Semestrální práce ke kurzu 4IT421 Zlepšování procesů budování IS	
Semestr	ZS 2018
Autoři	Nikolas Charalambidis, chan01 Denisa Tomanová, tomd03 Dagmar Žeravíková, zerd00
Téma	Adopting Continuous Delivery at teamplay, Siemens Healthineers
Datum odevzdání	21.12.2018

Abstrakt

Předmětem této semestrální práce je seznámit čtenáře s transformací současného systému společnosti Siemens Healthineers směrem k Continuous Delivery, které by řešilo automatizaci dodávání releasů aplikací založených na big data. V semestrální práci je základní popis a seznámení s Continuous Delivery včetně výhod a důležitých milníků, které je třeba splnit. Dalším cílem práce je krátké představení společnosti Siemens Healthineers, jejíž doménou je regulované lékařství a jeho transformace, výzkum, digitalizace a expanze.

Klíčová slova

Continuous Delivery, Continuous Integration, Siemens Healthineers, teamplay

Obsah

1. Úvod

Přístupů k vývoji softwaru existuje několik. Do popředí se čím dál častěji dostávají požadavky zákazníků, které se velmi často mění a je potřeba na tyto požadavky rychle reagovat. Efektivní vývoj, opakované nasazování či automatizace jsou zásady, které naplňují Continuous Delivery. Průkopníky těchto zásad jsou Jez Humble and David Farley, kteří ve své knize uvádí: „*Through automation of the build, deployment, and testing process, and improved collaboration between developers, testers, and operations, delivery teams can get changes released in a matter of hours – sometimes even minutes, no matter what the size of a project or the complexity of its code base.*" (Humble a Farley, 2010)

Hlavním cílem této semestrální práce je představit čtenáři, jak firma Siemens Healthineers začala transformovat svůj dosavadní systém směrem k Continuous Delivery, které řeší automatizaci dodávání releasů aplikací založených na big datech.

Na začátku práce je základní seznámení s Continuous Integration a Continuous Delivery, včetně výhod a důležitých milníků, které je třeba splnit.

V třetí kapitole je krátce představena firma Siemens Healthineers, která se zaměřuje na oblast zdravotní péče, digitalizaci či výzkum. Dále je zde představeno současné postavení firmy na trhu a jejich platforma teamplay.

Ve čtvrté kapitole je uvedeno, jak firma postupovala k adopci Continuous Delivery a co bylo jejich cílem. Také je zde popsán model pro CD včetně podpůrných principů založených na agilním vývoji, jako je například Behavior-Driven Development, Domain-Driven Design, Test-Driven Development, Párové programování a Team Pipeline.

2. Continuous Integration a Continuous Delivery

Continuous Integration (dále *CI*), neboli kontinuální integrace je princip tvorby softwarového produktu, jenž zajišťuje neustálé jednotkové a integrační testování po určitých přírůstcích ve zdrojovém kódu. Tato technika je v dnešní době často zaváděná. Díky vysoké úspěšnosti a velké škále různých nástrojů a možností integrace s verzovacími systémy, servery k nasazení a nástrojů statistických analýz atd., se stává standardem. Cílem této kapitoly bude v krátkosti objasnit pojem CI a porovnat ji s Continuous Delivery, jenž je určité rozšíření CI o automatizaci dalších fází vývoje.

