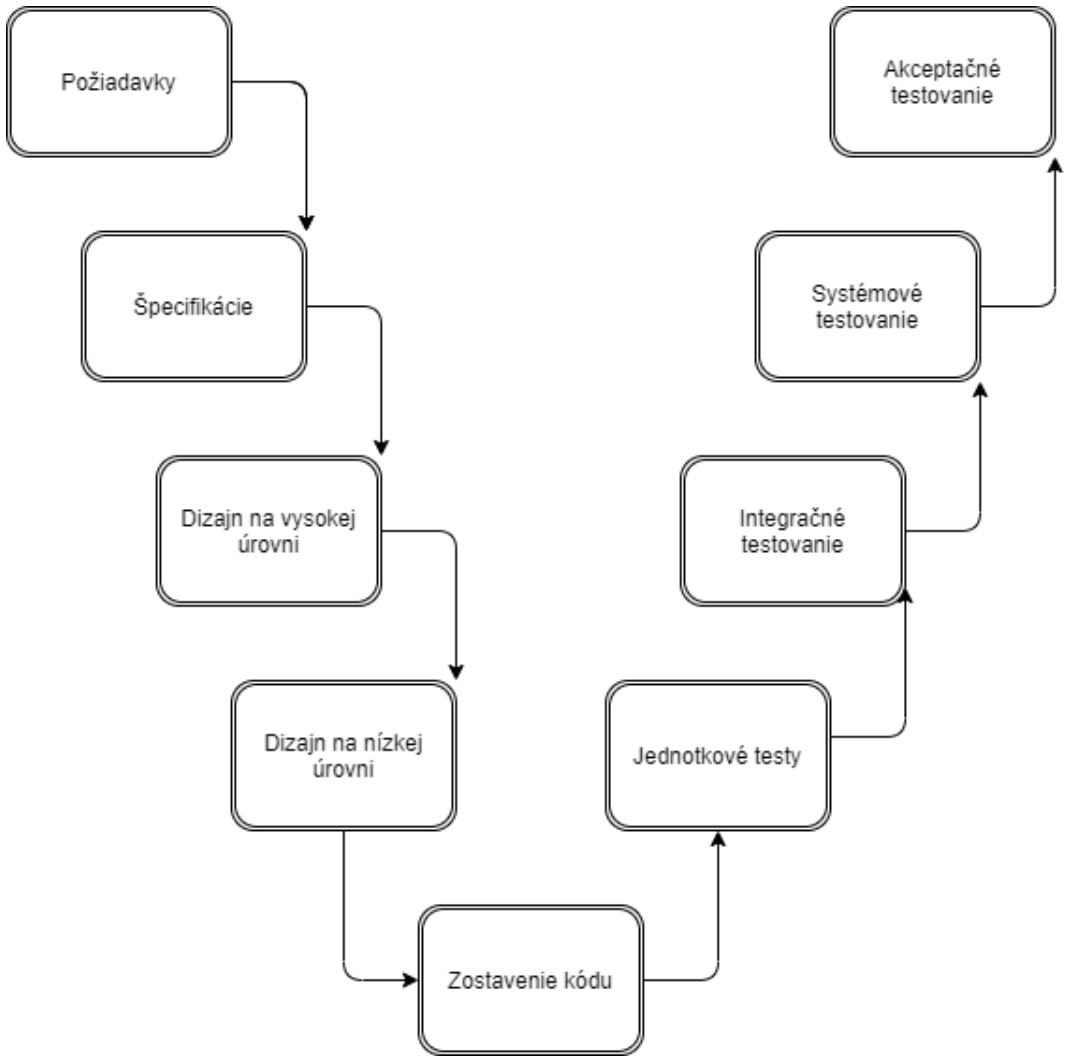
Nevýhody

- Problémy s jednou fázou sa nikdy nevyriešia počas tejto fázy a vlastne mnoho problémov týkajúcich sa konkrétnej fázy nastanú až je fáza ukončená, výsledkom je zle štruktúrovaný systém.
- Ak klient chce, aby sa požiadavka zmenila, nebude implementovaná v súčasnom procese vývoja

Napriek mínusom je tu veľa kladov, ktoré zabezpečia že zostane jedným z najpopulárnejších modelov používaných v oblasti vývoja softvéru.

1.3.2 V-Model

V-Model (model validácie a overovania) je modifikovaná verzia vodopádovej metódy. Na rozdiel on vodpoádovej metódy V-Model nebol navrhnutý na báze lineárnej osi. Namiesto toho fázy postupujú v protismere ihneď po ukončení vývojovej fázy. Tento proces je vyvážený a spolieha na overovanie krokov predtým, ako sa fáza ukončí a posunie sa na ďalšiu. To znamené, že v tomto prístupe programátori a testeri pracujú paralelne. Vo V-Modeli sú požadované systémové testovacie scenáre pripravené a založené ako dokument na vysokej úrovni. Integračné testovacie scenáre sú pripravené a založené ako dokument nízkej úrovne. Hneď ako sa dokončí programovanie začína testovanie a teda v tomto modeli je medzi fázou vývoja a testovania istý vzťah. [6]



Obrázok 2 V-Model životného cyklu softvéru

Výhody

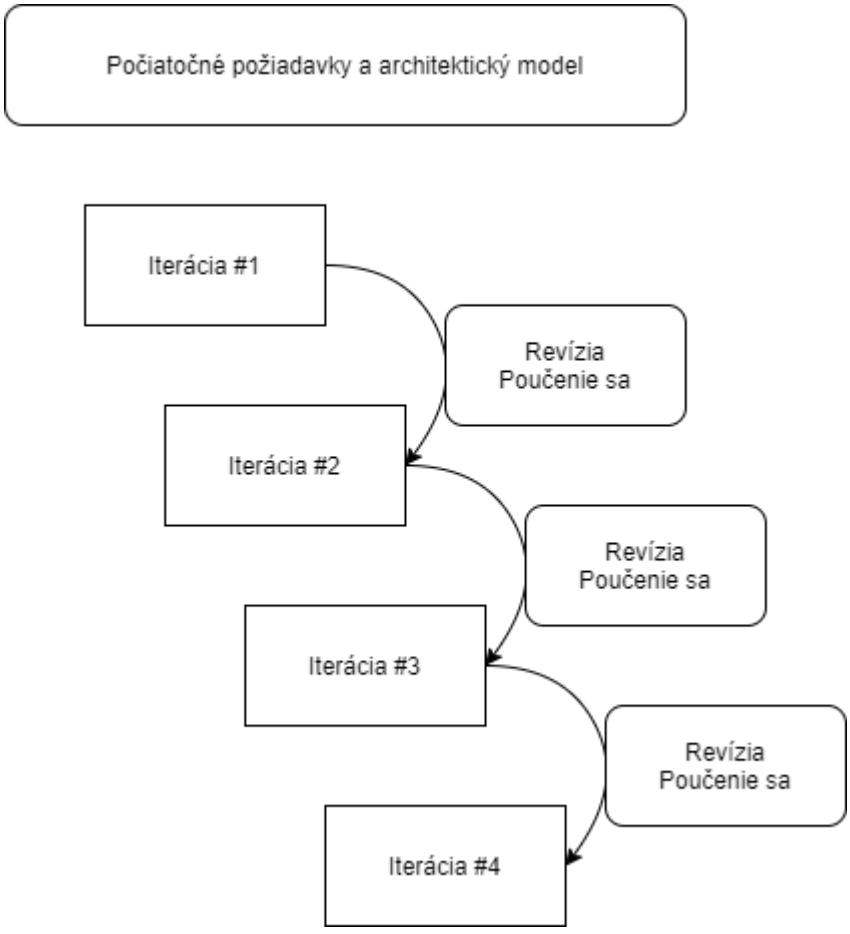
- Rovnaký proces ako pri vodopádovom modeli
- Rola testerov je zapojená v požadovanej fáze
- Požiadavky na zmenu sú možné v každej časti

Nevýhody

- Veľmi prísny a neflexibilný model
- Pri akejkoľvek zmene je potrebné zmeniť nielen dokument so špecifikáciou požiadaviek ale aj testovaciu dokumentáciu
- Nie je vhodný pre krátke projekty nakoľko každá fáza vyžaduje prehodnocovanie

1.3.3 Agilný model

- Názov agilný odráža vlastnosť tohto modelu a teda obratný, rýchly, flexibilný
- Agilné metódy majú prispôsobivé tímy, ktoré sú schopné reagovať na zmeny požiadaviek
- Poskytuje spokojnosť zákazníka pri rýchlom dodávaní najužitočnejších funkcionalít
- Zmeny požiadaviek sú vítané
- Fungujúci softvér je dodávaný pravidelne behom iterácií
- Najdôležitejšia vlastnosť je rýchle a pravidelné dodávanie menších častí softvéru pre uspokojenie potrieb zákazníka



Obrázok 3 Model Agilnej metódy životného cyklu softvéru

Výhody

- Schopnosť reagovať na nové a zmenené požiadavky
- Jasná komunikácia medzi zákazníkom a dodávateľom o tom, čo má byť spravené

Nevýhody

- Ak sú veľkých projektoch je ťažké odhadnúť celkový objem práce a koľko času bude celkové vyhotovenie trvať
- Táto metóda viac vyhovuje skúsenejším programátorom a je náročnejšie do tohto procesu pritiahnuť juniorov bez toho, aby ich starší kolegovia zaúčali [6]

V rámci Agilných metodík poznáme viacero konkrétnych prístupov ako sú Kanban, Scrum, Extrémne programovanie, DSDM. Konkrétne metóda Scrum je vo firme využívaná na projektové riadenie. V súvislosti s Jirou ako nástrojom na riadenie projektov zameraných na Scrum.

1.4 IT Servis manažment

IT servis manažment (ITSM) je integrovaný procesný prístup, ktorý umožňuje IT organizáciu poskytovať servis zameraný na biznis a požiadavky zákazníka. Hlavným zámerom ITSM je zabezpečovanie celého životného cyklu IT služieb. Jeho zámerom zvyčajne nie je projektové riadenie alebo vývoj softvéru a aplikácií. ITSM procesy by mali byť navrhnuté a implementované tak, aby usmernili a integrovali projektové riadenie a procesy vývoja softvéru a aplikácií. Dôsledné využitie správne navrhnutých a implementovaných procesov umožňuje IT podniku: [7]

- Usmerniť vynaložené úsilie s biznis cieľmi
- Zabezpečiť súlad s príslušnými regulačnými kontrolami
- Dosiahnuť spokojnosť zamestnancov aj zákazníkov

Efektívne procesy umožnia rovnako zákazníkovi ako aj IT oddeleniu vedieť, čo musí byť spravené a ako to má byť spravené. V tejto práci sa nebudeme venovať konkrétnym procesom, ale nástrojom a integrácií týchto nástrojov, za účelom nastavenia týchto procesov čo najefektívnejšie. Preto je namieste najprv si vysvetliť dôležitosť zavedenia a optimalizáciu týchto procesov. Pre efektívne zostrojenie týchto procesov sú esenciálne nasledujúce komponenty: [7]

- **Ľudia** – Jednotlivci a tímy, ktoré podporujú zákazníka vykonávaním procesov
- **Procesy –** vzájomne prepojené pracovné aktivity, ktoré na základe vstupu vyprodukujú požadovaný výstup predstavujúci požiadavky zákazníkov

- **Technológia-** Nástroje a technológie, ktoré ľudia využívajú pri práci
- **Informácie –** dáta a informácie, ktoré ľudia potrebujú na vykonanie úloh, meranie efektivity procesov a identifikovanie možností na zlepšenie

V dnešnej dobe rýchlo vyvíjajúceho sa biznisu je každý pracovník zodpovedný za určitý proces aspoň na nejakej úrovni. Tak ako sa vyvíjajú nové informačné technológie, vyvíjajú sa aj požiadavky, kde biznis lídri očakávajú, že informačné technológie a riešenia za pomoci informačných technológií pomôžu priniesť primárne biznis ciele ako sú napríklad: [7]

- Zlepšovanie biznis procesov
- Znižovanie nákladov podniku
- Zlepšovanie použitia informácií a analýz pri rozhodovaní
- Zlepšenie efektivity práce
- Prilákanie nových alebo udržanie si zákazníkov
- Vytvorenie nových, moderných produktov a služieb

Tieto procesy teda môžeme rozlíšiť do dvoch základných skupín: [7]

- **Hlavné procesy**, ktoré predstavujú hlavné podnikové kompetencie (finančné služby, výroba, IT služby a podobne) Tieto procesy sa zameriavajú na poskytovanie služieb alebo produktov pre zákazníkov
- **Podporné procesy**, ktoré riadia alebo kontrolujú a podporujú podnikové aktivity. (napríklad účtovníctvo, ľudské zdroje, ITSM a podobne)

Na určenie rozsahu a ujasnenie potrieb ITSM bolo vyhotovených niekoľko frameworkov. Tieto frameworky poskytujú rady a popisujú best practices, ktoré IT podnik môže implementovať a priebežne vylepšovať ich ITSM možnosti. Ako sme už spomínali, best pracitces sú overené postupy ako dosiahnuť takmer optimálne výsledky. Best practices sú overené časom, skúsenosťami a výskumom naprieč mnohými odborníkmi a firmami.

Veľa organizácií si prispôsobilo postupy z viacerých frameworkov v snahe nastavenia procesov na základe ich konkrétnych požiadaviek. Najčastejšie využívané frameworky sú: [7]

- *ITIL* – Information Technology Infrastructure Library
- *COBIT* – Control Objectives for Information and related Technology
- *MOF* – Microsoft Operations Framework

ITIL je zoznam best practices zozbieraných z verejného aj súkromného sektora. Pôvode bol vytvorený v roku osemdesiatych rokoch British Government's Central Computer and Telecommunications Agency dnes známou ako Office od Government Commerce. Posledná verzia ITIL V4 z roku 2018 poskytuje 34 postupov/praktík zaradených v 3 kategóriách: [8]

- **Všeobecné riadenie**
    - riadenie architektúry
    - kontinuálne zlepšovanie
    - riadenie informačnej bezpečnosti
    - riadenie bázy vedomostí
    - meranie a reportovanie
    - riadenie organizačných zmien
    - riadenie portfólia
    - projektové riadenie
    - manažment vzťahov
    - risk manažment
    - finančný manažment
    - strategický manažment
    - dodávateľský manažment
- **Manažment služieb**
    - riadenie dostupnosti
    - biznis analýza
    - riadenie kapacity a výkonu
    - kontrola zmien
    - incident manažment
    - riadenie IT majetku
    - monitoring a event manažment
    - problém manažment
    - riadenie verzii
    - manažment katalógu služieb
    - manažment konfigurácií služieb
    - manažment kontinuity služieb

    - service dizajn
    - service desk
    - SLA
    - manažment požiadaviek pre služby
    - validácia a testovanie služieb
- **Technické riadenie**
    - riadenie nasádzania
    - riadenie infraštruktúry a platformy
    - vývoj softvéru a manažment

COBIT je IT vládny framework a podporná sada nástrojov, ktorá umožňuje manažérom premostiť dieru medzi požiadavkami na kontrolu, technickými problémami a biznis rizikami. Pôvodne bol vytvorený v roku 1992 Information Systems Audit and Control Association a IT Governance Institute. Posledná verzia je COBIT 2019. COBIT umožňuje taktický vývoj a dobré postupy pre IT kontrolu naprieč celou organizáciou. COBIT definuje aktivity naprieč piatimi skupinami: [9]

- **EDM** – ohodnotiť, riadiť a monitorovať
- **APO** – usmerniť, plánovať a organizovať
- **BAI** – vybudovať, osvojiť si a implementovať
- **DSS** – dodať, slúžiť a podporovať
- **MEA** – monitorovať, ohodnotiť a oceniť

ITIL a COBIT sa navzájom dopĺňajú. Napríklad je možné doplniť výhody ITIL IT operatívnych procesov spolu s kritickými faktor úspechu (CSF) a kľúčový ukazovateľ výkonu (KPI) z COBITu. CSF je merateľná vlastnosť, ktorá musí byť viditeľná pre úspešnosť procesu. CFS odzrkadľuje ciele procesu a podporuje biznis ciele. KPI je kľúčová metrika využívaná na riadenie procesu. Metrika merajúca výkonnosť. KPI podporuje kritické faktory úspechu.

