Research, Society and Development, v. 7, n. 2, p. 01-33, e672128, 2018
ISSN 2525-3409 (CC BY 4.0)

Proposta de Modelo para Avaliação da Maturidade DevOps

Framework Proposal for DevOps Maturity Evaluation

Carlos de Amorim Levita
Pontifícia Universidade Católica de São Paulo, Brasil
E-mail: carlos.levita@gmail.com
João Augusto Mattar Neto
Pontifícia Universidade Católica de São Paulo, Brasil
E-mail: joao.mattar@gmail.com

Recebido: 14/10/2017 – Aceito: 06/11/2017

Resumo

Este trabalho propõe um modelo para avaliação da maturidade DevOps nas empresas. Ametodologia foi baseada em uma revisão da literatura, a partir da qualfoi estruturada uma teoria que buscou relacionar as principais práticas DevOps. Para cada uma delas, foram definidos alguns constructos da pesquisa, que são justamente os objetos a serem mensurados por este modelo.Por meio de uma estruturação em formato de perguntas com respostas graduadas, é possível indicaro quanto a unidade de análise que está sendo avaliada está aderente a uma abordagem DevOps no desenvolvimento e implementação de aplicações.Portanto, a principal contribuição deste trabalho consiste no modelo de avaliação da maturidade DevOps que, por ser facilmente replicável, pode ser utilizado por diversas organizações para mapear eventuais pontos de melhoria na implementação da perspectiva DevOps no desenvolvimento de *software*.

Palavras-chave: DevOps; Entrega Contínua; Pipeline de Implantação; Integração Contínua; Infraestrutura como Código.

Abstract

This paper proposes a framework for evaluating the DevOps maturity in organizations. The methodology was based on a literature review from which a theory was developed that sought to relate the main DevOps practices. For each of them, some research constructs were defined, which are precisely the objects to be measured by this model. Through a structure based on questions with graded answers, it is possible to indicate how much the unit of analysis that is being evaluated adheres to a DevOps approach in the development and implementation of applications. Therefore, the main contribution of this work is the DevOps maturity evaluation

framework that can be easily replicated and, because of that, may be used by several companies to map possible improvement points in their implementation of the DevOps perspective for software development.

Keywords: DevOps; Continuous Delivery; Deployment Pipeline; Continuous Integration; Infrastructure as Code.

1. Introdução

A busca por uma forma mais eficiente para desenvolver aplicações sempre foi uma atitude constante nas empresas, porém existe um antagonismo entre as áreas de Desenvolvimento e Operações, pois elas têm focos totalmente distintos. O ideal seria que estes dois times trabalhassem de forma harmônica, unindo esforços para chegar a um objetivo comum, que é exatamente a proposta da perspectiva DevOps, cujo objetivo é possibilitar que as áreas de Desenvolvimento e Operações da Tecnologia da Informação (TI) passem a trabalhar de forma colaborativa, de modo a conseguir implementar aplicações com mais agilidade.

Este trabalho propõe um modelopara avaliação da maturidade DevOps nas empresas. A seção seguinte apresenta a revisão de literatura que serviu de fundamento para a construção do modelo e as práticas DevOps utilizadas. Em seguida é descrita a metodologia da revisão e elaboração do modelo, que é apresentado nos resultados.

2. DevOps

Para Erich, Amrit e Daneva (2014, p. 1), que fizeram uma revisão sistemática da literatura sobre DevOps, há relativamente pouca pesquisa acadêmica disponível sobre este tema. Além disso, eles verificaram que não há uma definição precisa de DevOps, nem um processo específico a ser seguido, ou seja, não há uma solução única que sirva para todas as empresas. Portanto, definiram DevOps como uma abordagem conceitual para suportar o desenvolvimento e a operação de sistemas de informação.