2.1. Continuous Integration

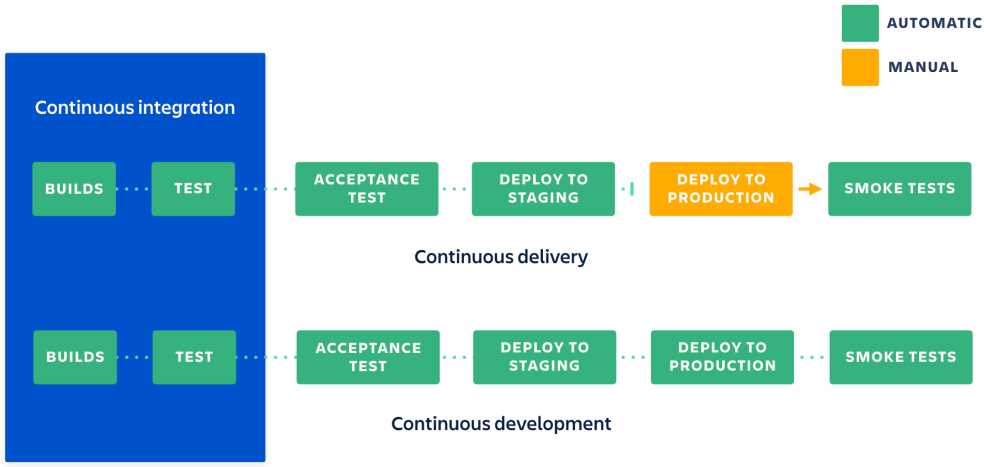
Volně přeložená definice je dle Gartnerovského IT slovníku následující:
„*Continuous integration (CI) systémy poskytují automatizaci buildu softwaru a validaci a kontroly výsledků postupně běžících nakonfigurovaných sekvencí procesů po každé změně odeslané do repozitáře se zdrojovým kódem. Tyto aktivity jsou v úzké vazbě s praktikami agilního vývoje za podpory DevOps nástrojů.*" (Gartner, 2018)

Podobnou definici nabízí i Martin Fowler (Fowler, 2006), kde popisuje CI na frekventovanou integraci více lidmi na denní bázi. Jednoduše se jedná o integraci přírůstků jednotlivých vývojářů do jedné společné základny.

2.2. Continuous Delivery

Continuous Delivery (dále *CD*) je rozšířením CI o několik prvků automatizace. Obrázek od Atlassianu představuje grafické znázornění vztahu mezi CI a CD a také Continuous Delivery, což je pojem pro další princip vývoje softwarového produktu založený na podobném fungování jako CD, jenž se právě s CD z velké části překrývá a je definovaný mnoha různými definicemi.

Následující obrázek popisuje rozdíl mezi CI a CD (Pittet, 2018):



Obrázek 1 - Rozdíl mezi CI a CD (Pittet, 2018)

Rozdíly jsou v následujících bodech:

- Nasazení na testovací a stage prostředí – Jedná se o rozšíření pipeliny o nasazení otestovaného přírůstku na určité neprodukční prostředí dle určitých pravidel.
- Běh automatizovaných testů – Jelikož je fungující aplikace nebo služba s novým otestovaným přírůstkem nasazená, tak je možné pomocí automatizovaných testů otestovat její funkčnost ve snaze simulovat její chování jako nasaditelný celek ještě před jejím poskytnutí do produkčního prostředí.
- Nasazení na produkční prostředí – Tento bod je pro Continuous Delivery a Continuous Deployment identický, nicméně liší se jen ve způsobu dodání. CD narozdíl od Continuous Deployment nemá nastavenou pipeline na automatické nasazení na produkční prostředí a je nutné jej nastavit manuálně z toho důvodu, že je vyžadována i kontrola člověkem, který zajistí bezpečné nasazení.
- Smoke testy – Tyto testy kopírují předchozí postup nasazení a automatických testů. Zde se jedná o smoke testy, které mají v rychlosti prověřit integritu systému, funkčnost integrací a systému jako celku a určit, zda je produkt

vhodný k následnému testování, například manuálnímu.

2.3. Nástroje CI a CD

Neexistují nástroje, které by plně podporovaly všechny články CI a CD včetně vývoje, testování a nasazení. Síla tohoto principu spočívá v integraci mnoha specializovaných nástrojů a pomocí CI pipeline je propojit a parametrizovat. Příkladem může být open-source nástroj Jenkins, který je schopný propojit verzovací systém a na základě identifikace vývojové větve spustit nakonfigurovanou pipeline a spustit jednotkové či integrační testy. Tyto nástroje disponují kompilačním prostředím, které je nutné pro chod těchto testů. Dále je možné vygenerované soubory odeslat, respektive za splněných podmínek nasadit na určitý server, jenž může být například v podobě cloudové služby, jako je Azure nebo Heroku. Mimo Jenkins je možné využít cloudové Continuous Integration služby jako je TravisCI a CircleCI, které mají tu výhodu, že jsou snadno integrovatelné s GitHubem a výsledkem open-sourcového světa.