MOF sa skladá z best practices, princípov a aktivít, ktoré poskytujú komplexné návody pre dosiahnutie spoľahlivosť IT riešení a služieb. Návod opisuje životný cyklus v 4 fázach: [7]

- **Plán**: plánovanie a optimalizovanie IT služieb pre nastavenie s biznis stratégiou
- **Dodanie**: Dizajn a dodanie IT služby

- **Prevádzkovanie**: prebiehajúca prevádzka a podpora
- **Riadenie**: risk manažment, organizácia tímov, change manažment

1.5 Aktuálna situácia v podniku

Podnik, ktorý je predmetom jednania sa s približne 50 zamestnancami zaraďuje medzi malé podniky. Podnik sa venuje najmä digitálnej transformácií podnikov a firiem za pomoci moderných Microsoft riešení pre zákazníkov vo finančnom sektore, telekomunikáciách, priemysle či štátnej správe. Jednotlivé riešenia, ktoré poskytuje sú najmä portálové riešenia, Power Apps, CRM riešenia, BI reporty a elektronické štátne služby. Túto pomoc poskytuje pomocou konzultačnej činnosti, vývoja na zákazku, integrácií, projektového riadenia, testovania a technickej podpory. [10]

Podľa náplne práce sa štruktúra podniku delí na štyri základné oddelenia:

- Manažment
- Delivery
- Sales
- Support

Na základe pomenovania oddelení môžeme tušiť náplň práce, ktorú jednotlivé oddelenia vykonávajú. Štandardne celý podnik využíva niektoré nástroje naprieč všetkými oddeleniami. Sú to najmä aplikácie spadajúce pod službu Microsoft 365. Microsoft 365 obsahuje balík Office365 MS Outlook, Word, Excel, PowerPoint a podobne), ktorý slúži na prácu s dokumentami. Tieto dokumenty sú pod správou DMS nástroja Sharepoint, ktorý slúži na vytváranie, úpravu, schvaľovanie, publikovanie, zdieľanie či archiváciu firemných dokumentov s prístupovým bodom na jednom mieste. Ďalším dôležitým prvkom je aj cloudové úložisko OneDrive slúžiace na zálohovanie a zdieľanie dokumentov a súborov. Spolu s nástrojmi na komunikáciu, ktoré prepájajú všetky spomínané prvky služby Microsoft365 MS Teams a Exchange serverom tvoria jadro firemného intranetu. Konkrétne MS Teams a Exchange serveru sa budeme venovať podrobnejšie neskôr v tejto práci za účelom oboznámenia sa s jednotlivými funkciami nevyhnutnými na pochopenie procesu integrácie.

Mimo týchto základných aplikácií využívajú jednotlivé oddelenia aj pre nich typické aplikácie podporujúce vykonávanie hlavných procesov. Manažment tím napríklad využíva

nástroje Business intelligence, Sales zas CRM MS Dynamics365 pre zlepšovanie a udržiavanie vzťahu so zákazníkmi alebo CMS pre úpravu a prezentovanie firemného webu. Delivery oddelenie má za úlohu pracovať na projektoch. Za týmto účelom momentálne využíva Azure DevOps Server spolu s nástrojom Git. Po ukončení vývoja produktu sa projekt presúva na Support oddelenie kde prebieha fáza prevádzkovania, konkrétne dodržiavanie SLA. Okrem toho Support oddelenie zastrešuje aj prevádzkovanie interných systémov a pracuje na interných projektoch. Na tieto povinnosti využíva nástroje Jira Software a Jira Service Desk.

1.5.1 Microsoft aplikácie

V tejto časti práce sa bližšie oboznámime s aplikáciami, ktoré budeme neskôr integrovať. Popíšeme si ich funkcionalitu a význam na základe čoho pochopíme aj dôvod integrovania týchto systémov.

1.5.1.1 Microsoft 365/AAD

Ako sme sa už oboznámili s firmou a spomenuli jej základnú aplikačnú štruktúru je zrejmé, že základom je služba Microsoft 365. Microsoft 365 je cloudová služba od spoločnosti Microsoft ktorá zastrešuje prístup k všetkým cloudovým službám spoločnosti Microsoft. Jedna s týchto služieb je aj Azure Active Directory (AAD). AAD je cloudová služba na manažment totožnosti a prístupov, ktorá pomáha zamestnancom prihlásiť sa pristupovať k rozličným zdrojom: [11]

- Externé zdroje akými sú tisíce cloudových aplikácií fungujúcich ako Softvér ako servis v cloudovom prostredí
- Interné zdroje, akými sú napríklad podnikový intranet alebo cloudové aplikácie vyvinuté priamo pre daný podnik.

Služba Azure Active directory je určená pre: [11]

- *IT adminov*. Ako admin je možné pomocou AAD kontrolovať prístupy k jednotlivým aplikáciám na základe potreby. Napríklad pri prihlasovaní sa k dôležitým firemným zdrojom je možne nastaviť požadovanie dvojfaktorovej autentifikácie. Navyše AAD poskytuje silné nástroje, ktoré automaticky chránia totožnosť a poverenie používateľov na splnenie požiadaviek pri správu prístupu.

- *Vývojárov aplikácií*. Ako vývojár je možné využiť štandardizovaný prístup k prihlasovaniu sa do vytvorenej aplikácie. AAD taktiež poskytuje API pre možnosť personalizovať možnosť prihlásenia sa do vytváranej aplikácie.
- *Požívateľov Microsoft 365, Office 365, Azure alebo Dynamics CRM online.* Ako používateľ týchto aplikácií ste automaticky používateľom AAD. V prípade, pridelenia licencie používateľovi je možné mu okamžite nastaviť prístupy do integrovaných cloudových aplikácií.

1.5.1.2 Azure DevOps Server – Azure Boards

Azure DevOps poskytuje integrované služby a nástroje na riadenie softvérových projektov od plánovania, vývoj, testovanie až po nasadenie. Služby sú poskytované pomocou klient/server modelu. Mnoho z nich sú poskytované pomocou webového rozhrania, ktoré je podporované väčšinou prehliadačov. Niektoré služby ako napríklad kontrolu zdrojového kódu, budovanie takzvaných „pipelines“ a monitorovanie priebehu práce sú dostupné aj cez klienta. [12]

Azure DevOps poskytuje následné skupiny služieb: [12]

- *Dashboards* umožňuje zobraziť prehľad o postupe tímu pridaním jedného alebo viacerých widgetov alebo grafov na hlavný panel. Prispôsobiteľné, vysoko konfigurovateľné panely poskytujú tímom flexibilitu pri zdieľaní informácií, sledovaní pokroku a trendov a zlepšovaní procesov pracovného toku. Každý tím môže prispôsobiť svoje informačné panely na zdieľanie informácií a sledovanie svojho pokroku.
- *Wiki* slúži na podporu tímu pracujúcom na projekte uschovaním a zobrazením informácií o projekte. Poskytuje bohaté formátovanie, tabuľky a obrázky.
- *Boards* slúži na riadenie tímov pracujúcich na projekte. Poskytuje širokú funkcionalitu na riadenie projektov s podporou Kanban a Scrum metód. O tejto službe si povieme viac neskôr v tejto kapitole nakoľko túto službu budeme integrovať.
- *Repos* je súbor nástrojov na kontrolu verzií, ktorý umožňuje kontrolu a manažovanie zdrojového kódu. Systémy nariadenie verzií pomáhajú sledovať vykonávané priebežné zmeny v kóde. Repos poskytuje možnosť napojenia dvoch rôznych nástrojov na kontrolovanie verzii: Git a TFVC. O gite si povieme viac v nasledujúcej kapitole.

- *Pipelines* je nástroj na podporovanie nepretržitej integrácie (CI) a nepretržitého dodávania (CD) softvéru. Umožňuje konštantne a konzistentne testovať a budovať zdrojový kód. Pipelines sa formulujú pomocou YAML syntaxe alebo pomocou používateľského rozhrania.
- *Test Plans* poskytuje výkonné testovacie nástroje využívané na zvýšenie kvality a spolupráce počas celého vývoja softvéru. Ľahko použiteľné riešenie pre správu testov založené na prehliadači poskytuje všetky funkcie potrebné na plánovanie manuálneho či akceptačného testovania a na zbieranie spätnej väzby zainteresovaných strán.
- *Artifacts* slúži na manažovanie balíčkov (NuGet, Maven, npm), ktoré sú plne integrovateľné do CI/CD pipelines na jedno kliknutie.

Azure Boards

Azure Boards je cloudová služba/aplikácia slúžiaca na projektové riadenie. Azure Boards na riadenie projektov poskytujú nasledujúce nástroje:

- *Pracovná položka*: jedná sa o takzvaný tiket, ktorý obsahuje informácie o úlohe, ktorú je potrebné vykonať
- *Tabuľa*: prezentuje tikety na základe ich stavu na zobrazenie prehľadného stavu vykonávaných úloh. Zobrazenie je podobné ako tabuľa s lístočkami ktorá sa bežne využíva pri metóde Kanban. Práca s tabuľou je založená na princípe potiahni a pusti.
- *Backlog*: Backlog je zoznam nevykonaných pracovných položiek (úloh). Jedná sa teda o plán práce a úložisko všetkých nevyriešených pracovných položiek. Využíva sa na plánovanie, určovanie priorít a organizáciu práce.
- *Šprinty*: Zobrazenie šprintov poskytuje filtrovaný zoznam nevyriešených pracovných položiek, ktoré tím pridelil ku konkrétnej iterácií projektu.
- *Dotazy*: dotazy sú filtrované zoznamy pracovných položiek založených na kritériách zadaných v editore dotazov.

V podniku je prevádzkovaná on-premise verzia Azure DevOps server. [13]

1.5.1.3 Git

Správa verzií je nástroj, ktorý zaznamenáva zmeny do súboru alebo sady súborov v priebehu času, aby ste si mohli neskôr spomenúť na konkrétne verzie. Pre príklady v tejto práci používame zdrojový kód softvéru ako súbory riadené verziou, aj keď v skutočnosti to môžeme urobiť s takmer akýmkoľvek typom súboru v počítači. Ak sme boli grafický alebo webový dizajnéri a chceli by sme zachovať každú verziu obrázka alebo rozloženia, je veľmi rozumné používať systém riadenia verzií (VCS). Umožňuje nám vrátiť súbory do predchádzajúceho stavu, vrátiť celý projekt do predchádzajúceho stavu, porovnať zmeny v čase, zistiť, kto naposledy upravil niečo čo môže spôsobovať problém a podobne. Používanie VCS tiež všeobecne znamená, že ak niečo pokazíme alebo stratíme súbory, môžeme ich ľahko obnoviť. Toto všetko navyše získame za veľmi malú réžiu.

Rovnako ako v prípade mnohých skvelých vecí v živote, aj Git začal s trochou tvorivej deštrukcie a ohnivých kontroverzií. Linuxové jadro je open source softvérový projekt s pomerne veľkým rozsahom. Po väčšinu života systému Linux (údržba jadra 1991 - 2002) boli zmeny v softvéri zaznamenané ako opravy a archivované súbory. V roku 2002 projekt jadra Linuxu začal používať patentovaný DVCS s názvom BitKeeper. V roku 2005 bol vzťah medzi komunitou, ktorá vyvinula jadro Linuxu, a komerčnou spoločnosťou, ktorá vyvinula BitKeeper ukončený a bezplatný stav nástroja bol odvolaný. Toto podnietilo vývojovú komunitu Linuxu (a najmä Linus Torvalds, tvorca Linuxu) vyvinúť vlastný nástroj založený na niektorých postupoch, ktoré sa naučili pri používaní nástroja BitKeeper. Niektoré z cieľov nového systému boli:

- Rýchlosť
- Jednoduchý dizajn
- Silná podpora pre nelineárny rozvoj (tisíce paralelných vetiev)
- Plne distribuovaný
- Schopný efektívne zvládať veľké projekty, ako je jadro Linuxu (rýchlosť a veľkosť dát)

Od svojho zrodu v roku 2005 sa Git vyvinul a dozrel tak, aby sa dal ľahko používať, a napriek tomu si tieto počiatočné vlastnosti zachoval. Je neuveriteľne rýchly, je veľmi efektívny pri veľkých projektoch a má skvelý systém rozvetvenia pre nelineárny vývoj systémov. [14]

1.5.1.4 MS Teams

Microsoft Teams je digitálne centrum cloudových aplikácií, ktoré prináša konverzácie, stretnutia, zdieľanie súborov a aplikácie spolu na jednom mieste. Chatovacie aplikácie v reálnom čase, ktoré fungujú na viacerých operačných systémoch a zariadeniach sú dnes všadeprítomné. Niektoré chatovacie aplikácie, ako napríklad Teams, ponúkajú funkcie, ktoré klasický e-mail nemôže. Sú to napríklad chatovacie miestnosti, videokonferencie a funkcie, ktoré kopírujú populárne sociálne siete. Do konca roku 2020 sa očakáva, že 41 percent organizácií bude globálne používať Microsoft Teams, pričom využitie nájde v rôznych organizáciách vrátane malých a stredných podnikov z rôznych priemyselných odvetví vrátane výroby, zdravotníctva, neziskových organizácií, školstva, vlády a financií. [15]

Základnou štruktúrou MS Teams sú rovnomenné tímy. Každý takýto založený tím potom obsahuje skupinu členov, vlákna a rôzne pridané aplikácie. MS teams sú štandardne postavené na platforme Microsoft 365, čiže pri zdieľaní dokumentov využívajú MDS Sharepoint, využívajú skupiny vytvorené, autentifikáciu a autorizáciu v rámci AAD.

1.5.1.5 MS Outlook a Exchange server

Microsoft Exchange je systém pre distribúciu e-mailových správ, zdieľanie adresárov, podporu spolupráce, správu kalendárov, úloh a kontaktov. Pre prácu so správami používa protokol MAPI/RPC, POP3, IMAP, SMTP. K Exchange serveru a sa dá napojiť pomocou MS Outlook aplikácie, ktorá je súčasťou balíka Microsoft 365 prípadne pomocou webovej aplikácie Outlook Web Application (OWA).

1.5.2 Atlassian aplikácie

Atlassian je Austrálska softvérová spoločnosť zameraná na produktov pre riadenie projektov a vývoj softvéru. V tejto práci sa budeme venovať dvom produktom a to Jira Software a Jira Service Desk. Obe inštancie sú on-premise verzie nainštalovane na Windows Server 2016.

1.5.2.1 Jira Software

Jira Software je súčasťou rodiny produktov určených na pomoc tímom všetkých typov pri riadení práce. Jira je silný nástroj na správu práce pre všetky druhy prípadov použitia od požiadaviek, cez právu testovacích scenárov až po agilný vývoj softvéru. Práve podpora agilného vývoja softvéru je to, v čom Jira vyniká. Pre tímy, ktoré používajú scrum alebo kanban je práve Jira ideálnou voľbou. Jira podobne ako Azure Boards poskytuje požiadavky (tikety), tabuľu, backlog, šprinty a dotazy. [16]

Firma momentálne prechádza agilnou transformáciou, čo je jedným z hlavných dôvodov plánovaná integrácia Jira platformy do podnikovej aplikačnej infraštruktúry. Najväčšia výhoda oproti Azure Boards je jednoznačne široká customizovateľnosť, možnosti reportovania, plánovania, vykazovania a podobne. Dá sa teda povedať, že v rámci Jiry sa dá upraviť podľa požiadaviek takmer všetko. Podrobnú ukážku možnosti úprav si ukážeme neskôr.

1.5.2.2 Jira Service Desk

Jira Service Desk zabezpečuje triedenie, sledovanie a priraďovanie prichádzajúcich požiadaviek z rôznych zdrojov pomocou radov a SLA zmlúv. Jira Service Desk poskytuje možnosť vytvorenia jednoduchého a intuitívneho portálu, cez ktorý môžu zákazníci zadávať požiadavky. V rámci podniku sa tento portál využíva na incident, problém a change manažment projektov, ktoré sú zazmluvnené pomocou SLA. Prístup na tento portál je teda umožnený externým partnerom.

2 CIEĽ PRÁCE, METODIKA PRÁCE A METÓDY SKÚMANIA

V tejto kapitole sa venujeme cieľu záverečnej práce, metodike práce a metódam skúmania, ktoré sme použili pri vypracovaní predloženej diplomovej práce.

2.1 Cieľ práce

Cieľom práce je preskúmať možnosti integrácie medzi vybranými produktami spoločnosti Atlassian s vybranými produktami spoločnosti Microsoft v rámci menšieho IT podniku. Produkty oboch spoločností sme si predstavili v prvej časti práce. Práca teda opisuje aplikáciu vybraných integračných nástrojov a ich funkcionalitu, ktorú následne vyhodnocuje a navrhuje optimálne využitie integrovaných nástrojov na základe prvotných požiadaviek. Na základe tohto prieskumu a testovania budú vybrané nástroje zakúpené a implementované v prevádzke. Dôvodom skúmania integrácií je vízia zaradenia jednaných Atlassian produktov do informačného systému firmy a postupné nahradenie Azure DevOps Server za Jira Software vo funkcií nástroja na riadenie projektov v rámci modernizácie.

