IMPACTOS NA IMPLANTAÇÃO DE METODOLOGIAS DE ENTREGA CONTÍNUA DE CÓDIGO: UMA REVISÃO SISTEMÁTICA DA LITERATURA

Gabriel Dezan Busarello e Walter Silvestre Coan

Bacharelado em Sistemas de Informação
Universidade da Região de Joinville (Univille) Joinville – SC – Brasil

{gabriel.busarello, walter.s}@univille.br

Abstract. *Continuous Delivery (CD), is a software engineering discipline, aiming to make software available to be delivered to the user at any time. The objective of this work is to carry out a sample survey of articles to analyze the impacts on the implementation of CD methodologies and tools. CD is linked to the agile methodologies that are increasingly needed. This article presents as a methodology, a systematic review of the literature, in which the impacts of the implementation of the continuous code delivery methodology are analyzed, having as the work result the positive and negative aspects of the implementation by several researched authors.*

Resumo. *Entrega contínua de código (CD), é uma disciplina relacionada com engenharia de software, objetivando disponibilizar um software para ser entregue para o usuário a qualquer momento. O objetivo desse trabalho é realizar um levantamento amostral de artigos para analisar os impactos na implantação de metodologias e ferramentas de CD. CD é ligada as metodologias ágeis que são cada vez mais necessárias. Este artigo apresenta como metodologia, uma revisão sistemática da literatura, na qual analisa-se os impactos da implantação da metodologia de entrega contínua de código, tendo como resultado do trabalho os aspectos positivos e negativos da implantação por vários autores pesquisados.*

1. Introdução

Entrega Contínua de Código (CD – *Continuous Delivery*), é uma disciplina relacionada com a engenharia de *software*, que tem como objetivo principal, a disponibilidade de um *software* ser entregue para o usuário a qualquer momento. Usando uma revisão sistemática da literatura como metodologia, foi realizado um levantamento do impacto da implantação de metodologias de entrega contínua de código, visando objetivar a questão de pesquisa, que é, qual o impacto da implantação de metodologias de entrega contínua?

Por meio da revisão, identificar impactos acerca do que a implantação gera, quais fenômenos comportamentais, econômicos e processuais que acontecem na implantação da metodologia, pela falta de conteúdos e materiais qualificados no assunto. Tendo como objetivo realizar um levantamento amostral de artigos para a analisar os impactos na implantação da metodologia de CD.

O desenvolvimento de *software* tendo em vista o uso das metodologias ágeis, é prejudicado, quanto a velocidade da entrega da solução, com CD é possível diminuir o custo de processamento das máquinas, automatizar processos, diminuindo erros e

garantindo a qualidade. A pesquisa teve como resultado mais importante, os impactos que cada autor citado no trabalho, obtiveram implantando a metodologia.

2. Revisão da literatura

Antes de entender a engenharia de *software* é necessário entender sobre a natureza do *software*, Pressman e Maxim (2016, p. 3) afirmam que "Hoje, o software tem um duplo papel. Ele é um produto e, ao mesmo tempo, o veículo para distribuir um produto". O *software* diferentemente do *hardware*, não sofre um desgaste com o tempo, por efeitos maléficos do ambiente, portanto, o "software não se desgasta. Mas deteriora!" (PRESSMAN; MAXIM, 2016, p. 5). O *software* não sofrendo com fatores do ambiente, ainda assim ele se deteriora, por causa de fatores como: mudança de requisitos, ajustes de erros, novas funcionalidades.

Engenharia de *software* é uma área voltada ao desenvolvimento de software que abrange um conjunto de métodos e ferramentas, para desenvolver um *software* com qualidade. Segundo Bourque e Fairley (1990 *apud* PRESSMAN; MAXIM, 2016, p. 15) engenharia de *software* é "(1) A aplicação de uma abordagem sistemática, disciplinada e quantificável no desenvolvimento, na operação e na manutenção de *software*; isto é, a aplicação de engenharia ao software. (2) O estudo de abordagens como definido em (1).". Entretanto "uma abordagem 'sistemática, disciplinada e quantificável' aplicada por uma equipe de desenvolvimento de software pode ser pesada para outra." (PRESSMAN; MAXIM, 2016, p. 15), isso quer dizer que nem toda abordagem aplicada a uma equipe pode ser aplicada à outra literalmente.