3. Siemens Healthineers

Firma Siemens Healthineers je dceřiná firma společnosti Siemens AG. Původně se jmenovala Siemens Medical Solutions, od roku 2008 byla označována jako Siemens Healthcare a v květnu 2016 byla opět přejmenována na Siemens Healthineers. Byla založena roku 1847 v Berlíně jako menší rodinná firma. (Underconsideration, 2016) Nyní má přibližně 50 000 zaměstnanců po celém světě, avšak největší zastoupení má právě v Německu. (Siemens Healthineers, 2018b)

Obrázek 2 - Logo společnosti Siemens Healthineers (Siemens Healthineers, 2018a)

Siemens Healthineers je mezinárodní technologická firma, která si klade za cíl inovovat a formovat oblast zdravotní péče. Usiluje o digitalizaci a inovuje své portfolio produktů v oblastech laboratorní diagnostiky, lékařského snímání

(například magnetická rezonance, rentgenové zařízení), molekulární medicíny, pokročilých terapií a služeb. (Siemens Healthineers, 2018a)

Přibližně 240.000 pacientů se každou hodinu dostane do kontaktu se systémy firmy a víc jak 70 procent klinických rozhodnutí jsou ovlivněny technologiemi, které firma Siemens Healthineers poskytuje. Proto lze firmu označit jako jednu z největších dodavatelů v zdravotnickém průmyslu. (Siemens Healthineers, 2018a)

Co se týká ekonomické stránky, firma dosáhla ve fiskálním roce 2018 tržeb ve výši 13,4 miliardy EUR, upravený zisk ve výši 2,3 miliardy EUR. (Siemens Healthineers, 2018b)

3.1. Teamplay platforma

Velmi důležitým softwarovým produktem vývojového oddělení Siemens Healthineers, u jehož vývoje je vhodný zavedení praktik Continuous Delivery, je platforma teamplay.

Jedná se o platformu, jejíž účelem je digitalizace v oblasti zdravotní péče s cílem řízení výkonnosti zobrazovacích oddělení, jednotlivých zaměstnanců a díky registraci a vzájemnému připojení zobrazovacích radiofarmaceutických zařízení, jako je například RTG nebo tomograf, je možné díky získaným datům optimalizovat a do určité míry automatizovat provoz a podporovat vzájemnou komunikaci. Díky platformě teamplay je, v rámci nařízení a direktiv Evropské Unie, možné snížit náklady a redukovat množství škodlivých dávek, které pacient během svého vyšetření nebo léčby přijímá. Velkou výhodou této platformy je i přehlednost a přítomnost jednoho velkého customizovatelného dashboardu, jenž slouží nejen pracovníkům, ale i managementu. Cílem je poskytnout pacientům rychlejší, dostupnější a také mnohem individuálnější péči.

Výsledkem může být zhodnocení zařízení pro zdravotní péči Iatropolis v Řecku, které bylo nuceno využít této procesní a produktové inovace kvůli nepříznivé ekonomické situaci, rostoucího finančního a časového zatížení, kde díky této platformě dokázali redukovat množství radiofarmaceutických dávek přijímaných pacienty o 10-15 % za poslední 3 měsíce.

4. Zavedení Continuous Delivery v Siemens Healthineers

Tradiční způsob vývoje, který byl aplikován v rámci teamplay narážel na několik častých chyb. Nejzásadnějším omezením byla neschopnost týmu produkovat nové části aplikace v souladu s tím, jak organizace a software rostly na složitosti.

K odstranění těchto překážek se rozhodl tým pracující na teamplay zavést vývojové praktiky Continuous Delivery. Cílem tohoto zavedení bylo zvýšit kvalitu a četnost zpětné vazby, zlepšit spolupráci v rámci celé organizace a nastavit takový přístupu k vývoji, který by kladl za cíl maximální možnou kvalitu kódu a dodržování nastavených pravidel.

