Semestrální práce ke kurzu 4IT421 Zlepšování procesů budování IS	
Semestr	LS - 2018/19
Autoři	Jan Zhouf, zhoj02 Vojtěch Podaný, podv00
Téma	Continuous Delivery. Huge Benefits, but Challenges Too
Datum odevzdání	19. května 2019

Abstrakt

Tato semestrální práce se zabývá jedním ze současných způsobů vývoje softwaru: Continuous Delivery. Continuous Delivery (nebo také kontinuální dodávání, zkráceně pak CD) je postup pro automatizovanou kontrolu kódu po vývojářově commitu, sestavení tohoto kódu, jeho následné otestování a příprava k nasazení na produkční prostředí.

Cílem této semestrální práce je tedy popsat kontinuální dodávání, jeho výhody, nevýhody a na modelových příkladech pak časté problémy při zavádění CD do prostředí podnikového IT.

Klíčová slova

Continuous Delivery, CD, kontinuální dodávání, CI/CD pipeline

Obsah

Úvod

Cílem této semestrální práce je popsat kontinuální dodávání, jeho výhody, nevýhody a na modelových příkladech pak časté problémy při zavádění CD do prostředí podnikového IT.

Kontinuální dodávání je nový přístup k vývoji softwaru. Představili ho autoři Jez Humble a David Farley ve své knize *Continuous Delivery – Reliable Software Releases Through Build, Test And Deployment Automation* (2010). Zde popsali kontinuální dodávání jako nástroj pro automatizaci sestavení a otestování kódu softwarové aplikace. Ve své podstatě znamená, že jakmile vývojář ukončí vytváření kódu a odešle jej do sdíleného repozitáře (např. GIT) pomocí commitu (příkazy GIT commit, GIT push), tak se kód automatizované sestaví a následně také otestuje pomocí aplikačních, integračních a akceptačních testů.

Firma dospěje k zavedení kontinuálního dodávání do svých procesů často ve chvílích, kdy si uvědomí, že vývoj, příprava testovacího prostředí, samotné testy, příprava produkčního prostředí a následné nasazení nové verze softwaru do produkce trvá neúměrně dlouhou dobu v řádu týdnů až měsíců, během kterých vývojáři ani analytici neobdrží zpětnou vazbu na svůj vznikající produkt.

Kontinuální dodávání je tedy nástroj, pomocí kterého dosáhne firma snadnějších změn v oblastech nových prvků systému, změn konfigurace systému a oprav chyb a jejich rychlejšího nasazení do produkčního prostředí, případně k rukám uživatele, za podmínek zvýšené bezpečnosti oproti manuálnímu nasazování a v rychlejším čase. (Humble 2017)

Kontinuální dodávání má za cíl zajistit to, že je kód za každých okolností vždy v nasaditelném stavu, ať už se jedná o malou aplikaci, nebo velký distribuovaný systém, do kterého na denní bázi přispívá mnoho vývojářů. Toto zajistí odstranění problémů vznikajících při integraci a konfiguraci, které následují obvykle po fázích vývoje v klasickém pojetí a celý proces standardizuje a zautomatizuje. (Humble 2017)