Por meio de uma revisão multifocal da literatura que, além de pesquisa acadêmica, inclui também fontes não publicadas, França, Jerônimo Junior e Travassos (2016, p. 56) buscaram definir DevOps e concluíram pela não utilização do termo metodologia, uma vez que não existe uma abordagem sistemática quanto à sua condução nas organizações. Trata-se, portanto, deum neologismo que aborda uma atitude diferente com relação à entrega de

software, por meio da colaboração entre as funções de desenvolvimento e operações de sistemas, com base em um conjunto de princípios, tais como cultura, automação, quantificação e compartilhamento.

Penners e Dyck (2015), em sua pesquisa, analisaram várias definições sobre DevOps e também discordam das propostas que definem DevOps como sendo uma metodologia de desenvolvimento de software, pois não há uma relação de procedimentos ou técnicas para sua implementação, como já existe para Scrum, por exemplo.Para Humble e Molesky (2011, p. 7), DevOps consiste em um alinhamento de incentivos de todos os envolvidos com a entrega do software. Além disso, uma de suas premissas fundamentais é que o atingimento mútuo de implantações confiáveis e frequentes, e de um ambiente de produção estável, não é um jogo de vencedores e vencidos, mas sim uma proposta ganha-ganha.

Spafford e Haight (2014) propõem o conceito DevOps fundamentado em quatro pilares: ter pessoas certas e a cultura adequada, melhorar continuamente os processos, alavancar a tecnologia/automação para quebrar barreiras e empenhar-se para obter informação/métricas precisas para otimizar a retroalimentação de toda a estrutura.

2.1. Práticas DevOps

Um dos objetivos do mapeamento sistemático feito por Jabbari et al (2016) foi justamente identificar e classificar as práticas associadas com a perspectiva DevOps. Eles selecionaram 15 artigos que abordam especificamente as atividades executadas no contexto deste tema e as categorizaram de acordo com as áreas fundamentais do conhecimento, propostas no Swebok (Software Engineering Body of Knowledge).Braga (2015) fez um mapeamento sistemático da literatura disponível e traçou um panorama sobre o uso de práticas DevOps nas indústrias de software.

Sharma e Coyne (2015, p. 9), por sua vez, propuseram uma arquitetura de referência para DevOps, explicando que ela provê, por meio dos seus componentes, um modelo de solução baseado em um conjunto de práticas que permitem a criação de uma plataforma DevOps. Com base nestes autores, foram abordadas as principais práticas DevOps, começando com o Planejamento Contínuo, que visa o estabelecimento de metas e o seu constante ajuste, decorrentes da necessidade de responder de forma ágil aos feedbacks dos clientes, que são recebidos no decorrer do desenvolvimento do software.

Como explicam Humble e Farley (2010, p. 55), na prática deIntegração Contínua, a cada mudança no software, toda a aplicação é compilada, um conjunto de testes

automatizados é executado e, caso ocorra alguma falha nos processos de teste ou compilação, todo o time é acionado para revolver o problema imediatamente. Isto permite detectar erros de forma muito mais rápida, o que representa uma mudança de paradigma para a equipe de desenvolvimento, mas requer, ao mesmo tempo, bastante disciplina para trabalhar de forma colaborativa.

Para Duvall, Matias e Glover (2007, p. 15), não existe integração contínua sem a implementação da prática de Testes Contínuos, pois é por meio deles que os desenvolvedores e demais partes envolvidas no projeto têm confiança nas mudanças feitas no software.Sharma e Coyne (2015, p. 12) explicam que o objetivo é testar o quanto antes e continuamente, durante o ciclo de vida do desenvolvimento, o que leva a uma redução nos custos e nos tempos de testes, bem como a uma melhor qualidade do software. Esta prática é viabilizada pela utilização de técnicas, tais como testes automatizados e virtualização, que permitem simular o ambiente de produção, para que os testes sejam feitos da forma mais real possível.