Gerenciamento de configurações de software, é compreendido pela identificação, controle, contabilidade de status, auditoria e entrega de configuração de software, e para isso tem-se ferramentas que ajudam no processo, isso é chamado de SCM (*Software Configuration Management* – Gerenciamento de Configuração de *Software*). No gerenciamento, tem-se a implementação do plano. Na implementação do SCM o SCMP (*Software Configuration Management Project* – Projeto de Gerenciamento de Configuração de *Software*), normalmente é necessário desenvolver uma série de procedimentos subordinados mais detalhados que definem como os requisitos específicos serão executados durante as atividades diárias - por exemplo, quais estratégias de ramificação serão usadas e com que frequência as compilações ocorrem e testes automatizados de todos os tipos são executados. (BOURQUE; FAIRLEY, 2004, p. 6-5). Esses processos, ajudam na definição de estratégias de controle de versão de código, como serão realizadas ramificações e de quanto em quanto tempo aconteceram compilações e testes do produto, que variam de caso a caso.

Para conseguir gerenciamento completo do processo, é importante identificar que configuração será necessária, nas palavras de Bourque e Farley (2004, p. 6-6) "A identificação de configuração de software identifica itens a serem controlados, estabelece esquemas de identificação para os itens e suas versões e define as ferramentas e técnicas a serem usadas na aquisição e gerenciamento de itens controlados", dessa forma, identificando esses assuntos, é possível realizar um gerenciamento melhor e mais assertivo. Métodos de *software* definem diversas tarefas a serem realizadas ao desenvolver *softwares*, os autores Pressman e Maxim (2016, p. 16) incluem nessas tarefas "comunicação, análise de requisitos, modelagem de projeto, construção de programa, testes e suporte". Essas tarefas são realizadas em todo o ciclo de vida do *software*, como

a comunicação, análise de requisitos e modelagem do projeto, que estão no início do ciclo, até a construção do *software*, testes e por fim o suporte, que estão no fim do ciclo.

Ciclo de vida, é compreendido desde o início do projeto – onde são realizadas a comunicação e, consequentemente o levantamento de requisitos – até o fim do uso do *software*. Paula Filho (2009, p. 89) pensa que "em um processo de desenvolvimento de software, o ponto de partida para a arquitetura de um processo é a escolha de um **modelo de ciclo de vida**", existem diferentes tipos de ciclos de vida.

A integração contínua, é útil para o trabalho em equipes, pois executa sempre a aplicação após alterações. "Integração contínua exige que a aplicação seja compilada novamente a cada mudança feita e que um conjunto abrangente de testes automatizados seja executado." (HUMBLE; FARLEY, 2014, p. 55), assim sendo, além de executar a aplicação novamente, a integração contínua executa os testes unitários, que são automatizados. Permite que várias pessoas trabalhem ao mesmo tempo, no mesmo projeto, sem muitos problemas de conflitos. Feita a ramificação e as alterações do desenvolvedor nessa ramificação, o desenvolvedor irá consolidar o código desenvolvido no tronco principal, e é ai que acontecem as validações, testes automatizados, automação de compilação de código, e podem ocorrer etapas customizadas para cada projeto, para provar que o *software* ainda funciona, sendo mais fácil e rápido de encontrar os erros, "Defeitos são descobertos mais cedo no processo de entrega, e é mais barato consertá-los, o que representa ganhos de custo e tempo." (HUMBLE, FARLEY, 2014, p. 56).

O CD (*Continuous Delivery*) – entrega contínua de código, consiste na disponibilidade de entrega do *software* a qualquer momento para o cliente, Fowler propõe que (2013 *apud* LAUKKANEN; ITKONEN; LASSENIUS, 2016, p. 55) "praticar CD reduz o risco de implantação, permite o rastreamento de progresso confiável e permite feedback rápido do usuário". Entrega Contínua (CD), já existe a alguns anos, mas nunca foi adotada largamente, "Embora instruções de como adotar CD existem a alguns anos, a indústria ainda não adotou a prática em geral, e ainda há àqueles que deram passos em direção ao CD e acharam um desafio" (LAUKKANEN; ITKONEN; LASSENIUS, 2016, p. 55).