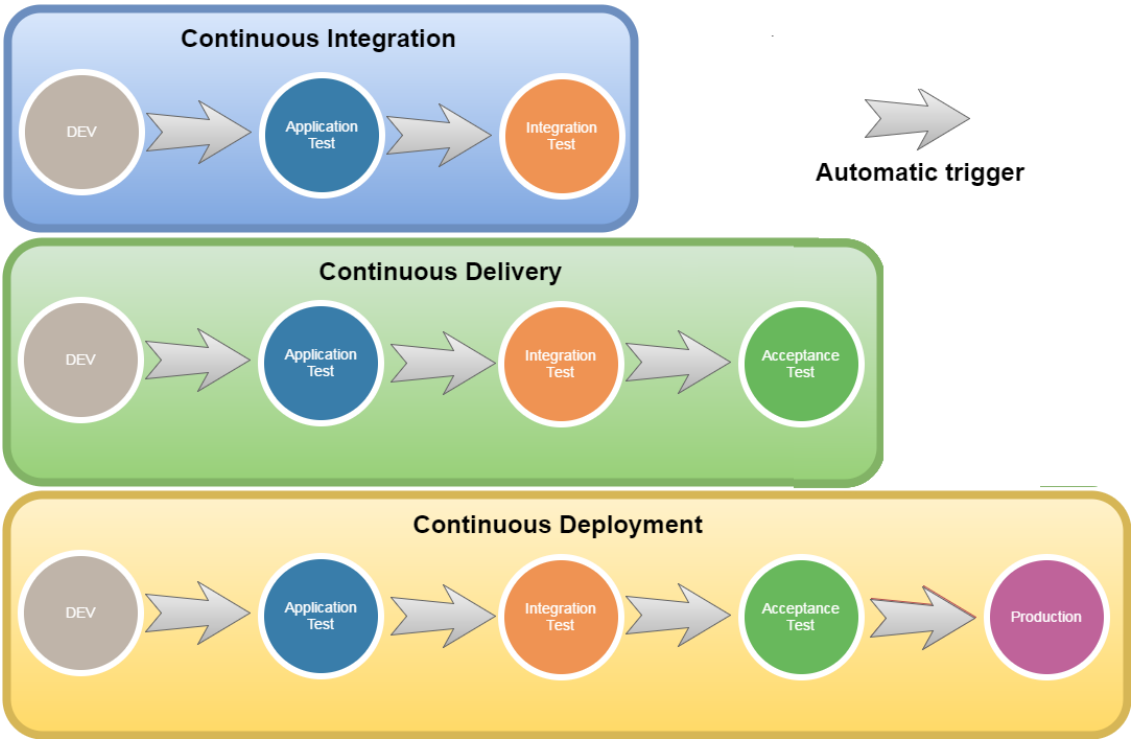
Vymezení pojmů

Je důležité si zde definovat i samotný pojem Continuous Delivery, protože rozdíly s dalšími příbuznými pojmy Continuous Integration a Continuous Deployment nejsou natolik zřetelné a samotný článek (Chen, 2015) je definuje velmi nejasně.

Continuous Integration – jedná se o automatickou kontrolu kódu při commitu, následné jednotkové testy, sestavení kódu a provedení aplikačních a integračních testů,

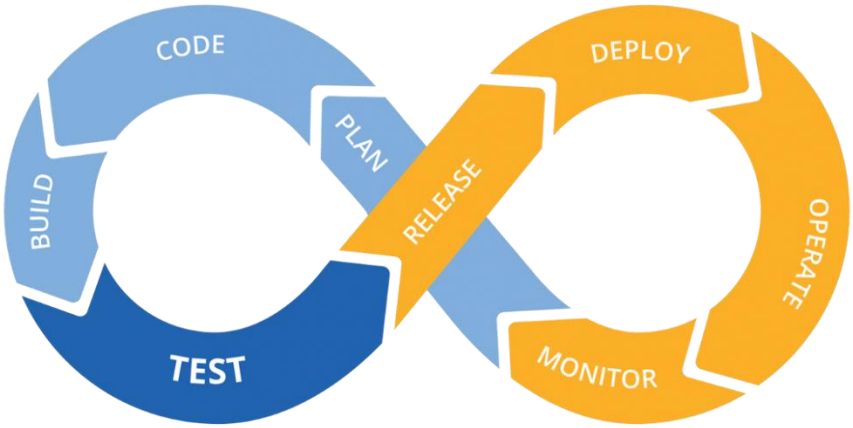
Continuous Delivery – je přidán automatizovaný akceptační test a příprava kódu pro nasazení na produkční prostředí, samotné nasazení na produkční prostředí zůstává manuálním úkonem,

Continuous Deployment – je přidána automatická konfigurace produkčního prostředí a automatické nasazení na něj. (Anastasov 2017)



Obrázek 1: Rozdíly mezi Continuous Integration, Delivery a Deploymentem (Stackoverflow 2018)

CI/CD pipeline je hlavní nástroj, který se používá při využívání kontinuální integrace a kontinuálního dodávání. Představuje kombinaci vícero dostupných nástrojů používaných pro automatizaci sestavení a otestování kódů, jakými jsou třeba Jenkins, Docker, CircleCL, Codeship a další. (Azeri 2019)



Obrázek 2 CI/CD pipeline (Azeri 2019)

1. Výhody a nevýhody CD

V této části bude na základě dostupných zdrojů a praxe uveden přehled a popis výhod a nevýhod zavedení kontinuálního dodávání do podnikového IT.