Výsledky práce teda budú prezentované a budú slúžiť ako:

- Integrácie budú reálne zavedené v rámci podniku
- Práca bude slúžiť ako dokumentácia k poskytnutému riešeniu
- Práca bude slúžiť ako možná pomôcka pre iné podniky, ktoré hľadajú možnosti integrácie Jiry a Microsoft produktov

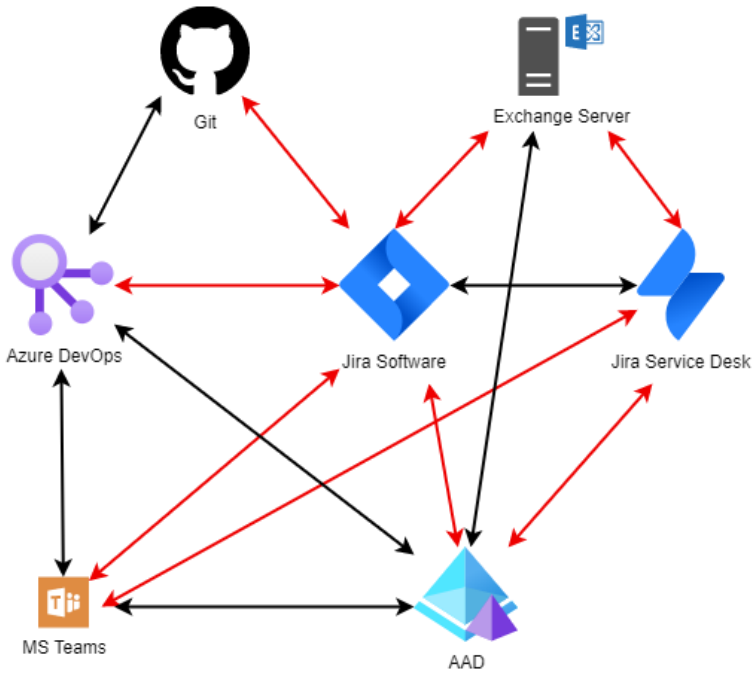
Hlavný cieľ práce teda vieme rozdeliť na jednotlivé pod ciele:

1. **Integrácia Jira Software v rámci intranetu**
    a. Zabezpečiť možnosť prihlásenia (autentifikácie) používateľov za pomoci doménových prihlasovacích údajov. Prihlásenie účtov na základe účtov nachádzajúcich sa v Microsoft Azure Active Directory (AAD).
2. **Integrácia Jira Software a Azure DevOps Server v oblasti tiketového systému**
    a. Synchronizácia dát tiketov. Synchronizovať všetky zmeny v poliach pre využívané typy tiketov pričom nezáleží v ktorom nástroji boli zmenené.
    b. Synchronizácia stavov tiketov. Synchronizovať zmeny a prechody medzi stavmi, pričom nezáleží v ktorom nástroji boli zmenené.

c. Importovanie existujúcich tiketov v rámci aktívnych projektov z Azure DevOps do Jira software

3. **Integrácia Jira Software a Git**
    a. Integrácia Jira a Git ako nástroj na kontrolu verzií.
4. **Integrácia Jira Software/ Jira Service Desk s aplikáciou Microsoft Teams**
    a. Možnosť zdieľania Jira tiketov prostredníctvom Microsoft Teams
    b. Možnosť vytvoriť nový tiket prostredníctvom Microsoft Teams
    c. Pridať komentár/upraviť tiket prostredníctvom Microsoft Teams
    d. Zobraziť základné filtre v Microsoft Teams bez nutnosti otvárania samotnej Jiry.
5. **Integrácia Jira Software/Jira Service Desk s mail serverom Microsoft Exchange**
    a. Automaticky vytvoriť tiket z mailu v aplikácií Jira Software
    b. Automaticky vytvoriť tiket z mailu v aplikácií Jira Service Desk

Finálny pohľad na infraštruktúru je zobrazený na nasledujúcom obrázku, kde sú nami požadované integrácie zobrazené červenou. Čiernou sú zobrazené integrácie, ktoré sú vyhotovené natívne priamo pri inštalácií ako v prípade Jira Software a Jira Service Desk alebo už sú v rámci podniku aplikované a zaužívané a teda nie je potreba ich nijak ďalej riešiť.



Obrázok 4 Mapa integrácií v podniku

2.2 Metodika práce a metódy skúmania

Metodika práce spočíva v postupnosti nasledujúcich krokov:

1. Výber integračného nástroja na základe požiadaviek v cieľoch práce, ceny a náročnosti nastavenia.
2. Inštalácia vybraných nástrojov
3. Konfigurácia nástrojov podľa potrieb podniku
4. Manuálne testovanie funkcionality nasadených riešení
5. Zhodnotenie nástroja opisnou charakteristikou funkcionality a zhodnotenie silných a slabých stránok

Pri výbere nástroja bude hlavným parametrom, aby obsahoval funkcionalitu, ktorá pokryje požiadavky na splnenie stanovených cieľov. Cieľom nie je nájsť najrobustnejší nástroj s funkcionalitou ktorú nepotrebujeme ale nájsť nástroj, ktorým zabezpečíme požadovanú funkcionalitu. Tým sa odvíja aj cena nástroja. Jira má taktiež veľkú komunitu používateľov, ktorý môžu jednotlivé aplikácie v rámci obchodu hodnotiť a recenzovať. Preto budeme hľadať spomedzi aplikácií s najvyšším počtom stiahnutí a najlepšími recenziami. Prirodzene keďže hľadáme iba vybranú funkcionalitu, hľadáme k tejto funkcionalite aj optimálnu cenu za licenciu. Ceny licencií budeme porovnávať v základnej skupine do 10 používateľov. Po nainštalovaní a konfigurácií jednotlivých nástrojov podľa potrieb budeme funkčnosť nástrojov manuálne testovať podľa scenárov očakávaného používania aplikácií. Po otestovaní detailnejšie opíšeme funkcionalitu, zhodnotíme jej pozitíva a prípadné nedostatky, na ktoré si treba dať pri používaní pozor.

3 VÝSLEDKY PRÁCE

V našej práci pri realizácií integrácie využijeme aplikácie tretích strán. Aplikácie tretích strán v Jire môže vyvíjať ktokoľvek a teda k dispozícií je široký výber rôznych riešení. V rámci našej práce sa venujeme integrácií na základe vybraných aplikácii ako sú:

- Spartez TFS4JIRA
- Adaptavist Scriptrunner for JIRA
- Git Integration for Jira
- Microsoft Teams for Jira Server
- Microsoft Azure Active Directory single sign-on for JIRA

Konkurentov týchto aplikácií ako aj samotné aplikácie si rozoberieme v kapitolách venovaných integráciám jednotlivých aplikácií.

Dostupnosť doplnkových aplikácií sa dá klasifikovať na dve skupiny:

a. Aplikácie schválené a predávané cez Atlassian obchod
b. Aplikácie, ktoré si vieme nahrať do Jiry manuálne

Nakoľko je postup inštalácie aplikácií do Jiry rovnaký, oboznámime sa s ním v tejto časti. Postup pri inštalácií doplnkovej aplikácie z Atlassian obchodu je nasledovný: [17]

1. Ako používateľ s admin právami v hlavnom navigačnom paneli zvolíme v menu nastavenia možnosť „Správa aplikácií“
2. V menu v ľavej časti obrazovky zvolíme „Nájsť nové aplikácie“
3. Vyhľadáme aplikáciu podľa mena alebo kategórie
4. Nasledujeme výzvu k Inštalácií, nákupu alebo skúšobnej verzie
5. Po nainštalovaní treba vyplniť položku Licenčný kľúč. Kľúč získame po prihlásení do Atlassian portálu v sekcií licencie.

V prípade, že inštalujeme aplikáciu, ktorá sa nenachádza na Atlassian obchode postupujeme nasledovne: [17]

1. Ako používateľ s admin právami v hlavnom navigačnom paneli zvolíme v menu nastavenia možnosť „Správa aplikácií“
2. V menu v ľavej časti obrazovky zvolíme „Správa aplikácií“

3. Následne klikneme na „Nahrať aplikáciu“
4. Vyberieme inštalačný súbor aplikácie z nášho počítača alebo použijeme URL adresu k požadovanej aplikácií a stlačíme „Nahrať“

Po inštalácií aplikácie tretej strany je potrebné každú aplikáciu nakonfigurovať. Konfigurácií jednotlivých aplikácií ako aj ďalšie potrebné kroky si vysvetlíme v ďalších kapitolách pri individuálnych riešeniach.

3.1 Integrácia Jira Software a Azure Active Directory

Možnosti nástrojov pre integráciu Jiry s AAD je naozaj mnoho: sú to napríklad:

- SAML Single Sign On (Jira SSO) Jira SAML SSO
- mO Jira SAML SSO/Jira SSO/Jira Single Sign On SSO/SAML Login
- Microsoft Azure Active Directory single sign-on for JIRA

Nástroj Microsoft Azure Active Directory single sign-on for JIRAsme vybrali z dôvodu, že jeho poskytovateľ je priamo spoločnosť Microsoft a teda ide o originál ich produkt, ktorý navyše poskytujú zadarmo. Navyše oba zvyšné nástroje poskytujú funkcionalitu, ktorú nepotrebujeme ako napríklad plná synchronizácie používateľov a skupín z AAD. Nakoľko do Jiry nebudú pristupovať všetci nie je potrebné aby tam boli a stačí nám jednoduchá autentifikácia, ktorá je zdarma.

Základný význam intranetu a funkcionality AAD sme si predstavili v predchádzajúcich kapitolách. Teraz sa poďme pozrieť na využitie funkcionality AAD ako nástroja na zabezpečenie jednotného prihlásenia pre cloudové aplikácie. Vďaka nastaveniu tejto funkcionality neskôr funguje synchronizácia užívateľov pri integrácií Jiry a DevOps, zobrazenie filtrov a práca s Jirou prostredníctvom integrácie s MS Teams a tak do istej miery aj pri automatizovanom vytváraní tiketov prostredníctvom mailu. Pre nastavenie tejto integrácie budeme musieť spraviť nastavenia v dvoch aplikáciách – AAD pomocou Azure portálu a aplikácie v Jire.

Ako je potrebné nastaviť AAD pre možnosť jednotného prihlásenia: [18]

1. Cez Azure portál na integračnej stránke aplikácie JIRA SAML SSO by Microsoft, nájdeme možnosť spravovať a zvolíme jednotné prihlásenie.
2. Po zvolení jednotného prihlásenia vyberieme možnosť SAML
3. Pri nastavení jednotného prihlásenia so SAML klikneme na ikonu ceruzky pre nastavenie

4. V základnej sekcií SAML nastavenia vyplníme nasledovné polia:
    a. Prihlasovacia URL zadáme URL v nasledujúcom tvare: https://<domain:port>/plugins/servlet/saml/auth
    b. Do poľa identifikátor URL v nasledujúcom tvare https://<domain:port>/
    c. Do poľa URL pre odpoveď zadáme URL v nasledujúcom tvare: https://<domain:port>/plugins/servlet/saml/auth

    pričom časť URL adries označených ako <domain:port> je nahradených našou reálnou doménou ktorá ukazuje na našu aplikáciu.

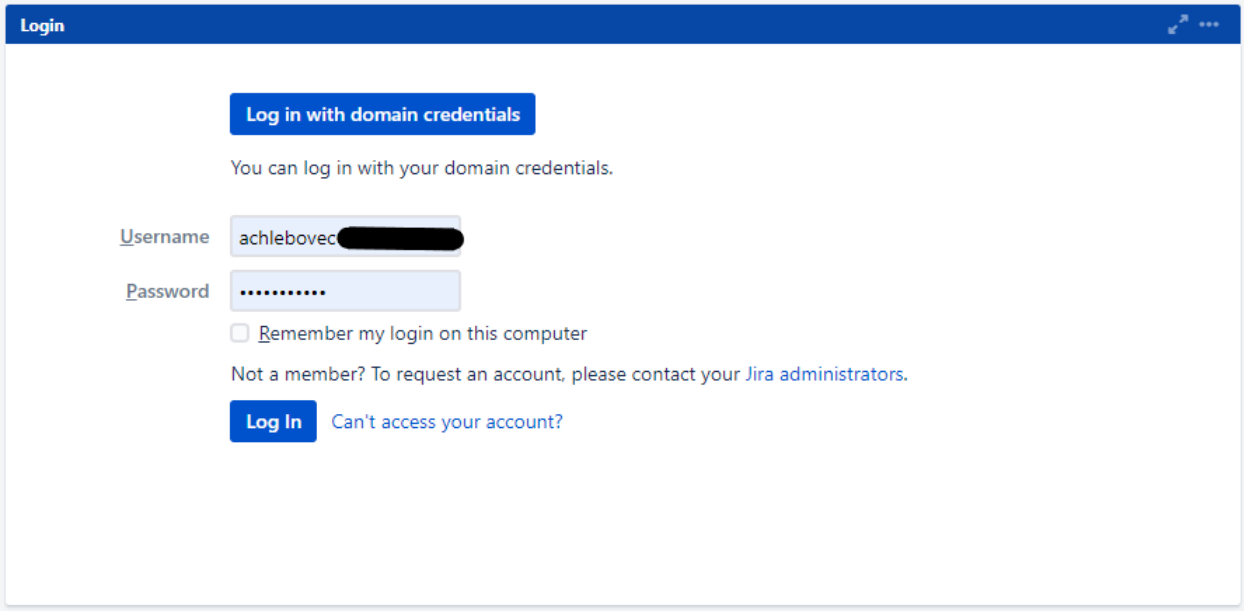
Ako druhé nainštalujeme a nastavíme aplikáciu JIRA SAML SSO by Microsoft na strane Jiry. Aplikácia sa nenachádza v Atlassian obchode a budeme ju musieť nainštalovať manuálne. [18]

1. Stiahneme aplikáciu JIRA SAML SSO by Microsoft z Microsoft centra.
2. Ako používateľ s admin právami v hlavnom navigačnom paneli zvolíme v menu nastavenia možnosť „Správa aplikácií“
3. Zvolíme možnosť „Nahrať Aplikáciu“ a nájdeme stiahnutý inštalačný súbor v našom počítači.
4. Po nainštalovaní sa aplikácia zobrazí v zozname nainštalovaných aplikácií. V tomto zozname aplikáciu nájdeme a klikneme na možnosť „nastaviť“
5. Vyplníme všetky nasledujúce polia:
    a. Do poľa Metadáta URL vložíme adresu Metadát federovaných aplikácii kopírovanú z Azure portálu a stlačíme tlačidlo vyriešiť
    b. Do poľa Identifikátor, URL pre odpoveď a prihlasovaciu URL skopírujeme hodnoty, ktoré sme zadali pri nastavovaní AAD v kroku číslo 4
    c. Do poľa názov tlačidla pre prihlásenie zadáme hodnotu, ktorú chceme zobrazovať na tlačidle pri prihlasovaní. V našom prípade „Log in with domain credentials“
    d. Do poľa opis tlačidla na prihlásenie zadáme inštrukciu pre používateľov. V našom prípade „You can log in with your domain credentials.“

e. V časti SAML ID používatela zaškrtneme možnosť ID používateľa sa nachádza v názve používateľa

Po tom ako máme nastavené obe časti nášho riešenia je potrebné v rámci Jiry vytvoriť používateľov tak, aby meno používateľa v Jire bolo rovnaké, ako prihlasovacie meno zadané v AAD. Na základe tohto mena sa potom následne prebehne autentifikácia.

Integrácia AAD s Jirou nám priniesla možnosť jednotného prihlásenia s ďalšími aplikáciami v rámci podnikového intranetu. Nástroj však funguje iba za účelom autentifikácie a teda je potrebné v rámci Jiry vytvárať používateľov manuálne a nastaviť im používateľské mená rovnaké, ako sú prihlasovacie mená do intranetu. Ako môžeme vidieť na obrázku nižšie, ku klasickej možnosti prihlásenia do Jiry máme aj nové tlačidlo pre prihlásenie sa pomocou ADFS – čiže jednoducho povedané za pomoci autentifikácie cez AAD. Ako môžeme vidieť, názov tlačidla ako aj popis pod ním je taký, aký sme zadali pri nastavovaní.



Obrázok 5 Zobrazenie možnosti prihlásenia sa pomocou autentifikácie cez AAD

Celý proces Autentifikácie teda prebieha tak, že používateľ stlačí tlačidlo pre prihlásenie sa a je automaticky presmerovaný autentifikovať sa pomocou prihlásenia do Microsoft 365. Keďže Microsoft účty sú uložené v rámci AAD, aplikácia sa tak autentifikuje na strane AAD a po úspešnej autentifikácií vytvorí klasický prihlasovací token napojený k používateľskému menu. Keďže používateľské mená sa musia rovnať, prihlásenie je autentifikované a prihlásenie

do Jiry úspešné. Takéto prihlásenie Jira vníma, ako klasické prihlásenie a je možné teda sledovať históriu prihlásení, neúspešných pokusov a podobne. Zaujímavý je aj fakt, že pri vytváraní účtu do Jiry nie je potrebné vôbec nastavovať žiadne heslo. Zaujímavé je, že prihlásenie cez stranu Jira Service Desk portálu nie je podporované, avšak ak je používateľ s licenciou pre Jira Software aktuálne prihlásený a nemá Jira Service Desk licenciu, tak je automaticky prihlásený aj na portál. Prihlásenie takýmto spôsobom cez portál však nie je možné. Táto integrácia pracuje na leveli aplikačného rozhrania.

Výhody

- Aplikácia potrebná pre integráciu je úplne zdarma
- Prihlásenie je rýchle a spoľahlivé
- V prípade potreby, je možné sa prihlásiť aj klasicky cez prihlásenie do Jiry
- Možnosť vytvoriť si skupiny používateľov odlišných od skupín v AAD bez negatívnych vplyvov

Nevýhody

- Potreba manuálne vytvárať používateľov s rovnakým používateľským menom ako v AAD
- Nefunguje na prihlásenie cez Service Desk portál
- Nie je možné využiť skupiny vytvorené v rámci AAD

3.2 Integrácia Jira Software a Azure DevOps Server v oblasti tiketového systému (Azure Boards)

Pri výbere integračného nástroja sme vyberali spomedzi dvoch dostupných aplikácií

- Exalate
- TFS4JIRA

Oba nástroje poskytujú synchronizáciu tiketov v reálnom čase pričom TFS4JIRA poskytuje možnosť synchronizácie všetkých polí vzťahov a taktiež poskytuje konfiguráciu nástroja prostredníctvom používateľského rozhrania, ktoré je oproti nástroju Exalate bohatšie, nakoľko na mapovanie dát je potrebné v prípade Exalate využívať zväčša iba skriptovanie. Okrem toho, že je aplikácia Exalate náročnejšia na použitie a poskytuje horšie konfiguračné schopnosti je aj

výrazne drahší. Zatiaľ čo TFS4JIRA stojí ročne pre 10 používateľov 10 dolárov, v prípade Exalate je to 295 dolárov ročne pre neobmedzený počet používateľov.