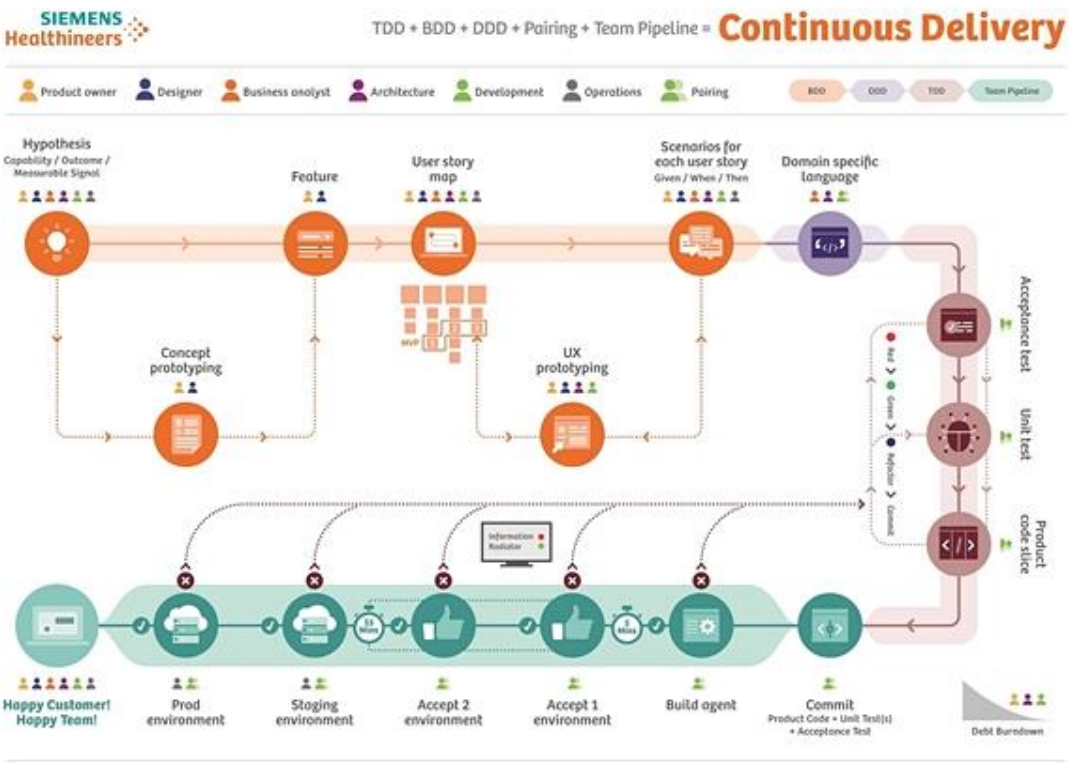
Tým se proto obrátil na Davida Farleyho, aby pomohl teamplay se změnou řady aspektů v jejich přístupu k softwarovému vývoji. Změna by měla zvýšit bezpečnost, stabilitu a efektivitu dosavadních procesů.

K dosažení tohoto cíle proběhla řada školení, tréninků a workshopů, ve kterých bylo přistupováno k této změně holistickým způsobem. Každý tým, který se podílí na teamplay, si prošel cvičením zaměřené na pochopení filozofie stojící za Continuous Delivery. To pomohlo k pochopení požadavků, které jsou základem k docílení požadované změny. (Ukis a Farley, 2018)

4.1. Proces adopce CI a CD

Výsledkem série workshopů a školení na téma Continuous Delivery, bylo vytvoření vlastního modelu CD pro teamplay platformu. Představuje způsob, jakým chce tým ze Siemens Healthineers definovat, testovat, zavádět a nasazovat změny na produktu.

Tento model je pak znázorněn na následujícím schéma.



Obrázek 3 - Model CD od firmy Siemens Healthineers (Ukis a Farley, 2018)

Na jeho samém vrcholu jsou vypsány metodologie, které budou užívány k naplnění Continuous Delivery, a to: Test-Driven Development, Behavior-Driven Development, Domain-Driven Design, Párové programování a Team Pipeline. Aplikováním všech těchto metodologií napříč všemi týmy bude tak dosaženo chtěného stavu, kdy je software vždy připravený k zveřejnění.