1.1. Výhody

Jak už zde bylo zmíněno, hlavním přínosem CD je zajistit co nejhladší a nejrychlejší cestu od odevzdání kódu (*code commit*) k úspěšnému nasazení aplikace (*deployment*) na produkční prostředí. Zde každá níže zmíněná výhoda referuje vlastně k odbourání nějakého negativního postupu, který ve firmě byl dříve používán.

Výhody budou popsány především na základě článku *Continuous Delivery Huge Benefits, but Challenges Too* od *Lianpinga Chena* (2015), který výhody CD popisuje na základě zkušeností s jeho zaváděním ve společnosti *Paddy Power*. Pokud budou použity informace o výhodách z jiného zdroje, tak to bude uvedeno.

Výhody jsou zde rozděleny na výhody ohledně zvýšení produktivity zaměstnanců a na ostatní výhody.

1.1.1. Zvýšení produktivity zaměstnanců

Hlavní výhodou je zvýšení produktivity zaměstnanců, což snižuje náklady firmy a zároveň zvyšuje její ziskovost. Produktivita zaměstnanců se zvýší z řady důvodů. Například kvůli větší automatizaci, okamžitému odhalení problémů a rychlejšímu feedbacku. Tyto důvody budou uvedeny níže.

Zvýšení automatizace

Vždy byl nedostatek IT pracovníků, a také platí, že jsou velice drazí a nechcete, aby trávili čas nad prací, která jde zautomatizovat. Kontinuální dodávání programátorům a dalším (například síťovým administrátorům) uvolní ruce na důležitější práci, například na návrh a na psaní samotného kódu.

Ve firmě Paddy Power trávili až 20 % času na připravování prostředí a další režii u vývoje a nasazování. Při dnešních mzdách IT pracovníků se jedná o znatelné náklady, u větších projektů se jedná tedy až o desítky miliónů.

Okamžité odhalení problémů díky automatickým testům

Lianping Chen zmiňuje jednu obrovskou výhodu. Automatické testy jsou spouštěny mnohem častěji než před implementací CD, a tedy vývojáři mají rovnou informaci, kde udělali chybu.

Bez CD často vývojáři měsíce vyvíjeli aplikaci, ale až těsně před jejím nasazením na produkční prostředí provedli automatické testy, bylo nalezeno spoustu chyb, což vedlo k významnému prodloužení celého vývojového procesu.

Rychlejší feedback od uživatelů

Produktivita zaměstnanců se také zvýší tím, že dostanou reakci od zákazníků dříve než za několik měsíců a nemusí po značně dlouhé době znova studovat problém. Zároveň mohou začít pracovat na nápravě chyby ihned a ne až po uplynutí poměrně dlouhé doby, kterou obvykle zabere odhalení a nahlášení problému.

Jednodušší verzování

U komplexního systému se často narazí na problém s verzováním projektu. Každý vývojář si u sebe dělá několik měsíců tzv. „na vlastním písečku“ a pak je značný problém toto spojit do jedné funkční verze.

CD tento problém adresuje tím, že nutí vývojáře často odevzdávat kód (code commit) do společného repozitáře, což značně eliminuje problém s integrací kódu od více programátorů. Zároveň je tento kód verzován a označen, takže se v budoucnu dá zpětně dohledat jeho stav v daném čase.

Zjednodušení řízení procesu

Ve velkých společnostech se spousty vývojovými týmy může být značně náročné řídit celý proces od návrhu přes vývoj až po sestavení balíku pro produkční prostředí. (Singleton 2019)

CD výrazně pomůže dát celému procesu řád a kontrolu na procesem.

1.1.2. Ostatní výhody

Další výhody se týkají především zrychlení celého procesu a větší spolehlivosti. V námi nalezených zdrojích jsou označeny za klíčové ale ne tak významné, jako výhody spojené se snížením celkových nákladů.

Zrychlení dodání nových funkcionalit softwaru

Důležité je nejdříve představit pojem Time-to-market (TTT). TTT je termín pro časové období mezi prvním návrhem funkcionality a dostupností pro zákazníky. (GONÇALVES 2019)

Zavedením CD se ve firmě Paddy Power se výrazně snížila TTT. Dříve nová verze aplikačního softwaru byla maximálně 2x za rok, po zavedení CD je i několikrát za týden. Ve výjimečných případech byla nová verze i několikrát denně, a to přímo na produkčním prostředí.