Pri riešení tejto integrácie, je potrebné si najprv ujasniť nezrovnalosti v názvosloví. V prvom rade je potrebné si uvedomiť, že v ide o tiketové systémy a preto je základnou jednotkou tiket. Každý zo systémov má pre tiket svoje vlastné pomenovanie aj keď ide o rovnakú vec. Jira nazýva tiket požiadavka zatiaľ čo Azure DevOps pracovná požiadavka. V kapitole o Azure DevOps sme si povedali, že tiketový systém sa nazýva Azure Boards. V tejto práci sa však stretávame aj s pojmom TFS (Team Foundation Server). Jedná sa o predchodcu Azure DevOps z čias ešte pred cloudovými riešeniami kde sa TFS „premenovalo“ na Azure DevOps server až v roku 2018. Ide len o vylepšenú, modernejšiu verziu aplikácie s tým istým názvom.

Pre integráciu Jira Software a Azure DevOps Server (konkrétne aplikácia Azure Boards) sme vybrali riešenie od spoločnosti Spartez Software. Toto riešenie sa skladá z dvoch častí: [19]

1. *TFS4JIRA* – doplnok inštalovaný do Jiry
2. *TFS4JIRA Synchronizer* – webová aplikácia

Tieto dve aplikácie zabezpečujú automatizovanú synchronizáciu tiketov medzi Azure Devops a Jirou v oboch smeroch. To znamená, že v prípade ak nastane akákoľvek zmena v Jire tak sa automaticky aktualizuje aj DevOps a tak isto ak by nastala zmena tiketu na strane Jiry, tak sa tiket automaticky aktualizuje aj na strane Azure DevOps Board.

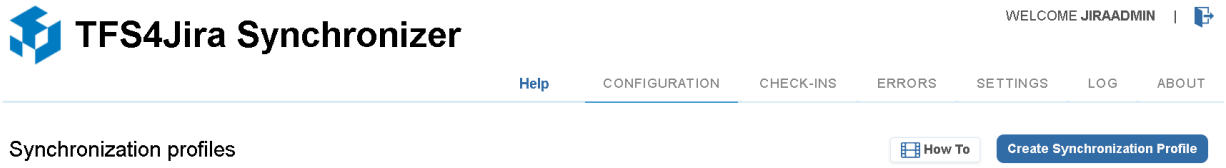
Inštalácia

Ako sme už spomínali je potrebné nainštalovať dva komponenty: [19]

- TFS4JIRA Jira plugin nainštalujeme ako klasický plugin. Inštaláciu klasického pluginu sme si opísali na začiatku tejto kapitoly.
- TFS4JIRA Synchronizer je backendová webová aplikácia.
    1. Pripojíme sa na ľubovoľný server, na ktorom nám bude bežať aplikácia. Na type servera nezáleží dôležité je, aby mal sieťový prístup ako aj k Jire tak aj k DevOpsu. V našom prípade inštalujeme na rovnaký server (Windows Server 2019) na akom máme nainštalovanú našu on-premise Jira inštanciu

2. Spustíme ako administrátor inštalačný súbor stiahnutý z oficiálnej stránky firmy Spartez Software.
3. Postupujeme podľa inštrukcií. V rámci inštalácie nie je potrebné nič meniť ani nastavovať.
4. Po nainštalovaní reštartujeme server
5. Keďže sme nenastavovali žiadne špecifické veci aplikácia nám beži na serveri na porte 222 a teda k aplikácií pristupujeme cez adresu http://localhost:222/
6. Pre prihlásenie do aplikácie je potrebné použiť prihlasovacie údaje, ktorými sa prihlasujeme aj na náš Windows server.

Po prihlásení sa nám zobrazí úvodná stránka s vylistovanými Synchronizačnými profilmi.



Obrázok 6 TFS4Jira Synchronizer hlavné menu

Nakoľko sa budeme venovať iba nastavovaniu Synchronizačného profilu. Budeme sa pohybovať iba v sekcií „Configuration“, kde zvolíme možnosť vytvoriť nový synchronizačný profil (Create Synchronization Profile). Po kliknutí na nový synchronizačný profil následne musíme nastaviť pripojenie nástroja ako na Jiru tak aj na Azure DevOps.

1. V hlavných nastaveniach zadáme názov synchronizačného profilu a časový interval, na základe ktorého bude nástroj kontrolovať zmeny
2. V časti konfigurácie Jiry vyplníme jednotlivé polia:
    a. URL adresu našej Jira inštancie, v prípade, že by sme mali Jiru v cloudovej verzii môžeme zaškrtnúť možnosť Jira cloud.
    b. Vyplníme prihlasovacie údaje účtu, pod ktorým bude prebiehať synchronizácia. Účet musí mať priradené práva na tvorbu a úpravu požiadaviek.
    c. Vyplníme pole projekt, v ktorom určíme, pre ktorý projekt je synchronizačný profil určený.

d. Do poľa „TFS Custom Field name“ zadáme pole z Jiry, do ktorého bude zadané ID tiketu z Azure DevOps pre potrebu spárovania dvojice synchronizovaných tiketov.

3. V časti TFS (DevOps) konfigurácie vyplníme jednotlivé polia:
    a. URL adresu našej DevOps inštancie, v prípade, že by sme mali DevOps v cloudovej verzii môžeme zaškrtnúť možnosť Azure DevOps Services.
    b. Následne vyplníme prihlasovacie údaje účtu, pod ktorým bude prebiehať synchronizácia. Účet musí mať priradené práva na tvorbu a úpravu požiadaviek.
    c. Vyberieme kolekciu projektov a konkrétny projekt, pre ktorý je synchronizačný profil určený
    d. Následne tak ako pri nastavení Jiry potrebujeme vybrať pole, do ktorého budeme ukladať ID požiadavky z Jiry. Nakoľko v Azure DevOps nevieme pridať vlastné pole pre tento účel, zvolíme možnosť uchovávania Jira ID požiadavky za pomoci komentára

Pred tým, ako budeme konfigurovať synchronizačné profily, je potrebné vykonať potrebnú analýzu ako Jiry tak aj DevOps Boards. Nakoľko Integrujeme Jiru ako nový nástroj, do ktorého budeme synchronizovať dáta, budeme musieť tento nástroj prispôsobiť. Široká customizácia Jiry je jedným z dôvodov, prečo je Jira taká populárna, a prečo bola zvolená ako nové riešenie na vedenie projektov. V prvom rade je teda potrebné zmapovať polia, stavy, vzťahy, typy požiadaviek prípadne jednotlivé hodnoty na základe využívaných hodnôt v Azure DevOps. Zoberieme teda položky v DevOps Boards, vytvoríme im ekvivalenty v Jire a následne ich zmapujeme. Možnosti nastavenia mapovania sú napríklad nastavenie smeru synchronizácie, nastavenie konkrétnych hodnôt a nastavenie či chceme polia mapovať všeobecne ako reťazec znakov alebo konkrétnu hodnotu poľa. Všetky mapovania sú orientované na stav a potreby firmy. Možnosti a kombinácie mapovania sú veľmi flexibilné.

Ako prvé zmapujeme jednotlivé polia. Polia. Väčšina polí sa v Jire nenachádza a preto je potrebné ich vytvoriť. Tabuľka nižšie zobrazuje polia, ktoré chceme navzájom mapovať.

Jira pole	Smer Synchronizácie	Azure Board
Epic Name	↔	Title
Summary	↔	Title

Assignee	↔	Assigned To
Sprint	↔	Iteration Path
Epic link	↔	Area
Story Points	↔	Story Points
TFSWorkItemID	↔	ID
Steps to reproduce	↔	Repro Steps
Created in TFS	←	Created Date
Activity	↔	Activity
Acceptance Criteria	↔	Acceptance Criteria
Changed By	↔	Changed By
Changed Date	↔	Changed Date
Closed By	↔	Closed By
Closed Date	↔	Closed Date
Closed Status	↔	Closed Status
Closed Status Code	↔	Closed Status Code
Sprint ID	↔	Iteration ID
Resolved By	↔	Resolved by
Severity	↔	Severity
TFS Resolved Reason	←	Reason
Description	↔	Description
Environment	↔	System Info
Labels	↔	Tags
Priority	↔	Priority
Reporter	←	Created by
Time Tracking Estimated	↔	Original Estimate
Time Tracking Logged	↔	Completed work

Tabuľka 1 Tabuľka mapovania polí

Ako sme už spomínali, keďže Jira je v rámci Informačného systému nová, nenachádzajú sa v nej všetky potrebné polia. Preto je potrebné zistiť, či sa zodpovedajúce pole v Jire nachádza a ak nie, je potrebné ho vytvoriť.

Jira pole	Pôvod	Typ poľa
Epic Name	Systém	Textové pole
Summary	Systém	Textové pole
Assignee	Systém	Výber užívateľa
Sprint	Systém	Textové pole
Story Points	Systém	Číselné pole
TFSWorkItemID	Vlastné	Textové pole
Steps to reproduce	Vlastné	Textové pole
Created in TFS	Vlastné	Dátum
Activity	Vlastné	Výber zo zoznamu
Acceptance Criteria	Vlastné	Textové pole
Changed By	Vlastné	Výber užívateľa
Changed Date	Vlastné	Dátum
Closed By	Vlastné	Výber užívateľa
Closed Date	Vlastné	Dátum
Closed Status	Vlastné	Výber zo zoznamu
Closed Status Code	Vlastné	Výber zo zoznamu
Sprint ID	Vlastné	Číselné pole
Resolved By	Vlastné	Výber užívateľa
Severity	Vlastné	Výber zo zoznamu
TFS Resolved Reason	Vlastné	Textové pole
Description	Systém	Textové pole
Environment	Systém	Textové pole
Labels	Systém	Pole reťazcov
Priority	Systém	Výber zo zoznamu
Reporter	Systém	Výber užívateľa
Time Tracking Estimated	Pole Doplnku	Číslo
Time Tracking Logged	Pole Doplnku	Číslo

Tabuľka 2 Tabuľka pôvodu a typov používaných polí v Jire

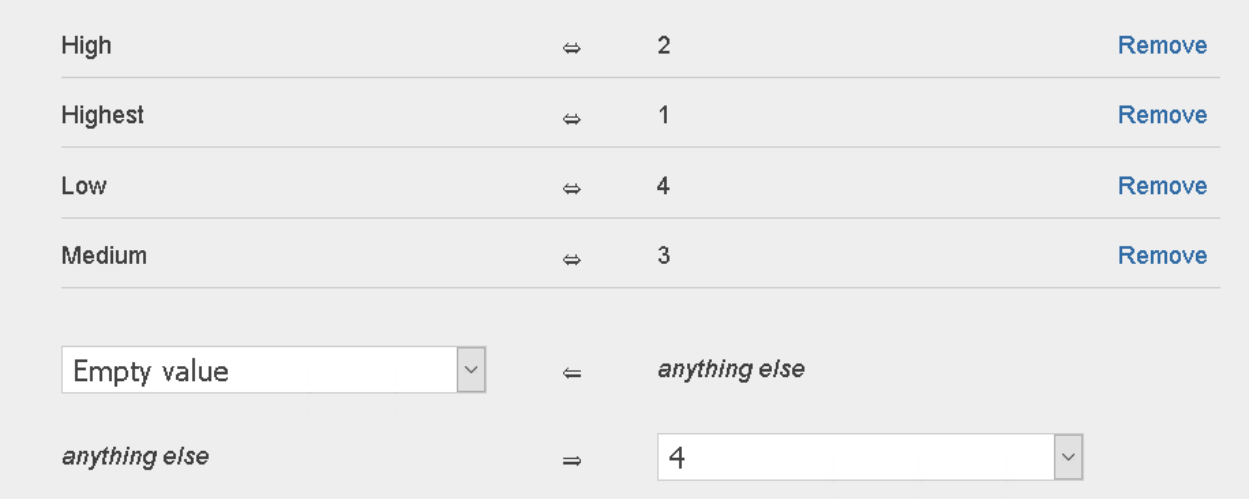
Vlastné pole sa v Jire vytvorí nasledovne: [20]

1. V účte s admin právami zvolíme v hlavnom menu „Správa Systému Jira“ → „Požiadavky“
2. Na stránke Požiadavky potom nájdeme v paneli v ľavej časti „Vlastné polia“
3. V sekcii vlastné polia potom klikneme na „Pridať vlastní pole“
4. Vyberieme typ poľa a klikneme tlačidlo „Ďalej“
5. Zadáme meno poľa a stručný popis na čo pole slúži a klikneme tlačidlo „Vytvoriť“

Takto vytvorené pole však nie je dostačujúce, nakoľko ho systém Jiry možno eviduje, avšak nato aby sme s poľom mohli pracovať je potrebné ho priradiť na obrazovku. [21]

1. V účte s admin právami zvolíme v hlavnom menu „Správa Systému Jira“ → „Požiadavky“
2. Na stránke Požiadavky potom nájdeme v paneli v ľavej časti „Obrazovky“
3. Tu môžeme vytvoriť novú obrazovku alebo konfigurovať už existujúcu. Keďže my pracujeme so synchronizačným nástrojom, ktorý požiadavky vytvára alebo edituje je potrebné pridať polia na obrazovku nastavenú pri vytváraní alebo editovaní požiadavky. Nájdeme preto požadovanú obrazovku a klikneme na „Konfigurácia“
4. Zobrazí sa nám zoznam polí viditeľných na obrazovke. Ak prejdeme na spodok zoznamu nájdeme tam pole poskytujúce výber zo zoznamu. V tomto výbere vyberieme potrebné polia a pridáme ich na obrazovku pomocou tlačidla „Pridať“

Niektoré polia ako napríklad typu výber zo zoznamu je potrebné ešte špecificky mapovať. Dosiahneme to tak, že pri zmapovanej dvojici kliknime na tlačidlo „...“ následne tak môžeme zmapovať jednotlivé hodnoty.



Obrázok 7 Náhľad na nastavenie mapovania poľa priority

Pri každom mapovaní je taktiež možné nastaviť, čo sa udeje v prípade, že sa v zozname Jiry nevyskytuje položka zoznamu z TFS a naopak. Ako môžeme vidieť na obrázku vyššie, hodnota priority v Jire (stĺpec vľavo) môže byť prázdna. Naopak v prípade DevOps Boards je vyplnenie poľa priorita povinné a preto sa v prípade prázdnej hodnoty nastaví najnižšia priorita – 4.

Ďalším krokom je nastavenie mapovania jednotlivých typov požiadaviek respektíve pracovných položiek. Mapovanie typov tiketov je dôležité z hľadiska nastavených pracovných tokov (workflowov), individuálnych zobrazených polí a významu samotného tiketu. Mapovanie je zobrazené v nasledujúcej tabuľke.

Jira požiadavka	Smer synchronizácie	Azure Board pracovná položka
Bug	↔	Bug
Epic	↔	Epic
Feature	↔	Feature
Story	↔	User Story
Task	↔	Task

Tabuľka 3 Tabuľka mapovania typov tiketov

Nesmieme zabudnúť, že medzi jednotlivými typmi tiketov sú pre zlepšenie prehľadnosti a uľahčenie projektového riadenia určité vzťahy. Najčastejšie vyskytujúce sa vzťahy sú

rodič/potomok a takzvaný Epic. Zatiaľ čo vzťah rodič/potomok je zahrnutý spolu so synchronizáciou komentárov, príloh a hypertextového prepojenia medzi štandardným prednastaveným mapovaním, epic mapujeme na základe poľa Epic link (Jira) a Area (DevOps). Zvyšné využívané vzťahy je potrebné mapovať ručne. Mapované vzťahy zobrazuje nasledujúca tabuľka

Jira vzťahy	Smer synchronizácie	Azure Boards vzťahy
relates to / relates to	↔	Related / Related
is duplicated by / duplicates	↔	Duplicate of / Duplicate

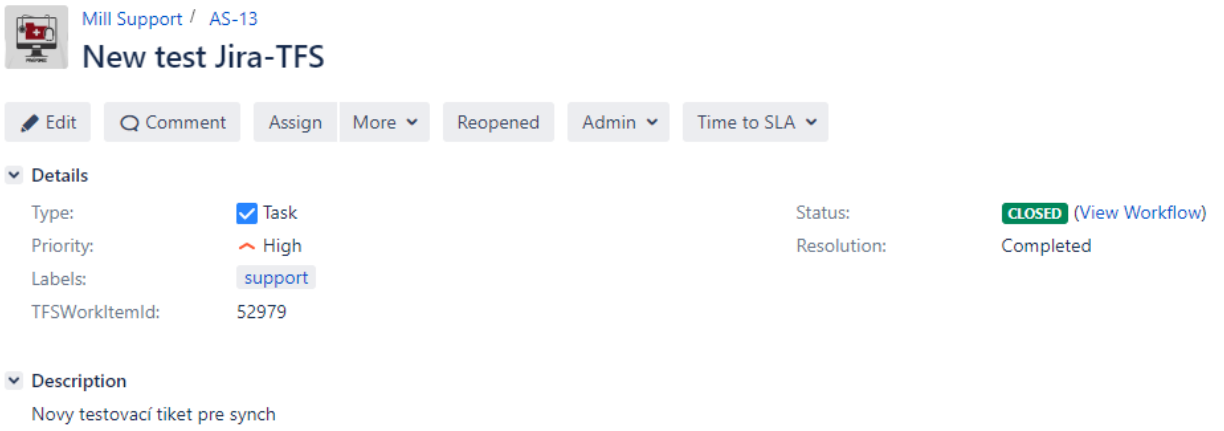
Tabuľka 4 Tabuľka mapovania vzťahov

Ako posledné je potrebné zmapovať jednotlivé stavy. Stavy sú základné kamene zobrazujúce tok práce jednotlivých druhov tiketov. Nakoľko názvy jednotlivých stavov sa môžu čiastočne líšiť, no ich význam môže byť pre našu potrebu rovnaký, niektoré stavy mapujeme aj pomerom 2:1. V takomto prípade stav označený ako „Default“ bude mať pri mapovaní väčšiu prioritu. Ak sa v tikete taký stav nenachádza, synchronizačný nástroj skontroluje výskyt druhého stavu. Mapovanie stavov zobrazuje nasledujúca tabuľka.