Segundo Sharma e Coyne (2015, p. 13), a prática de Entrega Contínua viabiliza que novas funcionalidades sejam disponibilizadas para os usuários o mais rapidamente possível. Trata-se da prática que deu origem ao movimento DevOps, pois ela eleva a integração contínua a um patamar superior, possibilitando a criação do pipeline de implantação automatizado. Para Humble e Farley (2010, Prefácio), esta prática possibilita que o time de desenvolvimento consiga entregar o software em produção de forma confiável, previsível e com baixo nível de riscos. Em suma, busca-se reduzir ao mínimo possível o tempo entre fazer alguma alteração na aplicaçãoe conseguir implementá-la, garantindo que os problemas sejam identificados o quanto antes, quando são mais fáceis de serem corrigidos. Para Feitelson, Frachtenberg e Beck (2013, p. 11), a entrega contínua de software faz com que cada nova implantação introduza apenas uma pequena quantidade de código, reduzindo muito o risco de problemas, pois facilita a sua validação. Ademais, esses autores afirmam que encorajar o time a fazer novos commits o mais cedo possível, ajuda a vencer o medo de fazer releases em produção; esta habilidade de implantar rapidamente, em pequenos incrementos e com menos temor, permite um rápido florescimento de inovação.

Virmani (2015, p. 79) explica que a prática de Infraestrutura como Códigopossibilita que todo o provisionamento do ambiente seja tratado como código, mantido em um repositório, da mesma forma que se faz com o software.Segundo Hüttermann (2012, p. 136), escolher a linguagem ou ferramenta adequada e começar a criar uma solução que atenda às necessidades, no formato de uma especificação executável, que possa ser aplicada aos sistemas de forma eficiente e repetitiva, traz agilidade para a equipe de operações.Spinellis

(2012, p. 87) cita que estas ferramentas são um dos principais facilitadores da abordagem DevOps e funcionam por intermédio de regras, que especificam como o setup do ambiente deve ser feito, ou seja, os scripts que gerenciam a configuração são compostos essencialmente de comandos, da mesma forma que se escreve o software, e por isso o nome infraestrutura como código. Para Skelton (2016, p.41), uma explosão de ferramentas está ajudando a impulsionar DevOps nas empresas, pois são fáceis de instalar e de configurar, além de serem feitas permitindo a integração entre si, inclusive no que tange ao provisionamento da infraestrutura. Disponibilizar o ambiente por meio de código testável possibilita o uso de práticas que antes só estavam disponíveis para o mundo do software; somado a isso, serviços de nuvem passaram a tratar a infraestrutura como commodity, o que leva a uma significativa queda nos preços.

A prática de Monitoração Contínua permite que métricas estejam disponíveis para todas as partes envolvidas na entrega do software, para que possam ter uma visão clara de como otime está progredindo e também identificar eventuais gargalos no processo.Gottesheim (2015, p. 3) aborda DevOps com foco em performance e, para tanto, afirma que é fundamental estabelecer um entendimento e também uma linguagem comum entre os times de desenvolvimento, testes e operações a respeito das métricas que serão utilizadas. Para Hamunen (2016, p. 28), o recomendado é justamente criar as funcionalidades de monitoração da aplicação em paralelo com o seu desenvolvimento, por meio de mecanismos que possam ser acionados automaticamente, o que proporciona uma melhor medição da sua estabilidade.Segundo Liu, Li e Liu (2014), o valor inicial da implementação de uma abordagem DevOps é rapidamente reconhecido por meio da integração do software, da automação contínua e da colaboração entre os times de trabalho. Entretanto, este sucesso inicial é apenas o primeiro passo nesta caminhada, pois é importante uma monitoração contínua e a consequente otimização, para que seja possível alcançar um compromisso de longo prazo quanto à entrega de software rápida e eficaz.ParaHernantes,Gallardo e Serrano (2015), a virtualização e a computação em nuvem facilitaram bastante o provisionamento dainfraestrutura,mas, ao mesmo tempo, a quantidade de elementos a ser monitorada aumentou significativamente, pois as aplicações passaram a rodar em pontos geograficamente dispersos, em nuvens públicas ou privadas, ou até mesmo em uma combinação delas (nuvens híbridas), o que dificulta o seu gerenciamento. A escolha da ferramenta de monitoração da infraestrutura deve ser baseada nas funcionalidades necessárias para atender às necessidades do negócio, nos fatores relativos à implementação e manutenção desta ferramenta, de forma que atenda às expectativas e competências do time de TI e, claro, no seu custo de aquisição.