Schéma také zahrnuje role, které se podílí v jednotlivých krocích procesu. Jedná se o roli Product Ownera, Business Analytika, Designera, Vývojáře a Operačního inženýra. Barvy šipek a hlavních linií schéma reprezentují metodologii, kterou mají dané role využít.

Toto schéma je pro celou platformu velmi užitečné, neboť pomáhá všem členům týmu zorientovat se v rámci procesu a rychle pochopit, co mají dělat, aby podpořili hladký průběh Continuous Delivery.

Celý proces bude nyní demonstrován popsáním jednotlivých metodologií, z kterých proces sestává. (Ukis a Farley, 2018)

4.2. Behavior-Driven Development

Behavior-Driven Development je sada praktik, jejichž cílem je snížit množství nepotřebných chyb během vývoje software. Dalším cílem je také eliminovat komunikační propast mezi jednotlivými členy týmu a pečovat o co nejlepší pochopení zákazníků a jejich potřeb.

Velmi častým problém je, že se lidé v rámci organizace rozcházejí v pohledech na to, jak má software fungovat, a jaký problém vlastně software řeší. Avšak týmy, které pracují na principech BDD, se tomu snaží předejít. Záměrně hledají možná slepá místa, které přehlíží nebo ignorují, ještě předtím, než začne fáze vývoje. Tím jsou efektivnější a předchází zbytečným přepracováním a práci navíc.

Tým ze Siemens Healthineers v rámci BDD definuje řadu hypotéz, které jsou základem pro tvorbu každé nové části produktu. Pokud jsou hypotézy potvrzeny, je dál nápad rozpracován do jednoduchého prototypu, který je předložen i zákazníkovi, aby se ověřilo, zde rezonuje s jeho potřebami.

Poté následuje tvorba User Story mapy, která zachycuje, jakou cestou zákazník prochází při práci s určitými části produktu. Jednotlivé uživatelské cesty jsou dále rozšířeny o BDD scénáře, které jsou psány v Gherkin jazyce. Na tomto kroce se podílí celý tým od designéra po vývojáře, protože každý z nich se může na danou cestu dívat z jiné perspektivy a je vhodné jich zachytit co nejvíce. (Ukis a Farley, 2018)

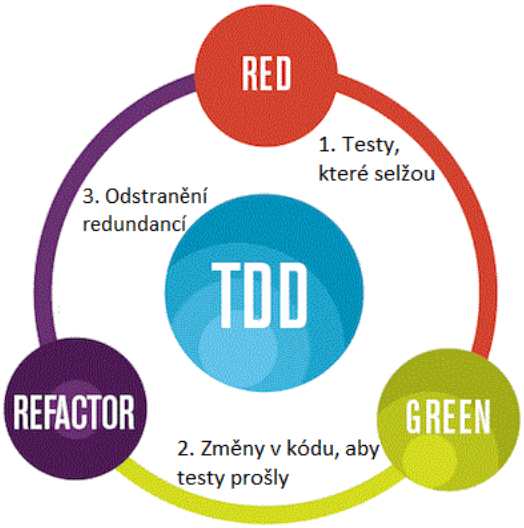
4.3. Test-Driven Development

Automatizované akceptační testy jsou nedílnou součástí Test-Driven Developmentu (zkráceně TDD). Dave Farley je popisuje jako „*testy, které vyhodnocují systém z pohledu externího uživatele v testovacím prostředí podobnému produkčnímu. Vytvářejí se ve formě spouštěcích specifikací pro definované chování systému.*" (Linders, 2017)

Akceptační testy chceme nasadit jednou, a přitom spustit všechny testy, proto by navzájem měly být nezávislé. Další charakteristikou akceptačních testů je opakovatelnost. Pokud spustíme vícekrát testovací případ, měl by pokaždé fungovat stejně. Při psaní testů by se měl vývojář soustředit na to „co" daný systém musí

udělat spíše než „jak" to systém dělá. Testy by měly být účinné a testovat každou změnu systému. (Farley, 2014) To znamená, že by neměly být příliš detailně napsané a využívat testovací data nikoliv produkční. Testovací data chceme jednoduché, aby umožnily se přesně zaměřit na chování systému, které testujeme. (Linders, 2017)