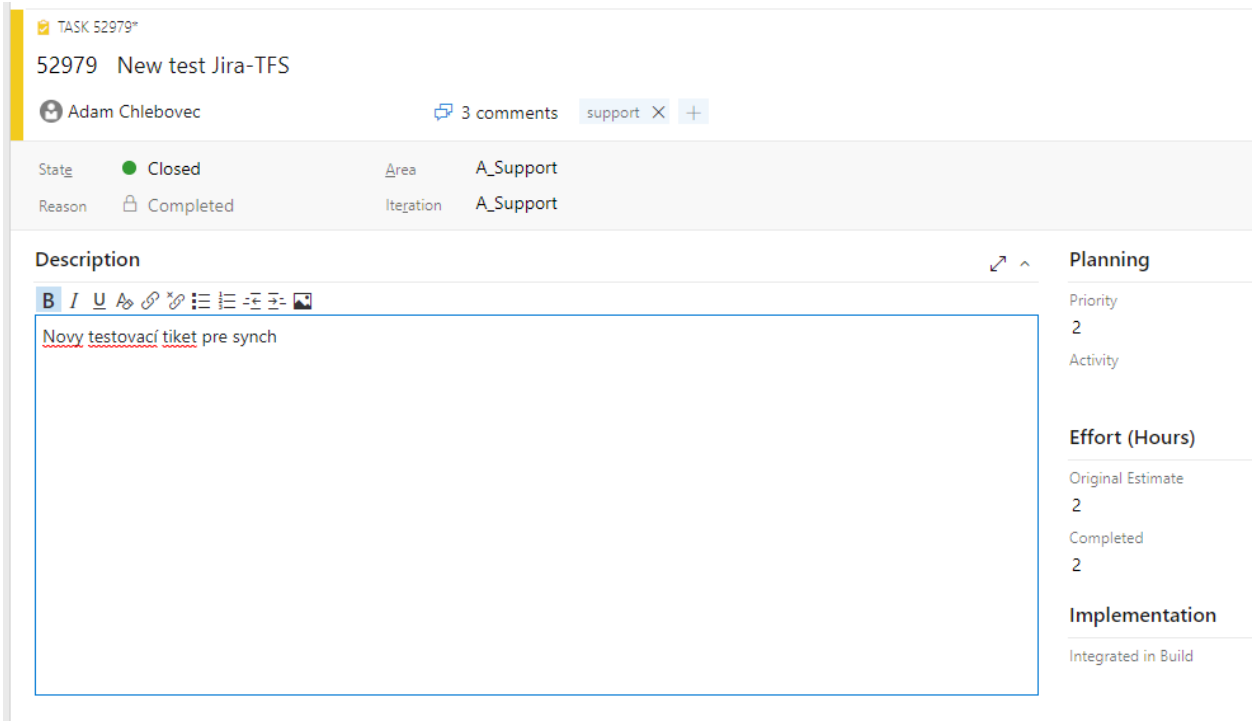
Jira stav	Smer synchronizácie	Azure Board stav
Active (Default)	↔	Active
Reopened	↔	
Approved	↔	Accepted
Closed (Default)	↔	Closed
Passed	↔	
In Progress	↔	In Progress (Default)
	↔	Ready
New	↔	New
Open	↔	Design
Removed	↔	Removed
Resolved	↔	Resolved

Tabuľka 5 Tabuľka mapovania stavov

Po zmapovaní a nastavení všetkých parametrov aktivujeme synchronizačný profil, ktorý sleduje všetky uložené zmeny na oboch stranách. Fungovanie si ukážeme a otestujeme prostredníctvom vytvorenia požiadavky v Jire a sledovanie stavu poli v TFS a naopak. Synchronizované dáta si ukážeme na nasledujúcich obrázkoch.



Obrázok 8 Ukážka základných synchronizovaných polí Jira

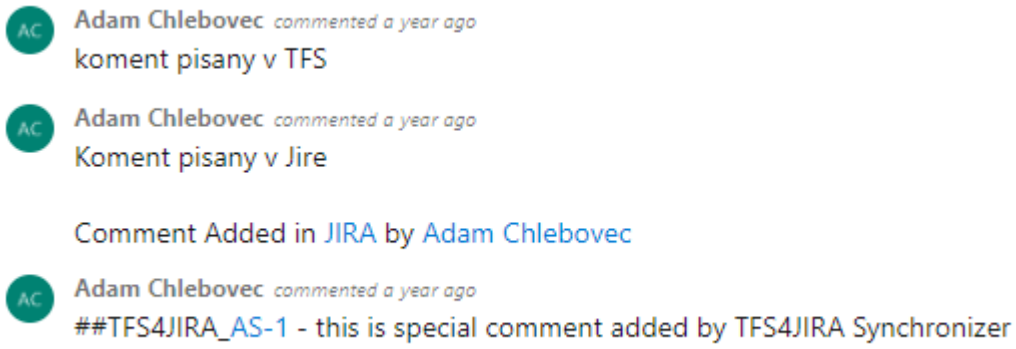


Obrázok 9 Ukážka základných synchronizovaných polí DevOps Boards

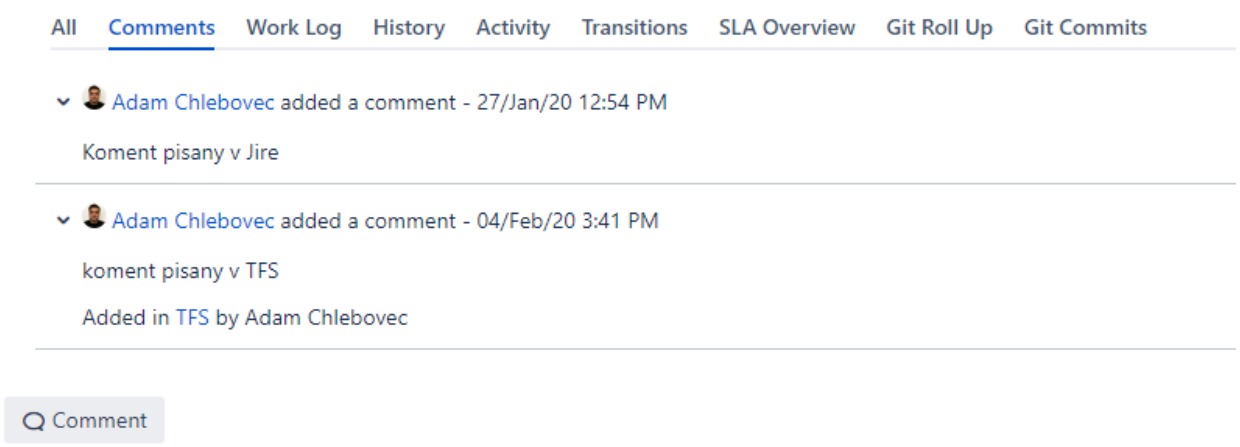
Ako môžeme vidieť na obrázku polia obsahujú rovnaké hodnoty v poliach:

- Typ požiadavky: „Task“
- Názov požiadavky: „New test Jira-TFS“
- Stav: „Closed“
- Reason/Resolution pole je riešené jednosmernou synchranizáciou DevOps → Jira z dôvodu, že pole v DevOps sa volí automaticky a nie je možné ho pomocou synchronizátora meniť. Informáciu preto z DevOps prevedieme do Jiry do poľa „TFS Resolved Reason“ a následne ho automatizovane pri prechode do vyriešeného stavu doplníme. V tomto prípade môžeme vidieť hodnotu „Completed“
- Popis požiadavky: „Nový testovací tiket pre synch“
- Priorita je synchronizovaná podľa jednotlivých hodnôtObrázok 7 Náhľad na nastavenie mapovania poľa priority (Obrázok 7 Náhľad na nastavenie mapovania poľa priority)
- Štítok je taktiež v oboch prípadoch s hodnotou „support“

Tieto polia sú teda synchronizované správne podľa ich nastavenia mapovania. Ďalej sa pozrieme na synchronizáciu komentárov.



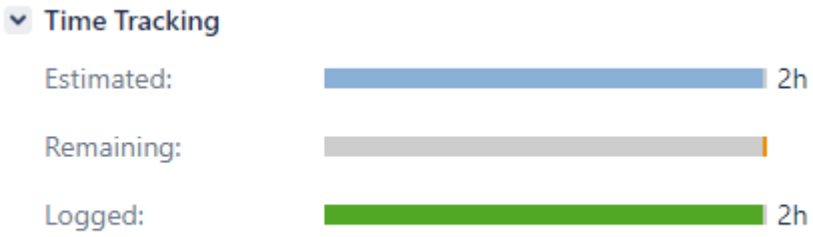
Obrázok 10 Ukážka synchronizovaných komentárov DevOps



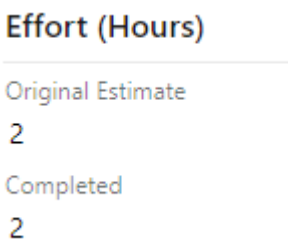
Obrázok 11 Ukážka synchronizovaných komentárov Jira

V rámci synchronizácie komentárov môžeme vidieť rôzne pridané informácie. Ako prvý sa pridá komentár do DevOps Boards, ktorý sa nesynchronizuje. Tento komentár slúži na vytvorenie dvojice s požiadavkou v Jire. Ide o výsledok nastavenia vytvorenie väzby na komentár pri prvotnom nastavovaní synchronizačného nástroja. V Jire je opozitom tohto komentára pole „TFSworkItemId“. V prípade synchronizácie štandardných komentárov nástroj vždy v prenesenom komentári nechá informáciu o tom, kto komentár originálne pridal. Je to z toho dôvodu, že nástroj funguje a vykonáva zmeny v mene používateľského účtu, ktorým je pripojený. Je teda Jedno kto by pridal komentár, ak je v tomto prípade nastavený účet Adam Chlebovec, tak budú všetky komentáre a akcie vykonané používateľom Adam Chlebovec. Preto je dôležité zachovať informáciu o pôvodnom tvorcovi komentáru.

Podobný prístup prenosu pod pripojeným účtom vidíme aj v prípade vykázaných a naplánovaných pracovných hodín.



Obrázok 12 Ukážka synchronizovaných polí v oblasti plánovania a vykazovania v Jire



Obrázok 13 Ukážka synchronizovaných polí v oblasti plánovania a vykazovania v DevOps Boards

Ako môžeme vidieť, hodnoty sú synchronizované správne, v tomto prípade nastáva problém podobný ako pri komentároch, no bohužiaľ tu už nie je tak ošetrený. Ide teda o problém, že prenesená hodnota je zapísanú používateľom, ktorý je nastavený pre synchronizáciu. Dáta sú tým pádom čiastočne znehodnotené a nedajú sa využiť na reportovanie výkazov a plánovanie. Táto skutočnosť zabraňuje bezproblémovej súbežnej práci v oboch nástrojoch súbežne nakoľko dochádza k znečisteniu dát. Ak však zoberieme do úvahy, že vykazovanie a reportovanie práce v DevOps Boards je na podobnej úrovni ako pri synchronizácií, nemôžeme tento nedostatok brať ako úplný mínus. Tento synchronizačný nástroj pracuje na leveli používateľského rozhrania.

Výhody

- Široká možnosť nastavení
- Funguje aj na staršej verzií DevOps bez nutnosti vytvorenia vlastného poľa
- Možnosť mapovania konkrétnych hodnôt

Nevýhody

- Všetky zmeny sú vykonané pod používateľským účtom nastaveným pri vytváraní synchronizačného profilu
- V prípade nastavenia príliš krátkeho synchronizačného intervalu a veľkého počtu synchronizačných profilov sa môže stať, že sa vďaka FIFO prístupu nástroj zasekne.

3.3 Integrácia Jira Software a Git

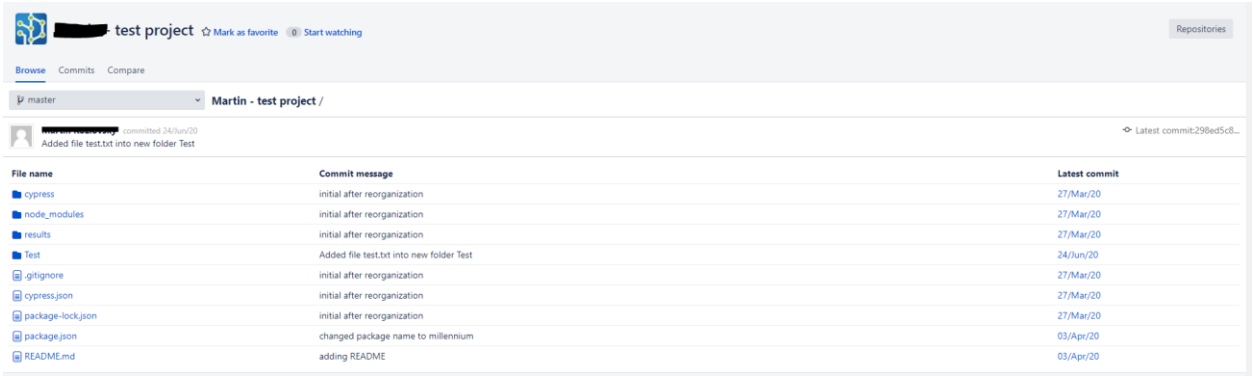
Pre integráciu Jiry s Git použijeme nástroj Git Integrations for Jira, ktorý nájdeme na Atlassian obchode. Výber tohto nástroja bol z dôvodu, že sa jedná o konkrétne riešenie

poskytované Tento nástroj nainštalujeme podľa postupu zo začiatku tejto kapitoly. Po nainštalovaní aplikáciu nájdeme v zozname v zložke „Správa aplikácií“.

1. Po nájdení aplikácie v zozname zvolíme danú aplikáciu a klikneme na „začíname“
2. Následne stlačíme tlačidlo „...“ nachádzajúce sa vedľa tlačidla „Pripojiť“ a zvolíme možnosť Microsoft.
3. V poli Externá služba zvolíme možnosť Team Foundation Server – nakoľko sa tam nachádza náš firemný Git repozitár.
4. Vyplníme URL na ktorej sa nachádza náš repozitár
5. Vyplníme prihlasovacie údaje technického účtu, ktorý má prístupové práva ku všetkým repozitárom a stlačíme pripojiť

V prípade tejto integrácie je postup veľmi priamy a neposkytuje žiadne možnosti iného nastavenia.

Nakoľko Git je nástroj na kontrolu verzií, štandardne integrácia zabezpečuje najmä zobrazovanie jednotlivých súborov v rámci projektov a ich verzií. Integrácia nám zabezpečuje možnosť vidieť projekty a jednotlivé súbory, ako aj zmeny a zároveň poskytuje možnosť jednotlivé zmeny vo verziách priradiť k požiadavkám v Jire. Tento typ integrácie poskytuje pripojenie a zmeny na dátovom leveli.



Obrázok 14 Zobrazenie súborov v repozitári cez Jiru

Výhody

- Veľmi rýchla a efektívna integrácia
- Možnosť priradenia požiadaviek k jednotlivým zmenám

- Integračná aplikácia je zdarma

Nevýhody

- Nie sú

3.4 Integrácia Jira Software/ Jira Service Desk s aplikáciou Microsoft Teams

Pri integrovaní Jiry s MS Teams sú hlavnými favoritmi dve konkurujúce aplikácie:

- Microsoft Teams Jira Connector
- Microsoft Teams for Jira

Obe aplikácie poskytujú presne tú istú funkcionalitu s tým, že nami zvolená aplikácia Microsoft Teams for Jira je vytvorená a podporovaná priamo spoločnosťou Atlassian, takže ide o prémiový licencovaný produkt, ktorý je navyše zdarma oproti 29 dolárovej ročnej licencií aplikácie Microsoft Teams Jira Connector.

Pre integráciu Jiry a MS Teams je potrebné nainštalovať dve aplikácie tretej strany.