Z toho vyplývá, že kontinuální dodávání je důležitá součást agilního vývoje, která výrazně podporuje agilní vývoj. Díky kvalitně implementovanému CD budou změny na produkci ještě rychleji.

Spolehlivé nasazení nových verzí aplikačního softwaru

Kontinuální dodávání snižuje možnost chyb při nasazování téměř na minimum. Bez jeho implementace je nutné často upravovat konfigurační soubory se závislostmi na ostatní komponenty a proměnnými pro konkrétní prostředí. Také je nutné ručně zvolit cílové prostředí a ručně nový balíček s aplikací vložit na prostředí.

To může vyvolat řadu problémů. Může se stát, že balíček pro testovací prostředí se nahraje na produkční prostředí. Pokud například u testovacího prostředí je nastaveno, že při nasazení nové verze se má smazat celá databáze, tak to na produkci může mít nedozírné následky.

Přesný odhad délky integračního procesu

Jedna z výhod procesu, který je detailně řízen, je, že lze odhadnout, za jak dlouho bude změna viditelná na produkci. To ocení především vedení společnosti, která se často dotazuje stylem Kdy bude změna viditelná pro naše klienty?

1.2. Nevýhody

Z výše uvedeného zní kontinuální dodávání jako něco zázračného, co odbourává řadu problémů a dokáže „spasit“ společnost. Ale problém je v tom, že často může dojít ke ztrátě kontroly nad

procesem. Proto budou v této části popsány nevýhody, které s příliš rychlou, nepromyšlenou či nekontrolovanou implementací CD přicházejí

Hrozící nedostatečné otestování

V moderní době se příliš tlačí na to, aby byl proces od zadání až po nasazení na produkci co nejrychlejší nehledě na další okolnosti. (Tyson 2017)

Takto rychlý proces CD vede k tomu, že některé části, které byly běžnou součástí vývoje aplikací, bývají vypuštěny. To se týká především manuálního testování.

Manuální testování výrazně prodlouží proces celého vývoje, protože závisí na alokaci testerů. Samotné testování trvá nějaký čas, během kterého probíhá i komunikace mezi testery, vývojáři a analytiky. Je tedy celkem logické, že pokud je tlak na dodání nových funkcionalit co nejdříve, tak prvním nepříliš uváženým krokem může být právě ušetření času a nákladů při vypuštění manuálního testování.

Avšak to může být dost nebezpečné, protože na produkci se dostanou chyby, které nešli zjisti automatickým testováním. To může vést nejenom ke zhoršenému komfortu uživatelů, ale také ke chybám, které mohou využít hackeři.