3. Procedimentos metodológicos

Este trabalho consiste em uma revisão sistemática da literatura, que é, "uma forma de pesquisa que utiliza como fonte de dados a literatura sobre determinado tema" (SAMPAIO; MANCINI, 2007, p. 84), com essa abordagem, é possível obter um número grande de artigos sobre os temas e integrar as informações em um grupo de estudos e assim, objetivando responder a uma pergunta de pesquisa. A revisão sistemática da literatura, como Sampaio e Mancini (2007, p.84) expressam "disponibiliza um resumo das evidências relacionadas a uma estratégia de intervenção específica, mediante a aplicação de métodos explícitos e sistematizados de busca, apreciação crítica e síntese da informação selecionada", ou seja, é uma busca, com estratégia definida, e que realiza um processo de síntese – filtragem – para obter um resultado em um espectro maior de resultados relevantes, não se limitando a apenas alguns artigos.

Antes de iniciar uma revisão sistemática da literatura, Sampaio e Mancini (2007, p. 85) dizem que precisam ser consideradas três etapas: definir o objetivo da revisão – procurar o assunto e como ele será abordado –, identificar a literatura – buscar os artigos e tudo que é necessário nas bases de pesquisa e artigos científicos – e por fim selecionar

os estudos de serem incluídos – que é a filtragem dos artigos e estudos buscados – essas são etapas preliminares importantes, que auxiliam os pesquisados a adequar a pergunta norteadora da revisão com base na informação.

A questão principal que orienta a pesquisa é, qual o impacto da implantação de metodologias de entrega contínua? Para isso, o trabalho analisou a questão, com uma conclusão acerca da implantação, fenômenos comportamentais, procedurais e afins, sobre a metodologia. Como metodologia, essa pesquisa tem um levantamento amostral de artigos para analisar os impactos na implantação de metodologias e ferramentas de CD.

A busca dos artigos foi feita por meio de uma pesquisa manual, nos repositórios de artigos científicos, foram encontrados 53 artigos, distribuídos em 14 repositórios. Para buscar esses artigos, foram utilizadas *strings* de busca, as quais são: *devops*; *continuous deployment projects*; *continuous deployment*, tendo encontrado 20, 17 e 16 artigos de cada *string*, respectivamente. Abaixo está a quantidade de artigos encontrado em cada repositório científico consultado.

Tabela 1 – Repositórios consultados e a quantidade de artigos encontrado em cada um, em ordem decrescente.

Repositório	**Quantidade**	**Repositório**	**Quantidade**
IEEE	19	**Google Patents**	2
ACM Digital Library	11	**CEUR-WS**	1
SpringerLink	6	**Cutter IT Journal**	1
ResearchGate	3	**Moodle**	1
arXiv	2	**Plos Journal**	1
ScienceDirect	2	**IJCRT.ORG**	1
Wiley Online Library	2	**JACoW.org**	1
Total		**53**	

Fonte: Autor (2021)

Após a busca, os artigos foram filtrados, e seguiram critérios de inclusão e exclusão. Como de inclusão pode ser citado: Artigos que descrevam projetos reais de implantação e artigos que tem como ambiente de estudo, empresas de desenvolvimento de *software*, projetos de pesquisa acadêmica e projetos públicos. De exclusão: Artigos que se referem a ferramentas ou formas de usá-las e artigos que não focam em um caso real de aplicação da metodologia. E esse processo de filtragem, aconteceu duas vezes.

Os artigos que foram aceitos nos critérios de inclusão, após as duas filtragens, foram 24, sendo que 29 não foram aceitos. Os artigos que passaram, são analisados, e os casos são separados em tipos, para então, serem levantados os impactos da implantação, e assim ser possível exprimir com assertividade, dentro dos critérios definidos, como os autores lidaram com a implantação da metodologia.

4. Análise dos dados e discussão dos resultados

No total foram identificados, 123 impactos, dentre eles 24 relações causais, 52 impactos negativos e 47 impactos positivos, divididas em 7 temas: Integração; Teste; Projeto de

sistema; Fatores humanos e organizacional; Recursos; Projeto de construção e Liberação. O tema que mais teve impactos identificados foi o Fatores humanos e organizacional, seguido por Teste e Projeto de sistema. Os outros temas, foram Recursos, Integração, seguidos de Projeto de Liberação e Projeto de Construção.