- Do Jiry nainštalovať aplikáciu Microsoft Teams for Jira Server
- Do MS Teams nainštalovať aplikáciu Jira Server

Ako prvé nainštalujeme aplikáciu Microsoft Teams for Jira Server do našej inštancie Jiry. Túto aplikáciu nájdeme v Atlassian obchode a nainštalujeme štandardným spôsobom, ktorý sme si opisovali na začiatku tejto kapitoly. Potom je potrebné vykonať nasledujúce kroky: [22]

1. Prejsť do sekcie „administrácia“, následne vybrať možnosť „Aplikácie“ v pravom hornom rohu v Jire
2. Následne vyberieme „Integrácie“ a „Odkazy aplikácií“
3. Následne do okna pre odkaz vložíme adresu https://jira-server.msteams-atlassian.com/ a stlačíme „Vytvoriť nový odkaz“
4. Na prvej obrazovke dialógového ona zadáme do poľa názov aplikácie „Microsoft Teams“, začiarkneme políčko „Vytvoriť prichádzajúci odkaz“ a klikneme „Pokračovať“
5. Na nasledujúcej obrazovke dialógového okna vložíme nasledujúce údaje:

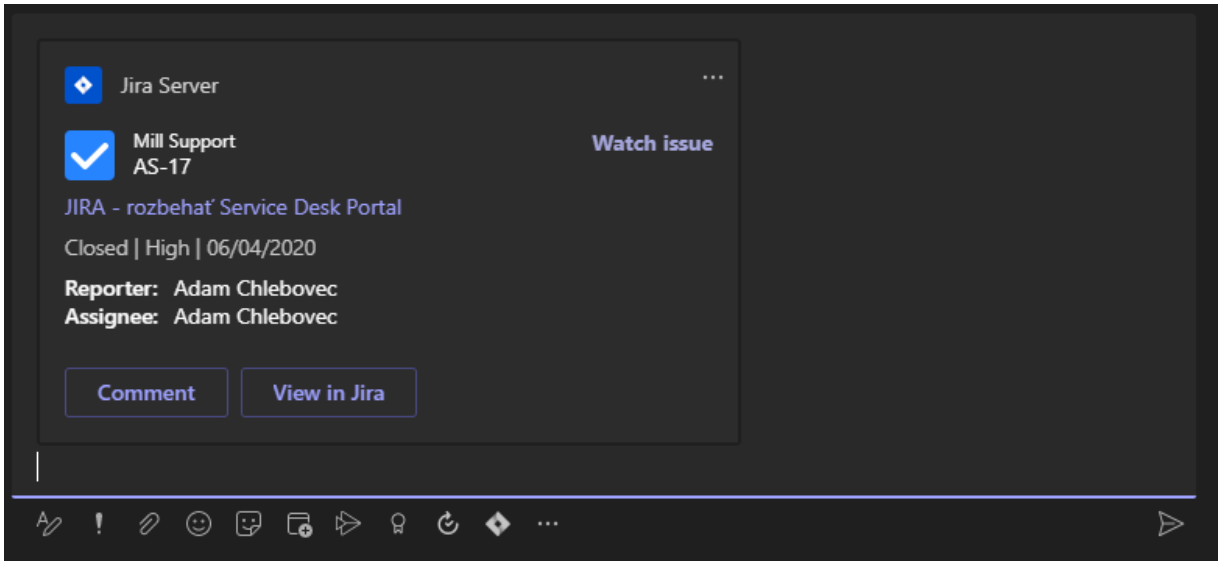
a. Do poľa kľúč zákazníka zadáme hodnotu „OauthKey“
b. Do poľa meno zákazníka zadáme hodnotu „MicrosoftTeamsIntegration“
c. Vyplníme pole verejný kľúč, tento kľúč nájdeme v našej nainštalovanej aplikácií v Jire v časti nastavenie

6. Klikneme na „Pokračovať“
7. Následne sa zavrie dialógové okno a zobrazí sa nám informácia o úspešnom nastavení

Následne nainštalujeme aj aplikáciu do MS Teams. Inštaláciu vykonáme nasledujúcim postupom: [22]

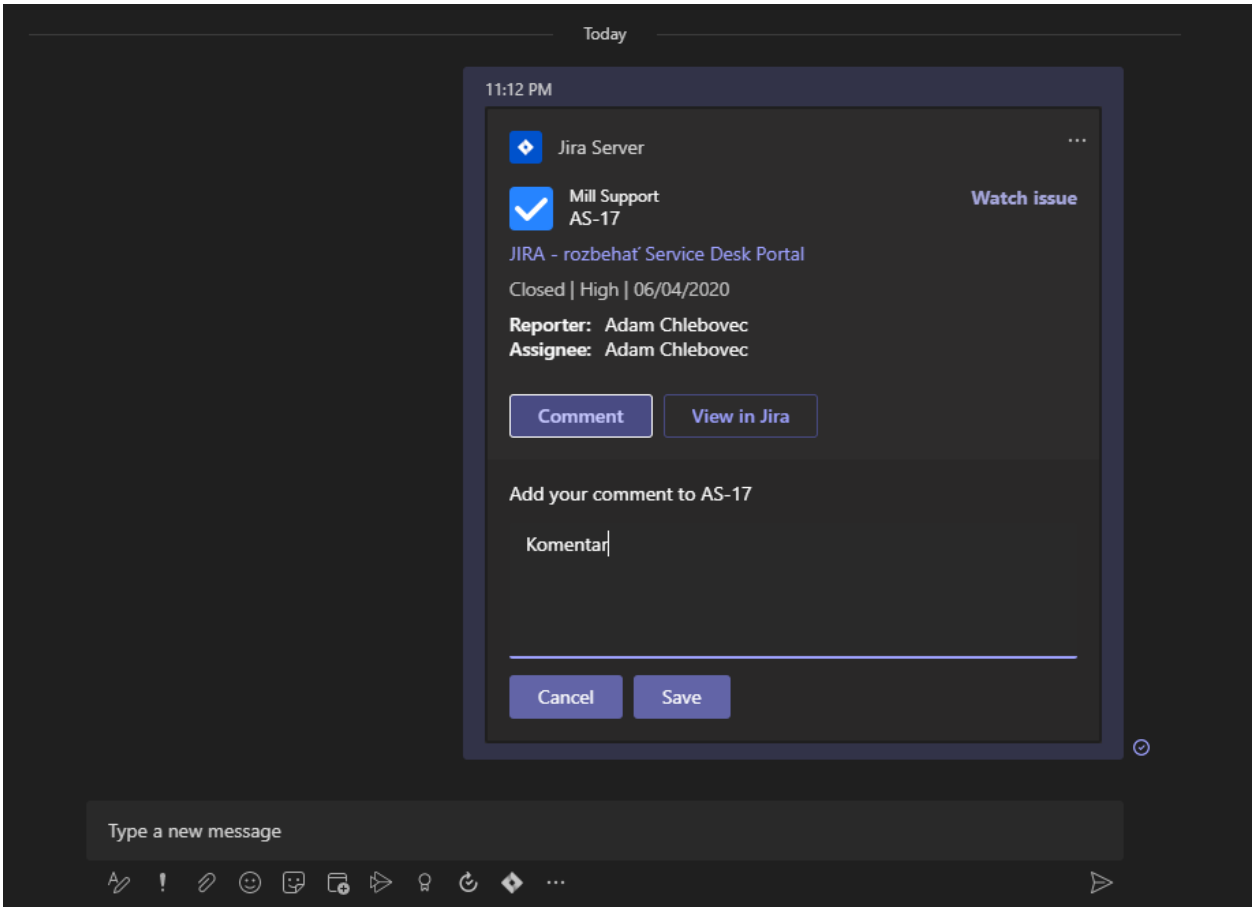
1. Pre inštaláciu aplikácií tretích strán do MS Teams je potrebné mať práva od Office 365 administrátora
2. V MS Teams nájdeme v ľavom dolnom rohu tlačidlo Aplikácie
3. Vo vyhľadávači nájdeme aplikáciu „Jira Server“
4. Zvolíme nainštalovať
5. Po nainštalovaní sa nám zobrazí dialógové okno, ktoré požaduje Jira ID. Jira ID nájdeme opäť v nainštalovanej aplikácii v časti „správa aplikácií“

MS Teams je v rámci podniku hlavný interný komunikačný nástroj. Je preto prirodzené, že jedna z hlavných požiadaviek na integráciu bude možnosť rýchleho a efektívneho zasielania a zdieľania jednotlivých požiadaviek. Pre zdieľanie požiadavky je potrebné v rámci chatu v paneli s doplnkami kliknúť na logo aplikácie Jira Server a následne vyhľadať požiadavku, ktorú chceme zaslať. Požiadavku je najlepšie hľadať podľa kľúča,. Kľúč sa v Jire skladá zo skratky projektu a čísla požiadavky oddelených pomlčkou. V našom zobrazení na obrázku je kľúč pre požiadavku AS-17.



Obrázok 15 Zasielanie požiadavky pomocou MS Teams

Ako môžeme vidieť, zaslaná požiadavka obsahuje aj zopár základných údajov o požiadavke. V prípade, že máme takúto požiadavku zdieľanú v konverzácií, vieme ju samozrejme jedným kliknutím otvoriť priamo v Jire, vieme ju však aj priamo v prostredí MS Teams komentovať po kliknutí na tlačidlo komentovať. Funkciu pridania rýchleho komentára môžeme vidieť na nasledujúcom obrázku.



Obrázok 16 Pridanie rýchleho komentára na požiadavku prostredníctvom MS Teams

Taktiež je možné vytvoriť komentár zo správy. Stačí kliknúť na pole správy a v pravom hornom rohu zvoliť tlačidlo „...“. Po stlačení tohto tlačidla sa nám zobrazí zoznam akcií kde vyberieme ďalšie možnosti a pridať komentár. Následne sa nám zobrazí dialógové okno, kde môžeme správu ešte upraviť a musíme vybrať požiadavku, ku ktorej sa má komentár pridať.

Nástroj podporuje aj funkcionalitu Jiry v oblasti filtrov. V rámci Jiry poznáme tri druhy vyhľadávania:

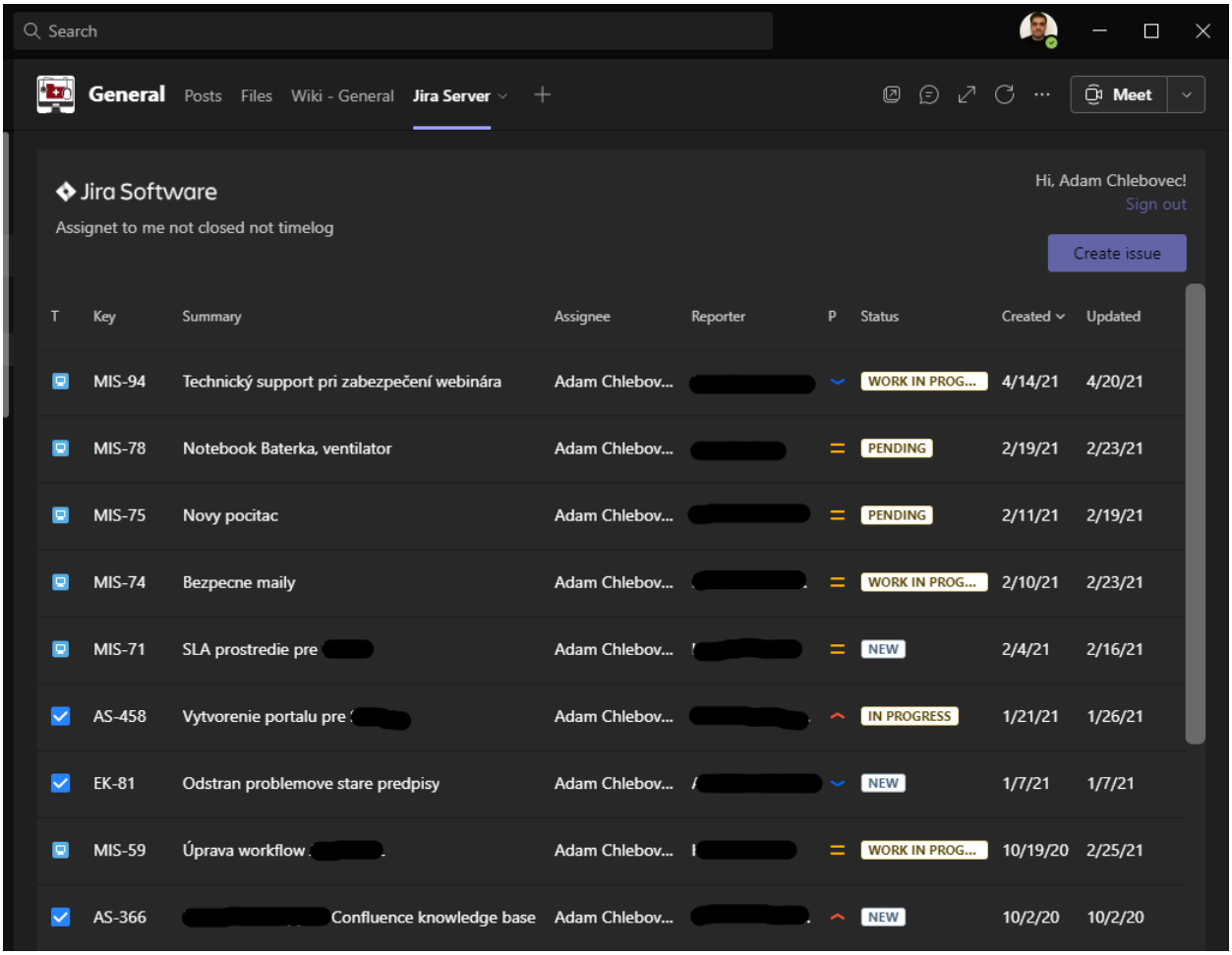
- Rýchle
- Štandardné
- Pokročilé

Keďže nás v rámci nastavovania a integrácie Jiry zaujíma nastaviť systém čo najlepšie, budeme sa venovať pokročilému spôsobu filtrovania. Pokročilé filtrovanie je veľmi užitočné a podporuje ho vlastný Jira dotazovací jazyk JQL (Jira Query Language). Tento jazyk je

navrhnutý tak, aby vyhovoval všetkým potrebám, od rýchleho vyhľadávania založeného na texte až po zložité komplexné vyhľadávanie naprieč tiketmi v Jire. K filtrovaniu pomocou JQL sa dostaneme cestou: požiadavky → Hľadať požiadavky → Rozšírené. Do dialógového okna potom následne píšeme požadovaný dotaz. Po napísaní dotazu stlačíme tlačidlo „Hľadať“. Takto zobrazený filter môžeme následne uložiť za pomoci tlačidla „Uložiť ako“, kde zadáme názov filtra a potvrdíme. V našom prípade, budeme využívať filter, ktorý filtruje aktuálne nevyriešené požiadavky používateľa, ktorý filter zobrazuje a neobsahuje uvedený štítok. Dotaz vyzerá nasledovne:

assignee = currentUser() AND statusCategory in ("In Progress", "To Do") AND labels not in (Intern_Timelog)

Na to, aby filter fungoval viacerým používateľom je potrebné aby im bol filter zdieľaný. Pre zdieľanie filtra je potrebné ísť na požiadavky → spravovať filtre následne nájsť uložený filter, ktorý chceme zdieľať kliknúť na ikonu nastavenia → upraviť a pridať používateľov alebo skupiny používateľov, ktorým chceme filter zdieľať. [23] Výsledok tohto filtra môžeme vidieť v MS Teams. Intefrácia komunikuje na leveli aplikačného rozhrania.