Přílišná důvěra – nedostatek řízení procesu

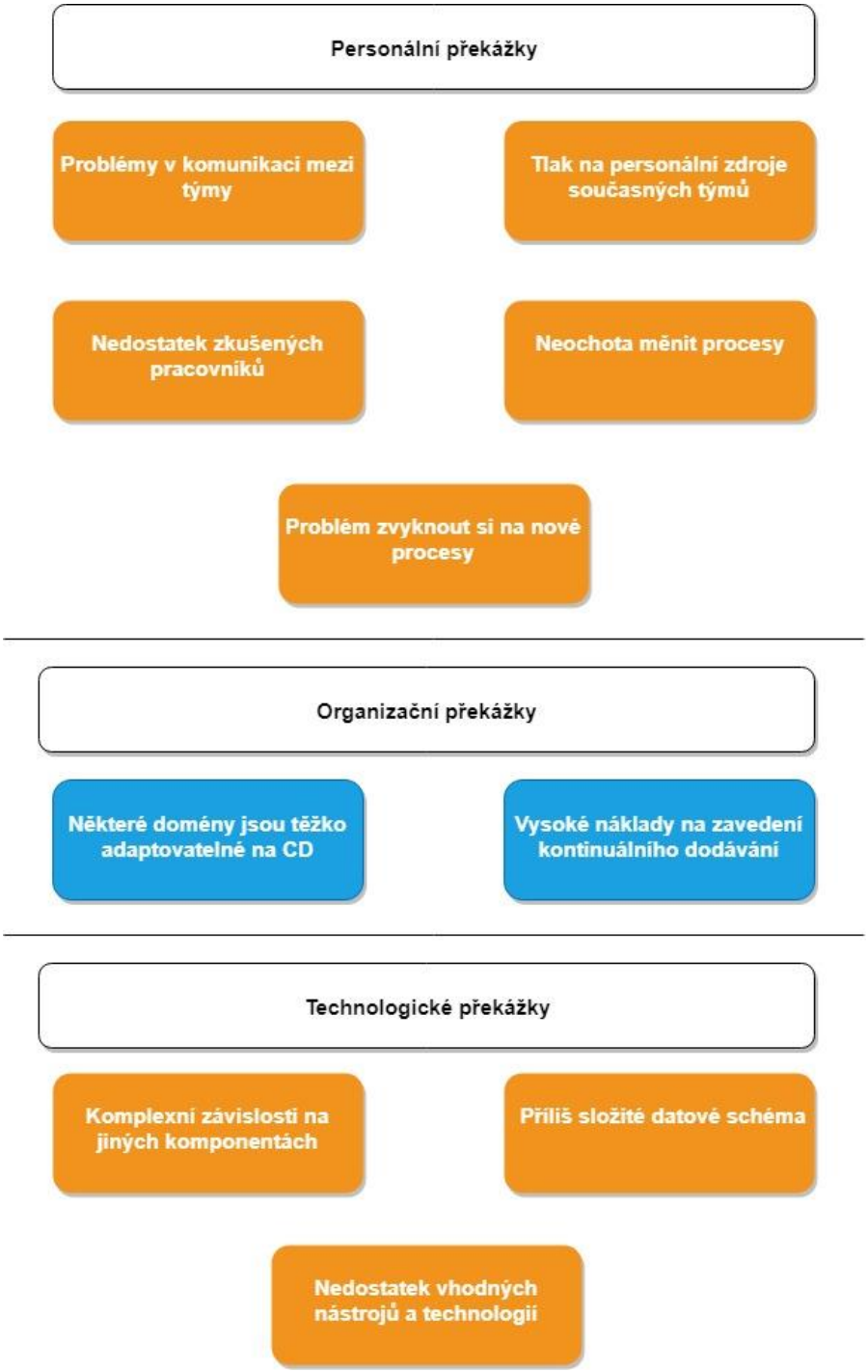
Přílišná automatizace může vést ke ztrátě kontroly nad procesem. Pokud jedním stisknutím tlačítka se provede sada procesů, tak to nikdo nehlídá a můžou se objevit nedozírné chyby. (BLAZEMETER 2016)

Proto v některých vysoce regulovaných segmentech (banky, armády atd.) mají společnosti problém svěřit se do rukou vysoce automatizovaného procesu.

V těchto segmentech lze doporučit postupnou integraci. To znamená zautomatizovat části systému, které nejsou tak klíčové pro danou společnost.

2. Výzvy a problémy při zavádění CD

Při zavádění CD integrátoři narazí na spoustu překážek. Autoři této semestrální práce vybrali nejvýznamnější překážky a rozdělili je to tří kategorií. Jedná se o personální, organizační a technologické překážky.



Obrázek 3 - Druhy překážek [Autoři]

2.1. Personální překážky

Některé z překážek jsou hlavně personálního charakteru. Především se jedná o komunikační problémy. Implementace kontinuálního dodávání často vytvoří tlak už na tak značně vytížené

vývojové a další týmy. A často je problém, že je velký dostatek pracovníků, kteří umějí navrhnout a implementovat CD v odpovídající kvalitě.

Problémy v komunikaci mezi týmy

Zavedení kontinuálního dodávání vyžaduje zvýšení spolupráce mezi týmy, které zajišťují celý životní cyklus aplikace (od jejího vývoje až po nasazení na produkci).

Toto se často ukazuje jako velký problém. Lianping Chen (2015, s. 53) popisuje, že ve firmě Paddy Power to byl klíčový problém. Každá část životního cyklu byla ovládána nezávislými týmy, který nebyly moc ochotní spolupracovat. Pokud chtěl od nějakého týmu pomoci, tak to trvalo i více jak půl roku.