Foram selecionados os impactos mais relevantes com a pesquisa, pois não haveria como comportar todos os impactos neste trabalho. Abaixo tem uma introdução dos artigos e depois os temas que mais tiveram impactos, até os que tiveram menos, começando por humano e organizacional, seguido por teste e projeto de sistema.

No artigo de (OLSSON; ALAHYARI; BOSCH, 2012) "*Climbing the 'Stairway to Heaven*'", tem como tema "Um estudo de caso múltiplo explorando as barreiras na transição do desenvolvimento ágil para a implantação contínua de *software*". Como resultado do estudo eles identificaram "barreiras na direção da implantação contínua de *software* – bem como as ações que precisam ser tomadas para superá-las".

O artigo dos autores (SAVOR; DOUGLAS; GENTILI, 2016) "*Continuous Deployment at* Facebook *and* OANDA", tem como tema, descrever "as práticas de implantação contínua em duas empresas muito diferentes: Facebook e OANDA". Como resultado eles apresentaram "que a implantação contínua não inibe a produtividade ou a qualidade, mesmo em face do crescimento substancial da equipe de engenharia e do tamanho do código [...]. Identificamos os elementos que consideramos que tornam a implantação contínua viável e apresentamos observações da operação em um ambiente de implantação contínua".

No artigo de (SHAHIN; BABAR; ZAHEDI et al., 2017) "*Beyond Continuous Delivery: An Empirical Investigation of Continuous Deployment Challenges*". Como objetivo apresenta "investigar e classificar empiricamente os fatores que podem impactar na adoção e implementação da prática de CD". Como resultado do estudo, "revela 11 fatores de confusão que limitam ou desmotivas as organizações de *software* para empurrar mudanças automática e continuamente para produção".

O artigo de (SHAHIN; BABAR; ZAHEDI et al., 2016) "*The Intersection of Continuous Deployment and Architecting Process: Practitioners' Perspectives*". Como objetivo a "pesquisa visa explorar empiricamente o impacto potencial da prática de CD no processo de arquitetura", teve como resultado identificaram uma série de desafios arquitetônicos recorrentes, arquitetura monolítica, dependências de equipe, ambientes e ferramentas, problemas com *logs* e testabilidade.

No artigo de (BUCENA; KIRIKOVA, 2017) "*Simplifying the DevOps Adoption Process*". Tem como objetivo "um método dedicado de adoção de *DevOps* é proposto para simplificar a adoção de *DevOps* em pequenas empresas" e tem como questão de pesquisa "Qual é o método para simplificar a adoção do *DevOps?*". Como resultado "O artigo relata os resultados de pesquisas na facilitação da adoção do DevOps em pequenas empresas [...]. O método proposto foi testado em uma filial nacional de uma empresa internacional com uma equipe interna de desenvolvimento de TI".

O artigo dos autores (CALIA; FUCHSBERGER; HOSTETTLER, et al., 2016) "Testing the untestable: a realistic vision of fearlessly testing (almost) every single accelerator component without beam and continuous deployment thereof", tem como objetivo "dar uma breve visão geral das estratégias de teste comumente usadas em projetos de desenvolvimento de software e ilustraremos sua aplicação em componentes

aceleradores, tanto de hardware quanto de software" e como resultado "vários exemplos de sistemas CERN nos quais essas técnicas foram ou serão aplicadas [...] e serão mostrados e descreverão por que vale a pena fazê-lo".

Fatores humanos e organizacionais

Fatores humanos e organizacionais, são relacionados com como a implantação da metodologia, impacta todos que estão ligados ao processo, abrangendo todos os comportamentos individuais ou culturais na organização.

Quadro 1 – Impactos negativos e positivos levantados pelos autores, sobre fatores humanos e organizacionais