Jakmile jsou akceptační testy hotové a provedeny, jsou vyhodnoceny s chybami a červeně, protože samotný kód ještě nebyl napsán. TDD využívá postupu, kdy nejprve jsou napsány testy, které skončí s chybami. Poté děláme změny v kódu se samotnou funkcionalitou tak, aby testy prošly. Jde o postup od chyb, úspěšného spuštění, refaktorizace až po commit. (Ukis a Farley, 2018)



Obrázek 4 - Test-Driven Development cyklus (Lewandowski, 2017)

4.4. Párové programování a jeho váha

Párové programování je jedním z dvanácti pilířů metodiky XP, známé jako extrémní programování, jenž vyplývá z názvu, je technikou programování dvou lidí na jednom počítači. Jeho cílem je doručit mnohem kvalitnější kód za cenu pomalejší dodávky, jenž se promítne jako vyšší náklad. Nicméně díky kvalitnímu kódu tím eliminuje rizika výskytu chyb a technického dluhu v budoucnu a lze tuto techniku brát jako investici do budoucna (Wells, 1997).

Co je pro účely této práce důležitější, než podrobné představení této techniky je způsob, jakým Siemens Healthineers párové programování využívá a jeho přidaná hodnota. Dvojice programuje konceptem Driver-Navigator, kdy jeden programátor

píše kód (Driver) a druhý naviguje (Navigator), promýšlí alternativy, design, kontext a poskytuje zpětnou vazbu a korekci. Dvojice se po čase prohodí. Siemens Healthineers i v případě, že by se tato technika ekonomicky vyplatila méně, tak i přes to v ní vidí způsob, prostřednictvím kterého týmy mohou učit se a přijímat nejen nové techniky, procesy, technologie, které nejsou jednoduché k pochopení, používání, ale také posilovat týmovou kulturu a podporovat inovativní myšlení v souvislosti s celkovou kvalitou (Ukis a Farley, 2018).

4.5. Team pipeline

Siemens Healthineers využívají pěti-fázovou pipeline, kterou prochází kód, poté, co byl commitnutý.

Ihned poté je kód kontrolován Build agentem, který ověřuje, zda kód lze zkompilovat, zda procházejí všechny jednotkové testy vytvořené v rámci TDD. Cílem této části je získání okamžité zpětné vazby v rámci minut. Dle Ukis a Farley by toto mělo proběhnout do pěti minut. Je to tedy dostatečná doba na to, aby si vývojáři odpočinuli a zároveň nezačali pracovat na něčem jiném. Neprojde-li kód skrz tuto kontrolu, tj. například neprojde byť jediný test, celá změna je zamítnuta a kód není připuštěn dál.

Pokud je kód v pořádku, je nasazen do dalšího Akceptačního prostředí č. 1, kde jsou spuštěny akceptační testy. Je-li v pořádku i tato část, přechází se do fáze třetí, a to Akceptačního prostředí č. 2, kde probíhají integrační testy. Nyní se usiluje o to, aby byla maximálně eliminována potřeba procházet přes Akceptační prostředí č.2. To by znamenalo, že z prvního akceptačního prostředí by byl kód nasazen na staging a následně rovnou na produkci. (Ukis a Farley, 2018)

5. Závěr

Práce si kladla za cíl seznámit čtenáře s transformací současného systému firmy Siemens Healthineers směrem k Continuous Delivery. V první části práce byly obecně vysvětleny pojmy Continuous integration a Continuous delivery, včetně rozdílu. Následovalo krátké představení firmy a popis oblastí, kterými se firma zabývá. V neposlední řadě byla popsána transformace, její cíl a detailněji představen model Continuous Delivery vytvořený týmem Siemens Healthineers.