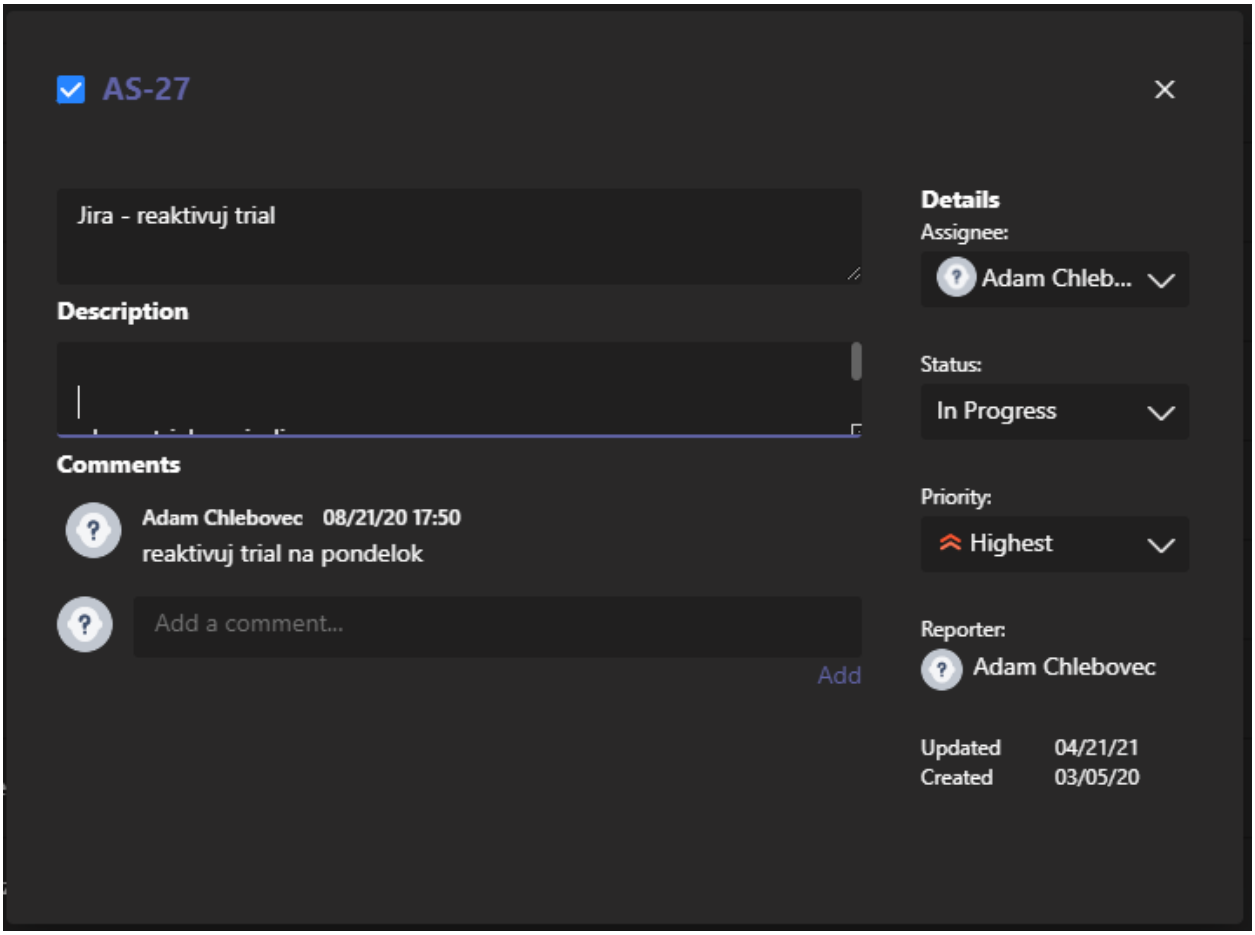


Obrázok 17 Zobrazenie vybraného filtra v MS Teams

Po vybraní požiadavky z vyfiltrovaného zoznamu sa nám zobrazí editovateľný náhľad. V tomto náhľade môžeme editovať vybrané polia rovnako tak ako pridať komentár. Polia v náhľade sú bohužiaľ iba základné informácie a nedajú sa nastaviť v rámci nastavenia obrazovky v Jire. Polia, ktoré sa zobrazujú v náhľade sú:

- Sumár
- Popis
- Komentáre
- Riešiteľ
- Stav
- Priorita
- Zadávateľ

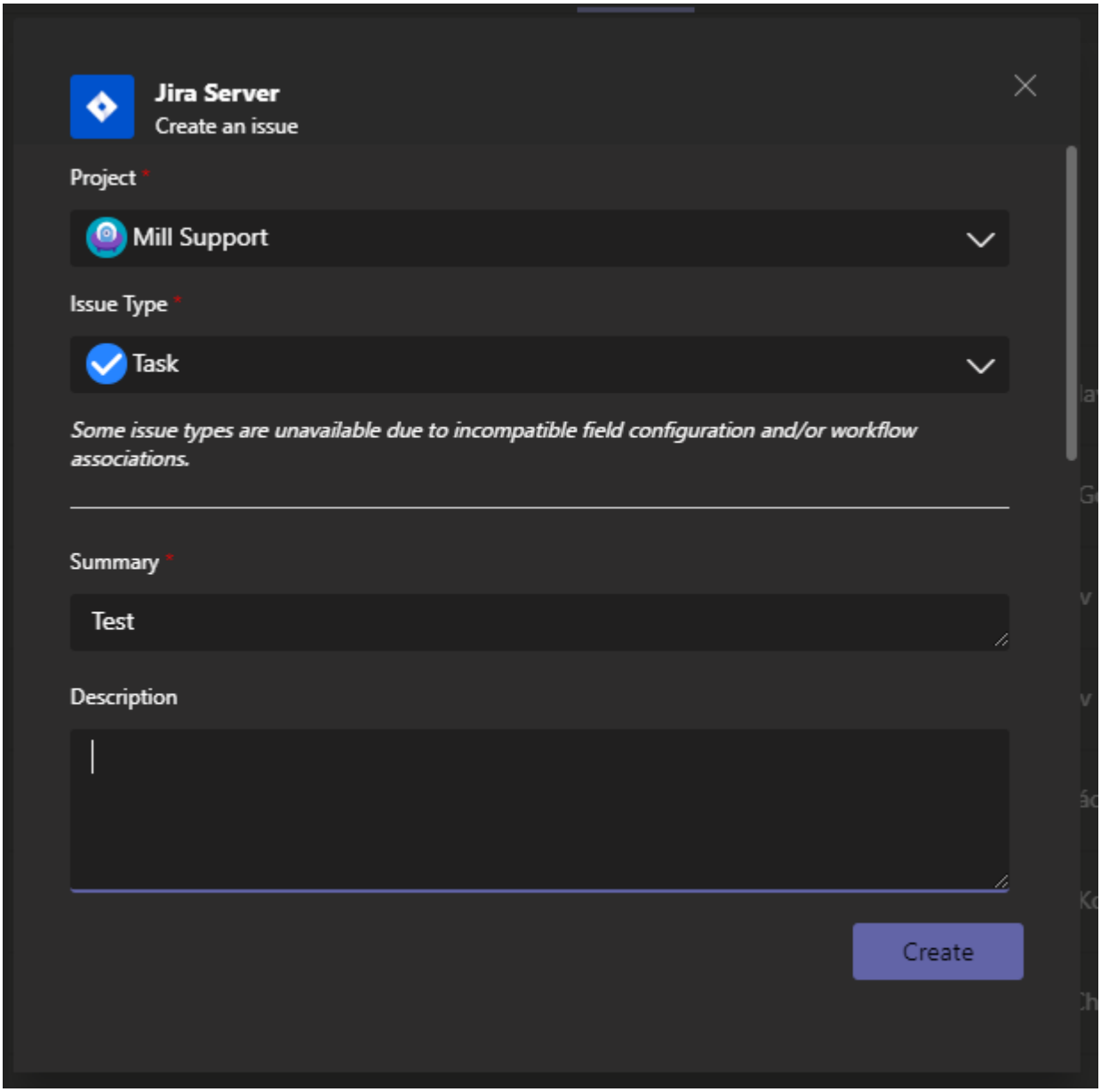
- Dátum poslednej zmeny
- Dátum vytvorenia požiadavky



Obrázok 18 Rozšírený náhľad požiadavky v MS Teams

Okrem zdieľania zobrazovania a pracovania s už vytvorenými požiadavkami aplikácia poskytuje aj možnosť požiadavky prostredníctvom MS Teams vytvoriť. Vytvorenie novej požiadavky je možné dvoma spôsobmi:

a. Vytvorenie novej prázdnej požiadavky pomocou kliknutia na aplikáciu Jira Server následne kliknúť na tlačidlo „+“ a zvoliť vytvoriť požiadavku. Takto sa nám zobrazí prázdny formulár na vytvorenie požiadavky aký je môžeme vidieť na obrázku nižšie. Na rozdiel od náhľadu požiadavky, kde nemôžeme ovplyvniť zobrazené polia nastavením obrazovky v Jire pri vytváraní novej požiadavky sa nám zobrazí obrazovka nastavená pri vytváraní požiadavky podľa projektu.



Obrázok 19 Vytvorenie novej požiadavky prostredníctvom MS Teams

b. Vytvorenie novej požiadavky zo správy. Podobne ako pri vytvorení komentára zo správy ideme na pole správy a v pravom hornom rohu zvoliť tlačidlo „...“. Po stlačení tohto tlačidla sa nám zobrazí zoznam akcií kde vyberieme ďalšie možnosti a vytvoriť požiadavku. Pri vytvorení požiadavky sa nám zobrazí rovnaké dialógové okno ako v prvom prípade s rozdielom, že pole popis obsahuje obsah správy a pole zadávateľ je odosielateľ vybranej správy.

Výhody

- Využíva rozpoznávanie používateľov za pomoci už nastavenej integrácie s AAD
- Široká funkcionalita a možnosť personalizácie filtrov či obrazoviek pri vytváraní požiadaviek

Nevýhody

- Pomalšie reagovanie aplikácie pri práci oproti práci priamo v Jire
- Nutnosť manuálne pridávať aplikáciu do každého vlákna v MS Teams

3.5 Integrácia Jira Software/Jira Service Desk s mail serverom Microsoft Exchange

Pri integrácií Jiry s mailom zvolíme dve riešenia. Prvá integrácia sa zaoberá integráciou s Jira Service Desk aplikáciou, kde využijeme možnosť konfigurácie mailu ako kanálu pre komunikáciu so zákazníkom. Pre prípad Jira Software využijeme aplikáciu „Adaptavist ScriptRunner for JIRA“. Dôvod, prečo pristupujeme k dvom riešeniam sú licencie a teda potreba licencie zvlášť pre Jira Service Desk a zvlášť pre Jira Software. V oboch prípadoch ide o integráciu s dátami na dátovom leveli.

Jira Service Desk

Pre integráciu Service Desku s mailovým serverom využívame priamo funkcionalitu zabudovanú v rámci nástroja. Pri nastavení automatického vytvorenia tiketu v Jira Service Desk aplikácií postupujeme nasledovne: [24]

1. Nastavenie vykonávame s účtom, ktorý má práva globálneho administrátora
2. V nastaveniach projektu, pre ktorý chceme nastaviť integráciu je potrebné vytvoriť taký typ požiadavky, ktorý bude obsahovať polia sumár a popis. Ak sa v type požiadavky vyskytujú aj iné polia je potrebné aby boli tieto polia označené ako nepovinné
3. V projekte zvolíme „Nastavenia projektu“ a „E-mailové požiadavky“
4. Zvolíme „Pridať e-mailovú adresu“
5. Vyplníme prístupové údaje pre mailovú adresu a mailový server
6. Pri vypĺňaní podrobností o e-mailovom účte vyberieme v poli E-mailový protokol možnosť IMAP. Rozdiel, medzi IMAP. Rozdiel medzi POP a IMAP pri tomto nastavení je ten, že IMAP po spracovaní e-mailu tento e-mail označí ako prečítaný, zatiaľ čo POP e-mail po spracovaní vymaže.

7. Po kliknutí na tlačidlo „Ďalej“ vyberieme typ požiadavky, ktorý sa nám pri spracovaní mailu vytvorí a stlačíme „Hotovo“

Pre vytvorenie požiadavky z e-mailu je potrebné, aby zasielateľ e-mailu bol zaevidovaný medzi zákazníkmi v projekte, prípadne bol projekt nastavený tak, že požiadavku môže zadať ktokoľvek.

Jira Software

Ako sme už spomenuli pre vytváranie požiadaviek z e-mailu v prípade Jira Software použijeme aplikáciu Adaptavist ScriptRunner for JIRA. Túto aplikáciu sme vybrali kvôli jej širokému využitiu a teda v záujme ušetrenia počtu potrebných licencií využijeme funkcionalitu tejto aplikácie. Ako prvé je teda opäť potrebné nainštalovať aplikáciu z Atlassian obchodu. Po inštalácií je potrebné nakonfigurovať spracovanie e-mailov: [25]

1. Prihlásený pod účtom s globálnymi administrátorskými právami vyberieme v Jire „Administrácia“ a „Systém“
2. V podstránke systém nájdeme v ľavom paneli možnosť „Prichádzajúca pošta“
3. V prichádzajúcej pošte najprv klikneme na „Pridať poštový server POP/IMAP
    a. Vyplníme meno servera a popis načo server slúži
    b. Tak ako v prípade pri nastavovaní Jira Service Desk vyberieme protokol IMAP čo zapríčiní, že sa nám maily nebudú po spracovaní mazať ale označia sa ako prečítané
    c. Vyplníme prihlasovacie údaje používateľa, ktorý ma prístup k danému poštovému serveru a stlačíme tlačidlo „Pridať“
4. Po pridaní poštového serveru klikneme na tlačidlo „pridať obsluhu prichádzajúcej pošty“
    a. Vyplníme meno
    b. V poli server vyberieme možnosť serveru, ktorý sme pridali v predchádzajúcom bode
    c. Ako obslužný program vyberieme „ScriptRunner Mail Handler“
    d. Do poľa názov priečinka vložíme názov priečinka, z ktorého chceme spracovávať prichádzajúce e-maily. V prípade, že tento priečinok necháme

prázdny, maily sa budú spracovávať z hlavného priečinku doručenej pošty. Stlačíme tlačidlo „Ďalší“

5. V prípade, že chceme spracovávať maily iba od určitého odosielateľa môžeme v poli zachytené e-maily zadať konkrétny mail, z ktorého sa budú maily spracovávať do požiadaviek.
6. V prípade, že vytvorenie požiadavky zlyhá, môžeme nastaviť mailovú adresu na ktorú sa zašle notifikácia o zlyhaní
7. Ak zaškrtneme pole vytvoriť používateľa, Jira automaticky z každej správy, ktorá prišla z nezaevidovanej e-mailovej adresy vytvorí nového používateľa, túto funkciu my nevyužijeme
8. Do poľa skript vložíme nasledujúci Groovy skript:

```
import com.atlassian.jira.component.ComponentAccessor

import com.atlassian.jira.service.util.ServiceUtils

import com.atlassian.jira.service.util.handler.MessageUserProcessor

import com.atlassian.jira.user.ApplicationUser

import com.atlassian.jira.user.util.UserManager

import com.atlassian.mail.MailUtils


def userManager = ComponentAccessor.getComponent(UserManager)

def projectManager = ComponentAccessor.getProjectManager()

def issueFactory = ComponentAccessor.getIssueFactory()

def messageUserProcessor =
ComponentAccessor.getComponent(MessageUserProcessor)


def subject = message.getSubject() as String

def issue = ServiceUtils.findIssueObjectInString(subject)
```

```
if (issue) {
   return
}

ApplicationUser user = userManager.getUserByName("achlebovec@***.sk")
def project = projectManager.getProjectObjByKey("AS")

def issueObject = issueFactory.getIssue()
issueObject.setProjectObject(project)
issueObject.setSummary(subject)
issueObject.setDescription(MailUtils.getBody(message))
issueObject.setIssueTypeId(project.issueTypes.find { it.name == "Task" }.id)
issueObject.setReporter(reporter)

messageHandlerContext.createIssue(user, issueObject)
```

9. Stlačíme tlačidlo „Uložiť“

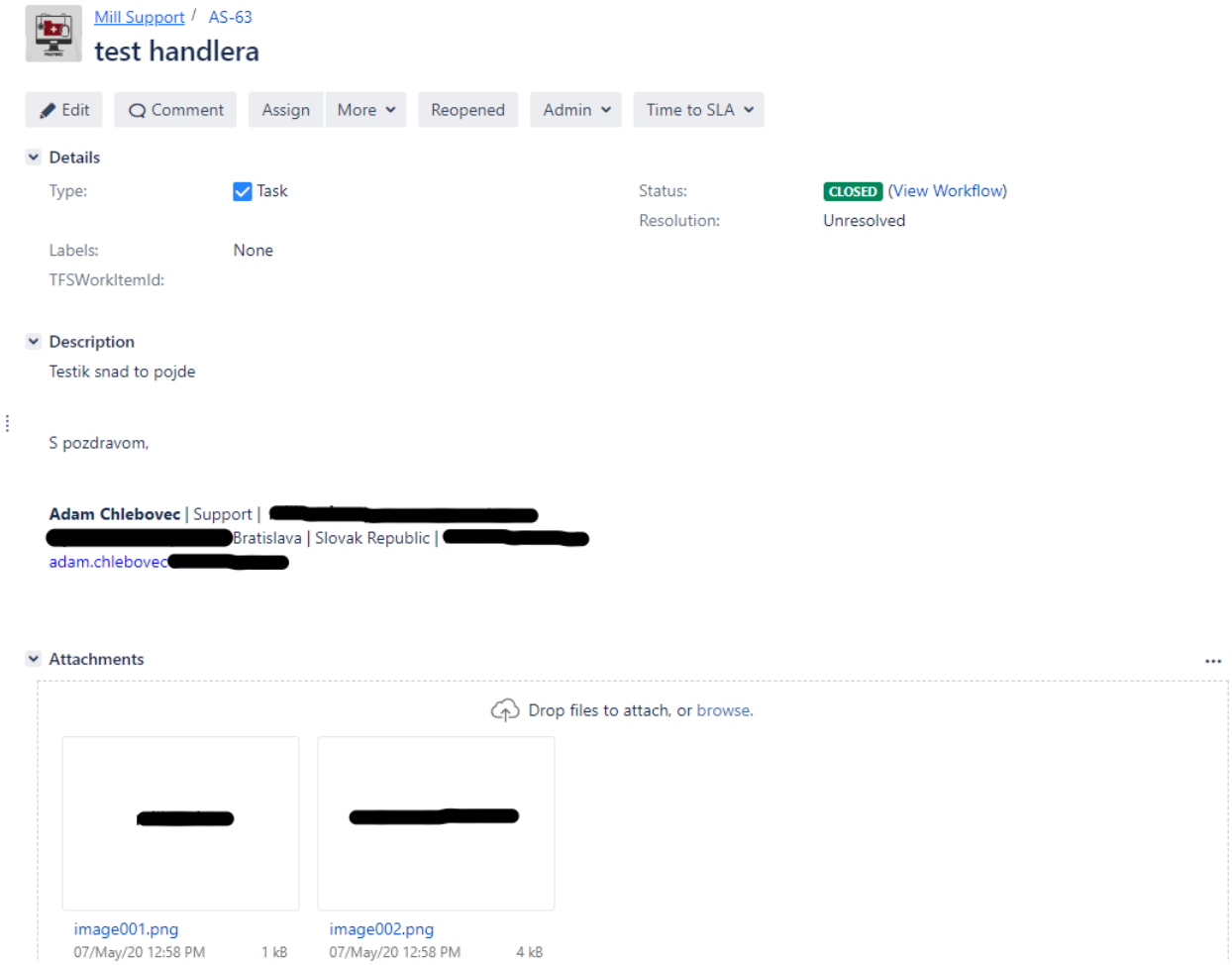
Skript je napísaný pomocou jazyka Apache Groovy. Apache Groovy je mnohostranný skriptovací jazyk pre Jira platformu. Ide o výkonný a dynamický jazyk s možnosťou statickej kompilácie. Jazyk je vytvorený pre zlepšenie produktivity známy vďaka ľahkej syntaxi. Je ľahko integrovateľný s Java aplikáciami a poskytuje Java aplikácií skvelé funkcia ako tvorba jazyku špeciálneho pre doménu či funkčné programovanie.