Důležité je všem pořádně vysvětlit význam CD pro firmu a také týmům uvolnit ruce, aby měli na požadavky ohledně CD čas.

Tlak na personální zdroje současných týmů

Další výzvou je tlak na současné týmy ve společnosti. (Shanin 2017, s. 14)

Tým, který implementuje kontinuální dodávání, často potřebuje pomoc od jiných týmů, jejichž členové zároveň musí pracovat na svých úkolech. To může být někdy dost nepřekonatelná překážka. Často tak vyvstane otázka, jestli se věnovat novým inovacím nebo současným úkolům.

Nedostatek zkušených pracovníků

Řada postupů a technologií pro úspěšnou implementaci kontinuální dodávání jsou poměrně náročné. Vyžaduje to určitou senioritu pracovníků. (Shanin 2017, s. 14) Ty je na přehřátém IT trhu téměř nemožné sehnat. Tento fakt může výrazně zkomplikovat či úplně zhatit proces zavedení CD.

Další variantou není hledat už zkušené pracovníky, ale vycvičit juniorní zaměstnance. Avšak to stojí obrovské množství peněz a času.

Neochota měnit procesy

Zavedená společnost funguje určitým způsob. Lze tedy natrefit na neochotu začít dělat něco jinak, měnit procesy. (Shanin 2017, s. 14)

Zde management má významnou roli. Musí podpořit zaměstnance u výše zmíněných problémů, ale hlavně musí držet silnou vizi.

Problém zvyknout si na nové procesy

Výše je zmíněná neochota měnit procesy, to je však problém, který je před a při implementaci CD. Avšak s tím souvisí další problém, a to zvyknout si na nové procesy a dodržovat je. (Shanin 2017, s. 16)

Tohle vyžaduje nutnost zaměstnance s novými procesy řádně seznámit a vytvořit časový prostor pro zvyknutí si na nové procesy.

2.2. Organizační překážky

Další skupinou výzev jsou organizační výzvy. Jedná se především o doménový problém, to znamená, že v některých organizacích by přístup uplatňovaný při kontinuálním dodávání mohl mít negativní vliv.

Některé domény jsou těžko adaptovatelné na CD

Některé oblasti byznysu mohou být značně náročné na implementaci CD. Jedná se především o vysoce regulované oblasti. (Shanin 2017, s. 17)

Tedy se například jedná se o aplikace bank, zdravotnických zařízeních, úřadů státu, či jiných významných institucí nebo dokonce softwaru armády.

U těchto institucí chtějí mít zodpovědné osoby pod kontrolou celý proces. Je obava, že příliš automatizovaný proces by nebyl dostatečně pod kontrolou a v systému by se na produkční prostředí mohly dostat fatální chyby. U těchto domén toto jde vyřešit tím, že je CD zavedeno pouze u části cyklu.

Vysoké náklady na zavedení kontinuálního dodávání

Náklady na zavedení CD hrají poměrně vysokou roli. Náklady vzniknou na více frontách. (Shanin 2017, s. 14)

Nejdříve musíme započítat náklady na hardware, na kterém poběží servery podporující CD.

Následně vzniknou velké náklady na práci vývojářů, analytiků a dalších pracovníků, kteří implementují CD. Nejdříve pracovníci odpovědní za implementaci musí zmapovat současný stav, komunikovat s celou společností, navrhnout podobu CD a následně jí implementovat.

Ve větších firmách implementace CD může stát doopravdy velké množství peněz a času.

Je nutné si tedy spočítat, jestli se CD vyplatí z finančního hlediska.

2.3. Technologické překážky

Poslední skupinou jsou technologické překážky. Jedná se o problémy, které vznikly chybným architektonickým návrhem v minulosti – například příliš komplikované závislosti jednotlivých částí softwaru, anebo málo pružné datové schéma.

Komplexní závislosti na jiných komponentách

Někdy bývá problémem nevhodná architektura aplikačního softwaru, především příliš komplexní softwarové závislosti. (Shanin 2017, s. 17)

U velmi provázaných systémů je náročné pochopit všechny vazby. Vazby často procházejí přes všechny systémy ve firmě, což vyžaduje nutnost zjišťovat informace skrz společnost. Zde zase narážíme na problém, který už byl zmíněn výše: problém v komunikaci mezi týmy.