V závěru je potřeba si položit otázku. Jak se tato transformace systému směrem k Continuous Delivery firmě Siemens Healthineers povedla? Firma sama uvádí, že je teprve v začátcích této transformace. Různé týmy využívají model Continuous Delivery na různých úrovních, některým týmům to jde rychleji, některým pomaleji, a to převážně v závislosti na technickém a lidském faktoru. Každopádně již zaznamenali pozitivní výsledky. Například se jim podařilo v rámci týmu díky programování řízené testy a párovému programování nasadit novou funkcionalitu bez produkčních chyb, což se jim předtím nepovedlo. Téměř většina týmů nyní využívá user story mapping, které se jim zdá být užitečné. Offline i online formu využívají pro porozumění cílů, sdílení informací či zajištění větší transparentnosti při rozhodování členů týmu. V rámci transformace také většina týmů přešla od Scrumu ke Kanban, který se jim osvědčil a přináší jim lepší pracovní výkony. (Ukis a Farley, 2018)

Siemens Healthineers se snaží být experimentální firmou a ověřovat si tak své domněnky či tvrzení přímo na produkci. Myslíme si, že se jim to díky tomuto přechodu směrem k Continuous Delivery daří. Tato úspěšná transformace tak může být jakýmsi možným návodem pro ostatní firmy, které o této změně uvažují či ji plánují.

Použitá literatura

- FARLEY, Dave, 2014. *Acceptance Testing for Continuous Delivery*. [online]. [cit. 2018-12-12]. Dostupné z: https://pipelineconf.info/wp-content/uploads/2014/03/davefarley-acceptancetesting-forcontinuousdelivery2015.pdf
- FOWLER, M., 2006. *Continuous Integration* [online]. 01. 05. 2006 [cit. 2018-12-10]. Dostupné z: http://martinfowler.com/articles/continuousIntegration.html
- HUMBLE, Jez a David FARLEY, 2010. *Continuous delivery: reliable software releases through build, test, and deployment automation*. Upper Saddle River, NJ: Addison-Wesley. ISBN 978-032-1601-919.
- LEWANDOWSKI, Michael, 2017. Three levels of TDD. [online]. 5.2.2017 [cit. 2018-12-12]. Dostupné z: http://lewandowski.io/2017/02/thre-levels-of-tdd-1/
- LINDERS, Ben, 2017. *Automated Acceptance Testing Supports Continuous Delivery.* 28.4.2017 [cit. 2018-12-12]. Dostupné z: https://www.infoq.com/news/2017/04/acceptance-testing-delivery
- PITTET, Sten, 2018. Atlassian: Continuous integration vs. continuous delivery vs. continuous deployment. *Atlassian* [online].[cit. 2018-10-12]. Dostupné z: https://www.atlassian.com/continuous-delivery/ci-vs-ci-vs-cd
- UKIS Vladyslav, FARLEY Dave, 2018. Adopting Continuous Delivery at teamplay, Siemens Healthineers [online]. Dostupné z: https://www.infoq.com/articles/continuous-delivery-teamplay
- WELLS, Don. Pair Programming. Extreme Programming [online]. 1997 [cit. 2018-12-20]. Dostupné z: http://www.extremeprogramming.org/rules/pair.html
- Gartner, 2018. Continuous Integration (CI). [cit. 2018-12-10]. Dostupné z: https://www.gartner.com/it-glossary/continuous-integration-ci/
- Siemens Healthineers, 2018a. *About Siemens Healthineers* [online].[cit. 2018-10-12]. Dostupné z: https://www.healthcare.siemens.com/about
- Siemens Healthineers, 2018b. *Background Information* [online].[cit. 2018-10-12]. Dostupné z: https://www.healthcare.siemens.com/press-room/backgrounders
- Siemens Healthineers, 2018c. *Teamplay* [online].[cit. 2018-11-12]. Dostupné z: https://www.healthcare.siemens.com/infrastructure-it/digital-ecosystem/teamplay

- Underconsideration, 2016. *New Name and Logo for Siemens Healthineers.* [online]. 26.5.2016 [cit. 2018-12-12]. Dostupné z: https://www.underconsideration.com/brandnew/archives/new_name_and_logo_for_siemens_healthineers.php

Seznam obrázků