Jira Software

Za pomoci skriptu sme vykonali nasledujúce nastavenia:

- Predmet sa uloží ako Sumár správy
- Telo správy sa nahrá do popisu
- Zadali sme projekt podľa kľúča, do ktorého sa požiadavka vytvorí
- Zadali sme typ vytvorenej požiadavky: „Task“
- Nastavili sme používateľa s právami, pod ktorého právami sa skript vykonáva

Prílohy a obrázky sa aj v prípade nahratia do popisu automaticky nahrajú ako príloha. Náhľad vytvorenej požiadavky môžeme vidieť na nasledujúcom obrázku.



Obrázok 20 Náhľad na požiadavku vytvorenú pomocou e-mailu v Jira Software projekte

Výhody

- Možnosť rozšíriť skript o ďalšiu funkcionalitu ako napríklad vyplnenie iných polí
- Aplikácia okrem integrácie poskytuje množstvo ďalších funkcií v oblasti automatizácie

Nevýhody

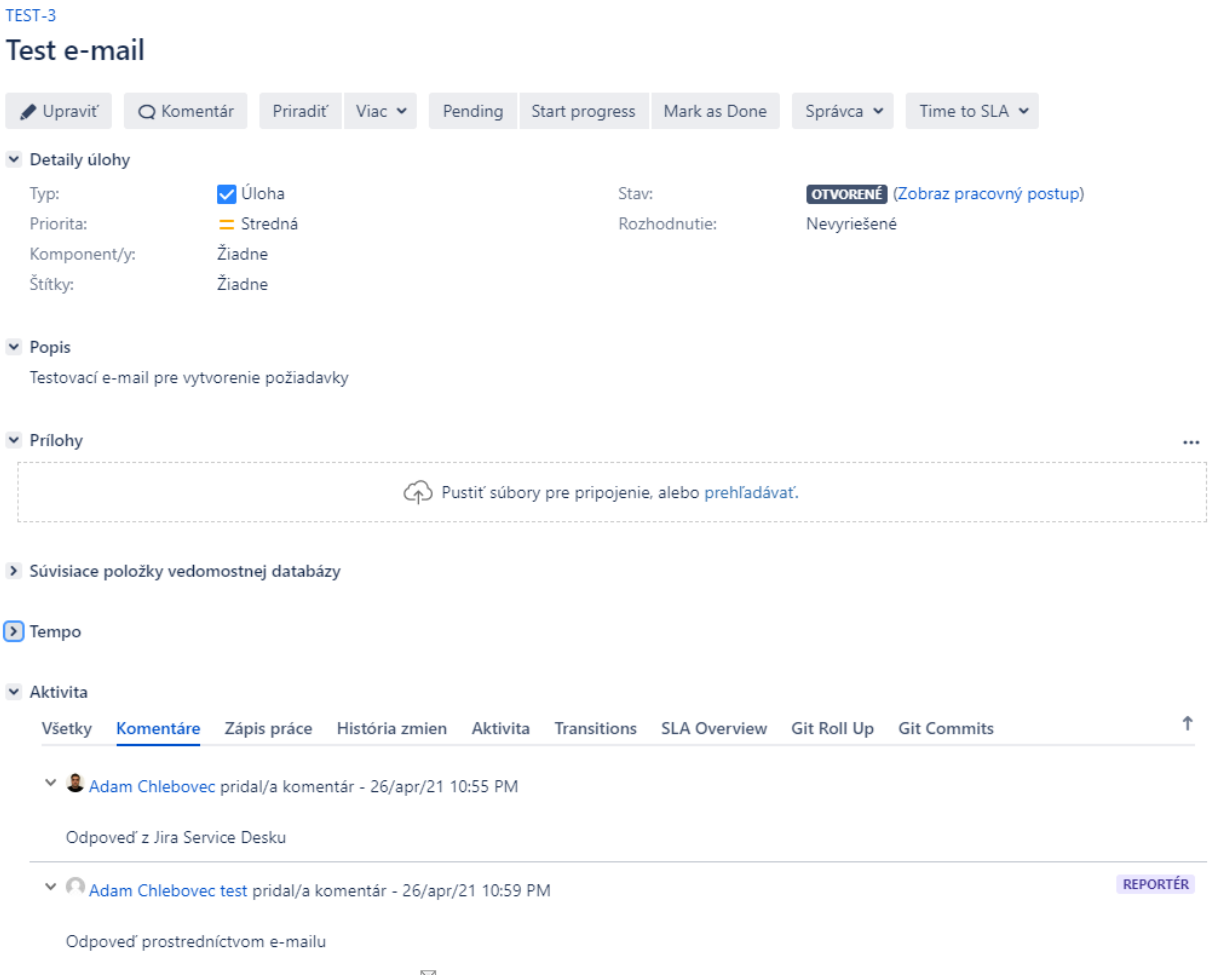
- Nie je možné komunikovať ako v prípade Service desku
- Problém pri filtrovaní mailov, z ktorých sa má vytvoriť požiadavka. Požiadavka sa tak vytvorí napríklad presunutím správy do konkrétnej zložky, z ktorej sa následne vytvorí požiadavka

Service Desk

Po nastavení vytvárania požiadaviek z prijatých mailov je funkcionalita nasledovná:

1. V prípade, že na zadanú mailovú adresu príde správa z e-mailovej adresy, ktorá je zaznamenaná medzi zákazníkmi v rámci Jira Service Desk projektu, vytvorí sa nová požiadavka do nastaveného projektu. Požiadavka sa vytvorí tak, že predmet správy je Sumár a telo správy je popis požiadavky. Prípadné prílohy sa prirodzene dajú ako príloha.
2. Ak tejto požiadavke pridáme komentár, zákazníkovi je odoslaný mail s obsahom komentáru.
3. Ak zákazník na túto správu následne odpíše, nevytvorí sa nová požiadavka, ale táto odpoveď sa pridá ako komentár. Ak odpoveď obsahuje prílohy, príloha je pridaná medzi ostatné prílohy požiadavky. Pre dodržanie tejto funkcionality je dôležité, aby zákazník pri odpovedi nemenil názov predmetu správy.

Náhľad takto vytvoreného tiketu môžeme vidieť na nasledujúcom obrázku.



Obrázok 21 Náhľad na požiadavku vytvorenú pomocou e-mailu v Jira Service Desk projekte

Výhody

- Je možné informovať zákazníka o priebehu riešenia jeho požiadavky
- Je možné zapnúť funkciu zbierania spätnej väzby, kde po uzavretí požiadavky príde zákazníkovi dotazník s prieskumom spokojnosti.

Nevýhody

- Niektoré mailové aplikácie zasielajú odpoveď, ktorá obsahuje celú predošlú konverzáciu. Táto konverzácia sa potom celá opakovanie ukladá v komentároch

ZÁVER

Cieľom práce bolo integrovať Jira platformu do spoločnosti, ktorá využíva čisto Microsoft technológie za víziou zefektívnenia pracovnej činnosti, komunikácie a reportovania. Pre uskutočnenie tejto integrácie sme si najprv predstavili jednotlivé aplikácie a ich využitie v rámci podniku, následne sme stanovili jednotlivé ciele, ktoré sme chceli integráciou dosiahnuť. Proces integrovania sme rozdelili do troch častí. V prvej časti sme vybrali nástroj (aplikáciu), ktorá nám integráciu zabezpečuje. Po výbere integračnej aplikácie sme aplikáciu nainštalovali a na základe analýzy požiadaviek nastavili jednotlivé parametre. Po úspešnej inštalácií a konfigurácií sme integráciu manuálne otestovali na základe očakávaných používateľských scenárov.

Tieto scenáre sa nám podarilo do značnej miery naplniť. V prípade autentifikácie máme funkčný spôsob prihlásenia sa za pomoci doménových prihlasovacích údajov uložených v Azure Active Directory. Metóda tejto autentifikácie je efektívna a stále umožňuje aj klasický spôsob autentifikácie poskytovaný priamo Jirou. Miernou nepríjemnosťou je nemožnosť pridania autentifikácie cez Jira Service Desk portál na zadanie interných požiadaviek na Support. V prípade, že je ale používateľ prihlásený cez Jira Software nie je ďalšia autentifikácia potrebná.

Pri integrácií Jery s Azure DevOps Boards sme najprv spravili analýzu na strane momentálne aktívne využívaného Azure DevOps Boards na základe ktorej sme vytvorili v Jire potrebnú konfiguráciu. Po konfigurácií samotnej integrácie sa nám podarila plne funkčná integrácia zabezpečujúca obojstrannú synchronizáciu tiketov bez závislosti na fakte, v ktorom nástroji bol tiket vytvorený alebo zmenený. Pri testovaní funkčnosti zmien v oboch nástrojoch sme však narazili na fakt, že všetky zmeny vykonané na strane nástroja, ktorý nebol pôvodne využitý na zmenu bol vykonaný v mene používateľského účtu zadaného pri konfigurácií integračného nástroja. Táto skutočnosť do istej miery znečisťuje dáta v oblasti vykazovania odvedenej práce, čo spôsobuje problémy pri tvorbe reportov na strane Jiry.

Integrácia Jiry s Git nástrojom na správu verzií súborov sme dosiahli presne očakávaný výsledok, kde sa nám podarilo napojiť Git repozitár do prostredia Jiry. Vďaka tomu je možné priamo cez Jiru sledovať jednotlivé repozitáre, zmeny stavov súborov a pridelovať jednotlivé zmeny k požiadavkám v Jire.

Integrácia Jiry s MS Teams výrazne pomáha zlepšiť komunikáciu pri práci. Nakoľko je MS Teams primárny nástroj na komunikáciu v rámci podniku, všetky požadované funkcionality sú splnené. Jediný mierny nedostatok nástroja je pomalšia odozva pri práci s požiadavkami. To je však akceptovateľné.

Posledná integrácia týkajúca sa integrovania Jira Service Desk a MS Exchange dopadla presne podľa očakávanie. Nakoľko ide o priamu funkcionalitu Jira Service Desku išlo o rýchlu konfiguráciu, ktorá funguje presne podľa požiadaviek. Jediná chyba krásy je, že ak zákazník Service Desku zašle v odpovedi pripnutú celú konverzáciu, tak sa celá táto konverzácia dokola nahráva do komentárov. Pri integrácií s Jira Software sme nemali veľké očakávania kde sme chceli dosiahnuť iba jednoduché vytvorenie tiketu na základe prijatého mailu bez nutnosti žiadnej ďalšej konverzácie so zasielateľom mailu. Táto skutočnosť sa nám podarila za využitia funkcionality nástroja, ktorý už v Jire používame a tak nebolo potrebné kupovať licenciu navyše.

Na záver môžeme skonštatovať, že ciele práce sa nám podarilo naplniť s jediným nedostatkom, kde sa nám v rámci integrácie Jiry a Azure DevOps Boards nepodarilo udržať informácie o vykazovaní. Nakoľko vykazovanie samotný Azure DevOps Boards podporuje len do veľmi malej miery, ktorá nezabezpečuje ani možnosť tvorby reportov sa táto skutočnosť dala očakávať a teda nemôžeme to považovať za chybu integrácie. Táto práca teda vo finálnom dôsledku bude slúžiť ako dokumentácia k zavedenému riešeniu v rámci podniku. Forma, akou bola práca spísaná je neutrálna, neobsahuje žiadne detaily procesov ani špeciálne nastavenia, ktoré by boli typické iba pre vybraný podnik. Na základe tejto skutočnosti je práca vhodná aj ako inšpirácia pre iné podniky.

POUŽITÁ LITERATÚRA

[1] T. Bruckner, J. Voříšek, A. Buchalcevová a kolektív, Tvorba informačních systémů, Praha: Grada Publishing, a.s., 2012.

[2] M. Grell, Informačné systémy v národnom hospodárstve, Bratislava: EKONÓM, 2005.

[3] IBM Cloud Education, „Application Integration," IBM, 14 April 2020. [Online]. Available: https://www.ibm.com/cloud/learn/application-integration. [Cit. April 2021].

[4] D. S. Linthicum, Enterprise Application Integration, Addison-Wesley, 2000.

[5] K. Murphy, „What You Need to Know About Application Integration," Planergy, 12 April 2019. [Online]. Available: https://planergy.com/blog/application-integration/. [Cit. Máj 2021].

[6] S. Balaji a M. S. Murugaiyan, „WATEERFALLVs V-MODEL Vs AGILE: A COMPARATIVE STUDY ON SDLC," *International Journal of Information Technology and Business Management,* zv. 2, %1. vyd.1, pp. 26-30, 2012.

[7] D. Knapp, The ITSM Process Design Guide: Developing, Reengineering, and Improving IT Service Management, Plantation: J. Ross Publishing, 2010.

[8] S. Chand, „edureka.co," Brain4ce Education Solutions Pvt. Ltd., 22 October 2019. [Online]. Available: https://www.edureka.co/blog/itil-v3-vs-itil-v4/. [Cit. 17 April 2021].

[9] M. Raza, „www.bmc.com," BMC Software, Inc, 3 March 2020. [Online]. Available: https://www.bmc.com/blogs/cobit-2019-vs-cobit-5/. [Cit. 20 March 2021].

[10] Millennium, spol. s r. o., „millennium.sk," Millennium, spol. s r. o., 2021. [Online]. Available: https://www.millennium.sk/. [Cit. April 2021].

[11] Miscrosoft, „What is Azure Active Directory?," 06 May 2020. [Online]. Available: https://docs.microsoft.com/en-us/azure/active-directory/fundamentals/active-directory-whatis. [Cit. March 2021].

[12] Microsoft, „What features and services do I get with Azure DevOps?," Microsoft, 11 November 2020. [Online]. Available: https://docs.microsoft.com/en-us/azure/devops/user-guide/services?view=azure-devops. [Cit. 2021].

[13] Microsoft, „What is Azure Boards?," Microsoft, 7 September 2020. [Online]. Available: https://docs.microsoft.com/en-us/azure/devops/boards/get-started/what-is-azure-boards?view=azure-devops&tabs=agile-process. [Cit. April 2021].

[14] S. Chacon a B. Straub, Pro Git, 2nd Edition, Springer Nature, 2014.

[15] M. Louis a D. Tapp, „Teaching with Teams: An introduction to teaching an undergraduate law module using Microsoft Teams,“ *Innovative Practice in Higher Education,* zv. 3, %1. vyd.3, pp. 58-66, 2019.

[16] Atlassian, „What is Jira used for?,“ Atlassian, [Online]. Available: https://www.atlassian.com/software/jira/guides/use-cases/what-is-jira-used-for#jira-for-software-development-teams. [Cit. April 2021].

[17] Atlassian, „Installing Marketplace apps,“ Atlassian, 4 Feb 2021. [Online]. Available: https://confluence.atlassian.com/upm/installing-marketplace-apps-273875715.html. [Cit. March 2021].

[18] Microsoft, „Tutorial: Azure Active Directory single sign-on (SSO) integration with JIRA SAML SSO by Microsoft,“ Microsoft, 28 Dec 2020. [Online]. Available: https://docs.microsoft.com/en-us/azure/active-directory/saas-apps/jiramicrosoft-tutorial. [Cit. April 2021].

[19] M. Żyromski, „TFS4JIRA Documentation,“ Spartez Software development, 18 Dec 2019. [Online]. Available: https://confluence.spartez-software.com/display/TFS4JIRA/TFS4JIRA+Documentation+Home. [Cit. Feb 2021].

[20] Atlassian, „Add a custom field to a screen,“ Atlassian, 31 Jan 2020. [Online]. Available: https://support.atlassian.com/jira-cloud-administration/docs/create-a-custom-field/. [Cit. April 2021].

[21] Atlassian, „Customize the issues in a project,“ Atlassian, 20 April 2021. [Online]. Available: https://support.atlassian.com/jira-software-cloud/docs/customize-the-issues-in-a-project/. [Cit. April 2021].

[22] SoftServe Inc., „Jira Server for Microsoft Teams Help,“ 2019. [Online]. Available: https://www.msteams-atlassian.com/JiraServer/#installing-into-teams. [Cit. April 2021].

[23] Atlassian, „Get started with Advanced Search and JQL,“ [Online]. Available: https://www.atlassian.com/software/jira/guides/expand-jira/jql#advanced-search. [Cit. April 2021].

[24] Atlassian, „Receiving requests by email,“ Atlassian, 11 Feb 2021. [Online]. Available: https://confluence.atlassian.com/servicemanagementserver/receiving-requests-by-email-939926303.html. [Cit. April 2021].

[25] Adaptavist, [Online]. Available: https://docs.adaptavist.com/sr4js/6.23.0/features/mail-handler. [Cit. March 2021].