Impactos	
Impactos Negativos	Equipes de desenvolvimento com dificuldade na implantação por fatores externos ou limitações de conhecimento (BUCENA; KIRIKOVA, 2017)
	Decisões empresariais e/ou gerenciais que impactam na tomada de decisão (SAVOR; DOUGLAS; GENTILI, 2016)
	Arquitetos conhecendo excessivamente o ambiente de operações (SHAHIN; BABAR; ZAHEDI et al., 2016)
	Fatores não técnicos, sendo fatores de confusão (SHAHIN; BABAR; ZAHEDI et al., 2016)
	Falta de suporte a nível de equipe e até mesmo financeiro (BUCENA; KIRIKOVA, 2017)
Impactos Positivos	Introduzir práticas de trabalho ágeis (OLSSON; ALAHYARI; BOSCH, 2012)
	Obter suporte gerencial para a iniciativa de CD (OLSSON; ALAHYARI; BOSCH, 2012)
	Quebrar barreiras na produção, dando mais liberdade (SHAHIN; BABAR; ZAHEDI et al., 2017)
	Melhorar a compensação entre risco e recompensa para os desenvolvedores (SHAHIN; BABAR; ZAHEDI et al., 2016)
	Aumentar a colaboração entre as equipes de desenvolvimento e operações (SHAHIN; BABAR; ZAHEDI et al., 2016)

Fonte: Primária (2021)

Testes

Testes estão ligados diretamente a qualidade do *software* que vai ser entregue para o usuário, como o processo de integração contínua é em grande parte automático, a garantia da qualidade é essencial.

Quadro 2 – Impactos negativos e positivos levantados pelos autores, sobre testes de *software*

Impactos

Impactos Negativos	Grande esforço e gasto de tempo na transição para CD (SHAHIN; BABAR; ZAHEDI et al., 2017)
	Atividades de teste sendo um desafio (OLSSON; ALAHYARI; BOSCH, 2012)
	Testes de longa duração (SHAHIN; BABAR; ZAHEDI et al., 2017)
	Falta de cobertura de teste (SHAHIN; BABAR; ZAHEDI et al., 2017)
	Falsos positivos (SAVOR; DOUGLAS; GENTILI, 2016)
	Impedimentos de *hardware* (OLSSON; ALAHYARI; BOSCH, 2012)
Impactos Positivos	Testes automatizados aumentam a velocidade e frequência da entrega (OLSSON; ALAHYARI; BOSCH, 2012)
	Fatiamento inteligente do sistema (CALIA; FUCHSBERGER; HOSTETTLER, et al., 2016)
	Aumentar o número de teste continuamente (OLSSON; ALAHYARI; BOSCH, 2012)
	Desenvolver uma infraestrutura de testes (OLSSON; ALAHYARI; BOSCH, 2012)
	Motivação e criatividade na criação de testes (SAVOR; DOUGLAS; GENTILI, 2016) e (SHAHIN; BABAR; ZAHEDI et al., 2016)

Fonte: Primária (2021).

Projeto de sistema

É um tema importante na implantação da metodologia, pois sem um projeto de sistema que possa suportar as modificações que devem ser feitas para adoção da metodologia, não é possível realizá-la.

Quadro 4 – Impactos negativos e positivos levantados pelos autores, sobre projeto de sistema

Impactos	
Impactos Negativos	Dependência de componentes (SHAHIN; BABAR; ZAHEDI et al., 2017)
	Arquitetura acoplada (SHAHIN; BABAR; ZAHEDI et al., 2016)
	Falta de canais de comunicação adequados (SAVOR; DOUGLAS; GENTILI, 2016)
	Geração de avisos falsos (SAVOR; DOUGLAS; GENTILI, 2016)
Impactos Positivos	Desenvolvimento modularizado (OLSSON; ALAHYARI; BOSCH, 2012)

	Ciclo de verificação interna reduzido (OLSSON; ALAHYARI; BOSCH, 2012)
	Conjuntos de testes automatizados (OLSSON; ALAHYARI; BOSCH, 2012)
	Altos graus de automação (SAVOR; DOUGLAS; GENTILI, 2016)
	Presença de infraestrutura de implantação e teste eficaz (SAVOR; DOUGLAS; GENTILI, 2016)
	Pensar antecipadamente sobre os problemas que podem ocorrer (SHAHIN; BABAR; ZAHEDI et al., 2016)

Fonte: Primária (2021)

Recursos

Se torna um tema importante, quando visto pelo lado de que as máquinas serão mais utilizadas para testes e integrações, para assim ser possível que o sistema seja entregue sempre que necessário, é importante ter um bom conhecimento do uso de recursos, tanto de *hardware* quanto de *software*.