Příliš složité datové schéma

Jedna z významných technologických překážek je komplikované datovém schéma a časté změny ve schématu. (Shanin 2017, s. 17)

Jednou z cest, jak tuto překážku zmírnit, je zjednodušit datové schéma, především v oblasti vazeb na cizí datové zdroje.

Nedostatek vhodných nástrojů a technologií

Kontinuální dodávání je stále poměrně nový přístup k dodávání softwaru, takže na trhu není dostatek vhodných technologií pro snadnou implementaci. (Shanin 2017, s. 14)

Tento problém často vede k tomu, že společnost si sama musí vyvinout odpovídající nástroje. To znamená vyšší náklady a také zpoždění v implementaci CD.

3. Příklady nevhodného zavedení CD

V této kapitole budou uvedeny dvě modelové situace, kdy je zavedení kontinuálního dodávání do IT společnosti problémové a spíše kontraproduktivní než prospěšné. Tyto situace se zakládají na nevýhodách a častých problémech při zavádění kontinuálního dodávaní, které byly popsány dříve v této práci.

3.1. První příklad

Výchozí podmínky pro první modelovou situaci nevhodného zavedení kontinuálního dodávání zahrnují velký tým mnoha vývojářů, který vyvíjí složitý a komplexní produkt vyvíjený za omezujících podmínek, které můžou být představovány byrokracií ať už ve smyslu intervencí veřejného sektoru nebo i procedurami korporátního prostředí, a dále pak zákonnými požadavky či omezeními a podobně.

Přestože existují tato omezení, tak panuje přesvědčení, že použitím kontinuálního dodávání dojde k úspoře času a nákladů, a to i za cenu omezování manuálních testů a preferenci automatizovaných.

Pokud nedojde k pravidelné kontrole automatizovaných testů a jejich ověřování pomocí dat, u kterých víme předem očekávaný výsledek testů, tak může dojít k problému, že automatizované testy nejsou ošetřené proti tzv. false positive výsledkům (neoprávněně označené jako vyhovující) nebo false negative (neoprávněně označené jako chybové). Toto samozřejmě snižuje přínos celého kontinuálního dodávání a podrývá jeho základní princip – kód neustále připravený k nasazení.

Další problém spojený s touto situací a pokusem o její řešení pomocí automatizovaných testů spočívá v tom, že automatizované testy pokrývají funkcionalitu softwaru, nikoliv však jeho užitečnost. Kód samotný je pak správný a funkční, ale není zaručena reálná funkcionalita aplikace.

3.2. Druhý příklad

Druhým příkladem je pak situace, kdy ve firmě existuje mnoho vývojářských týmů, přičemž každý má své zaběhnuté zvyklosti a postupy. Tyto rutinní návyky omezují jednotlivé pracovníky, i celé týmy v přijímání praktik nových, například i kontinuálního dodávání.

„Klasicky“ orientovaní programátoři zvyklí spíše na vodopádový model vývoje jsou často uzavřenější a zvyklí řešit své problémy sami spíše než komunikovat s kolegy, kteří by jim mohli poradit, také můžou cítit jistou nechuť ke sdílení kódu, a tudíž akceptaci společného repozitáře, do kterého všichni nahrávají svůj kód, který je pak automaticky prověřen a výsledky tohoto otestování jsou pak přístupné všem, stejně jako kód samotný.

Se vzrůstajícím počtem lidí v týmu, a také narůstajícím počtem týmů, začíná tento problém nabývat na důležitosti. Pokud v takovém prostředí nedojde ke správné podpoře týmu, která může obsahovat vysvětlení důvodů proč dochází k nasazení kontinuálního dodávání, vzdělávání členů týmu a jejich motivování k přijetí změn, navýšení společné komunikace a zlepšení vzájemné důvěry, tak může dojít k odmítání nového pořádku, a tedy k výraznému snižování pozitivních efektů, které by zavedení kontinuálního dodávání do firemního IT mělo přinést.

Na mezi týmové úrovni pak nemusí dojít ke stejné úrovni porozumění a stejnému zapojení do nového procesu, čímž dochází ke špatné koordinaci prací a případným rozporům.