Quadro 5 – Impactos negativos e positivos levantados pelos autores, sobre recursos de sistema

Impactos	
Impactos Negativos	Ciclo de verificação interna (OLSSON; ALAHYARI; BOSCH, 2012)
	Aumento de recursos pois são várias atualizações pequenas implantadas (SAVOR; DOUGLAS; GENTILI, 2016)
	Infraestrutura não suportar abordagens modernas (BUCENA; KIRIKOVA, 2017)
	Complexidade dos ambientes existentes (BUCENA; KIRIKOVA, 2017)
	Configurações de rede nas instalações dos clientes (OLSSON; ALAHYARI; BOSCH, 2012)
Impactos Positivos	Alertar desenvolvedores sobre o problema de quantidade de iterações, por meio de: Ferramentas de medição, análise estática e regras de código. (SAVOR; DOUGLAS; GENTILI, 2016)
	Criar unidades pequenas e implantáveis de forma independente, diminuindo a carga nas máquinas (SHAHIN; BABAR; ZAHEDI et al., 2016)

Fonte: Primaria (2021)

Integração

Integração liga-se diretamente a como o sistema irá lidar com as modificações para a implantação da metodologia, e como isso impactará os sistemas legados, e que não estão preparados para implantação da metodologia, existem pontos a serem cuidados.

Quadro 6 – Impactos negativos e positivos levantados pelos autores, sobre integração de sistema

Impactos	
Impactos Negativos	Atualizações e novos recursos sendo um desafio por causa do legado (SHAHIN; BABAR; ZAHEDI et al., 2017)
	Aumento no número de erros e chance de indisponibilidade do sistema (SHAHIN; BABAR; ZAHEDI et al., 2016)
	Colocar peso nos sistemas para reunir informações para suporte ao processo (BUCENA; KIRIKOVA, 2017)
Impactos Positivos	A arquitetura extremamente adaptável a mudanças imprevisíveis (SHAHIN; BABAR; ZAHEDI et al., 2016)
	Análise e monitoramento de *logs* no sistema (SHAHIN; BABAR; ZAHEDI et al., 2016)
	Coletar dados operacionais para monitorar o estado atual da aplicação na produção (SHAHIN; BABAR; ZAHEDI et al., 2016)

Fonte: Primária (2021)

Projeto de construção

É necessário que seja realizado um projeto de construção coerente com as especificações da metodologia, para que tudo que seja proposto, seja alcançado, não somente para a criação do sistema, quanto para a geração de informações úteis na identificação de problemas.

Quadro 7 – Impactos negativos e positivos levantados pelos autores, sobre projeto de construção de *software*

Impactos	
Impactos Negativos	Números de problemas críticos constantes, independentemente do número de implantações (SAVOR; DOUGLAS; GENTILI, 2016)
	Falhas identificadas no momento da implantação, sendo difíceis de isolar para identificar a causa raiz (SAVOR; DOUGLAS; GENTILI, 2016)
	Dependência ao nível de aplicação, sendo necessário implantar todos os aplicativos dos quais dependem (SHAHIN; BABAR; ZAHEDI et al., 2016)
Impactos Positivos	O servidor de CI deve construir e testar a nova versão do *software* para a implantação (CALIA; FUCHSBERGER; HOSTETTLER, et al., 2016)
	O servidor de CI, deve ser totalmente automatizado, e todas as instâncias do *software* em execução devem ser reiniciadas automaticamente, para esperar a próxima iteração (CALIA; FUCHSBERGER; HOSTETTLER, et al., 2016)

Fonte: Primária (2021)

Liberação do *software*

Por último, a liberação de *software* que ocorre no fim do ciclo, que é responsável pela liberação e implantação do *software* para os usuários e um tema chave para todo o ciclo funcionar corretamente.

Quadro 8 – Impactos negativos e positivos levantados pelos autores, sobre liberação de *software*

Impactos	
Impactos Negativos	Processo de implantação altamente burocrático (SHAHIN; BABAR; ZAHEDI et al., 2017)
	Aprovação adicionais para ambientes de produção (SHAHIN; BABAR; ZAHEDI et al., 2017)
	Nem todos os clientes usando ambientes compartilhados na nuvem, sendo um grande obstáculo para a mudança para o CD (SHAHIN; BABAR; ZAHEDI et al., 2017)
Impactos Positivos	Atenção aumentada e testes de desempenho serem conduzidos em escala de produção (SHAHIN; BABAR; ZAHEDI et al., 2017)
	Ao invés de ter equipes de recursos, ter equipes de componentes para reduzir o tempo de espera e melhorar a frequência de lançamento (OLSSON; ALAHYARI; BOSCH, 2012)

Fonte: Primária (2021)

Foi observado que dos impactos, o que teve mais impactos, foram os impactos negativos, com 42 destacados pelos autores, e tendo 26 positivos, propostos pelos autores para tentar solucionar os problemas mencionados.

5. Conclusão

Os profissionais de engenharia de *software* tentam cada vez mais melhorar o processo de entrega contínua de código, devido as metodologias ágeis que crescem a cada dia. Apesar das documentações e instruções existentes, os profissionais enfrentam dificuldades para implantar essa metodologia.

Foram relatados os impactos, positivos e negativos, para vários temas identificados, desde fatores humanos e organizacionais, testes, até recursos e projetos de *software.* Com base nos artigos pesquisados, neste estudo, foi feita a pergunta de pesquisa, qual o impacto da implantação de metodologias de entrega contínua? Que neste trabalho por meio de uma revisão sistemática da literatura, objetiva os impactos da implantação, mostrando os impactos negativos e positivos que os autores pesquisados tiveram quando implantaram essa metodologia.

Uma sugestão para trabalhos futuros, é pesquisar nas bases de artigos científicos, mais impactos acerca dos temas que tiveram menos número de impactos, tanto negativos quanto positivos, e até mesmo complementar os impactos identificados nesse trabalho.

Referências

BOURQUE, Pierre; FAIRLEY, Richard E. (Dick). The Guide to the Software Engineering Body of Knowledge (SWEBOK Guide). 2004. Disponível em: https://www.computer.org/web/swebok/v3. Acesso em: 17 ago. 2020.

BUCENA, Ineta; KIRIKOVA, Marite; Simplifying the DevOps Adoption Process. p. 15, 2017.

CALIA, A; FUCHSBERGER, K; HOSTETTLER, M; CERN; GENEVA; SWITZERLAND. TESTING THE UNTESTABLE: A REALISTIC VISION OF FEARLESSLY TESTING (ALMOST) EVERY SINGLE ACCELERATOR COMPONENT WITHOUT BEAM AND CONTINUOUS DEPLOYMENT THEREOF. **Barcelona**, p. 399-402, setembro 2016.

CHEN, Lianping. Continuous Delivery: Overcoming adoption challenges. **Lianping Chen Limited**, p. 72-86, fevereiro 2017.

HUMBLE, Jez; FARLEY, David. **Entrega Contínua**: como entregar software de forma rápida e confiável. 1. ed. Porto Alegre: Bookman, 2014.

LAUKKANEN, Eero; ITKONEN, Juha; LASSENIUS, Casper. Problems, causes and solutions when adopting continuous delivery – A systematic literature review. **Massachussetts**, p. 55-49, outubro 2016.

OLSSON, Helena Holmström; ALAHYARY, Hiva; BOSCH, Jan. Climbing the "Stairway to Heaven" A multiple-case study exploring barriers in the transition from agile development towards continuous deployment of software. **Gothenburg**, p. 392-399, outubro 2012.

PRESSMAN, Roger S.; MAXIM, Bruce R. **Engenharia de Software:** Uma abordagem profissional. 8. ed. Porto Alegre: AMGH, 2016. p. 933.

SAMPAIO, RF; MANCINI, MC. **ESTUDOS DE REVISÃO SISTEMÁTICA:** UM GUIA PARA SÍNTESE CRITERIOSA DA EVIDÊNCIA CIENTÍFICA. P. 83-89, jan./fev. 2007.

SAVOR, Tony; DOUGLAS, Mitchell; GENTILI; Michael. Continuous Deployment at Facebook and OANDA. p. 21-30, maio 2016.

SHAHIN, Mojtaba; BABAR; Muhammad Ali; ZAHEDI, Mansooreh; ZHU, Liming. Beyond Continuous Delivery: An Empirical Investigation of Continuous Deployment Challenges. p. 111-120, novembro 2017.

SHAHIN, Mojtaba; BABAR; Muhammad Ali; ZAHEDI, Mansooreh; ZHU, Liming. The Intersection of Continuous Deployment and Architecting Process: Practitioners' Perspectives. p. 10, Setembro 2016.