4. Závěr

Kontinuální dodávání je moderní způsob vývoje softwaru, kdy při používání nástrojů, jako je například CI/CD pipeline dochází k automatizaci sestavení a otestování kódu, který je tak vždy připravený pro manuální nasazení na produkční prostředí. Kontinuální dodávání je tedy další evoluční krok při vývoji softwaru a není třeba se u jeho použití limitovat pouze na agilní metodiky vývoje.

Při implementaci kontinuálního dodávání je potřeba zvážit, zde je doména daného řešeného problému i samotná společnost vhodná k dát si bacha například na legislativu nebo nepoužívat u konzervativních zákazníků (banky).

Ve zkratce zde na konec ještě uvedeme několik rad k doporučenému postupu při zavádění kontinuálního dodávání do podnikového IT podle webové stránky stránky Blazemeter (2016):

1) Zaměřit se na proces, nikoliv na nástroje,

2) Porozumět potřebám týmu,

3) Vytvořit jeden proces, který pokryje potřeby všech týmů ve společnosti,

4) Implementovat vybrané nástroje,

5) Vytvořit roli Continuous Delivery Engineere odpovědného za proces kontinuálního dodávání.

Literatura

ANASTASOV, Marko. *What's the Difference Between Continuous Integration, Continuous Deployment and Continuous Delivery?* [online]. 2017, Semaphore [cit. 2019-05-17]. Dostupné z: https://semaphoreci.com/blog/2017/07/27/what-is-the-difference-between-continuous-integration-continuous-deployment-and-continuous-delivery.html

AZERI, Izzy. *What is CI/CD?. Mabl* [online]. [cit. 2019-05-17]. Dostupné z: https://www.mabl.com/blog/what-is-cicd

BLAZEMETER. *Why Most Companies Are Getting Continuous Delivery Wrong. Agile Connection* [online]. 2016 [cit. 2019-05-17]. Dostupné z: https://www.blazemeter.com/blog/why-most-companies-are-getting-continuous-delivery-wrong

CHEN, Lianping. *Continuous Delivery: Huge Benefits, but Challenges Too*. In: *IEEE Software* [online]. 2015, 32(2), s. 50-54 [cit. 2019-03-19]. DOI: 10.1109/MS.2015.27. ISSN 0740-7459. Dostupné z: http://ieeexplore.ieee.org/document/7006384/

HELLER, Martin. *Continuous integration: The answer to life, the universe, and everything?*. In: *TechBeacon* [online]. [cit. 2019-03-19]. Dostupné z: https://techbeacon.com/devops/continuous-integration-answer-life-universe-everything

HUMBLE, Jez. What is Continuous Delivery? [online]. 2017 [cit. 2019-05-17]. Dostupné z: https://continuousdelivery.com/

HUMBLE, Jez a David FARLEY. *Continuous delivery: reliable software releases through build, test, and deployment automation.* Upper Saddle River, NJ: Addison-Wesley, 2010. ISBN 978-0-321-60191-9.

GONÇALVES, Luis. Time to Market, What Is And How You Can Speed Up The Process. *EVOLUTION4ALL* [online]. 2019 [cit. 2019-05-17]. Dostupné z: https://www.organisationalmastery.com/time-to-market/

SHAHIN, Mojtaba, ALI BABAR, Muhammad, ZHU, Liming. Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices. In: *IEEE Access* [online]. 2017, 5, s. 3909-3943 [cit. 2019-03-19]. DOI: 10.1109/ACCESS.2017.2685629. ISSN 2169-3536. Dostupné z: http://ieeexplore.ieee.org/document/7884954/

SINGLETON, Andy a spol. *Continuous Delivery: Benefits and Costs. A Guide to the New Continuous Agile* [online]. 2019 [cit. 2019-05-17]. Dostupné z: http://www.continuousagile.com/unblock/cd_costs_benefits.html

STACKVERFLOW. Continuous Integration vs. Continuous Delivery vs. Continuous Deployment. In: *Stackoverflow* [online]. 2018 [cit. 2019-05-19]. Dostupné z: https://stackoverflow.com/questions/28608015/continuous-integration-vs-continuous-delivery-vs-continuous-deployment

TYSON, John. Has Continuous Deployment Become a New Worst Practice?. Agile Connection [online]. 2019 [cit. 2019-05-17]. Dostupné z: https://www.agileconnection.com/article/has-continuous-deployment-become-new-worst-practice