Para Sharma e Coyne (2015, p. 14), a prática de Feedback Contínuo dos usuários permite que o time de operações possa melhorar o ambiente, ou que o time de desenvolvimento possa ajustar as funcionalidades do software, ou que a área de negócios possa rever os seus planos, se isto for necessário para melhorar a experiência dos seus clientes.Farroha e Farroha (2014, p. 289) afirmam que é necessária uma mudança cultural, de forma a incentivar a participação de todos os envolvidos no processo, inclusive do usuário ou área de negócios da empresa, pois isto permite um melhor gerenciamento das suas expectativas, viabilizando que as funcionalidades desenvolvidas sejam mesmo as que realmente importam para o cliente.O ideal é focar mais em trabalho em equipe e comunicação, ao invés de ferramentas e processos, por meio do envolvimento de todas as partes interessadas, de forma a garantir que a agilidade conquistada com a abordagem DevOps possa atender aos requisitos de negócios, mantendo os riscos em patamares adequados, já que foram avaliados por todos os envolvidos.

Segundo Spafford e Haight (2014), a dimensão mais importante para DevOps está relacionada a pessoas e tem a ver com a mudança de cultura da empresa, que necessita rever o papel da TI, o que muitas vezes acaba levando a uma reavaliação da sua estrutura organizacional e a uma maior valorização de profissionais que conheçam várias áreas, como desenvolvimento, teste e operações. Devido à sua importância, Pessoas e Cultura foi considerado como uma das práticas DevOps para fins desta pesquisa.

3. Metodologia

Foi realizada uma revisão de literatura guarda-chuva baseada emParé et al (2015),com buscas nas bases de dados do Portal de Periódicos da Capes e Google Acadêmico, no intuito de localizar eventuais revisões de literatura já realizadas sobre DevOps. Foram encontrados três artigos acadêmicos e uma dissertação de mestrado.A partir dessas quatro revisões da literatura, foram identificados artigos acadêmicos citados nas suas referências. Separados os artigos acadêmicos, procurou-se aqueles que discorriam sobre práticas e técnicas DevOps, principalmente que buscavam algum tipo de análise da sua maturidade nas empresas.

Como resultado da construção do referencial teórico, foram estabelecidos os constructos do modelo, elaborados com base nas oito práticas DevOps que foram identificadas:

a) Pessoas e Cultura: colaboração e estratégia;
b) Planejamento Contínuo: requisitos, planejamento e painel de controle;

c) Integração Contínua: gerenciamento, colaboração e painel de controle;
d) Testes Contínuos: gerenciamento, automação e painel de controle;
e) Entrega Contínua: gerenciamento, automação e painel de controle;
f) Infraestrutura como código: gerenciamento, automação e painel de controle;
g) Monitoração Contínua: gerenciamento e capacidade;
h) Feedback Contínuo: feedback e otimização.

A ideia de confeccionar um modelo de avaliação da maturidade DevOps foi baseada no trabalho de McCarthy et al(2015, p. 601), que propõe uma análise de maturidade para esta perspectiva de desenvolvimento de software. O próximo passo foi pautado em Azoff (2016, p. 8), que avalia várias empresas que oferecem serviços de gerenciamento de release, a partir de uma abordagem DevOps, e lista os fornecedores tidos como líderes neste tópico. Tomando-se como base as sete empresas que se destacaram nesse relatório comparativo, foram feitas pesquisas nos seus websites, com o intuito de verificar se elas dispunham de informações referentes à avaliação da maturidade DevOps. O modelo proposto baseou-se no material obtido junto à cinco destas empresas: Automic, IBM, Incycle, Microsoft e XebiaLabs.

4. Resultados

O modelo completo de avaliação da maturidade DevOps é apresentado nesta seção e pode servir de apoio para a condução de entrevistas estruturadas em um estudo de caso. Recomenda-se que seja seguido um roteiro preestabelecido, que visa cobrir as várias práticas utilizadas na abordagem DevOps de desenvolvimento de software. Esta forma de condução facilita bastante a coleta de informações, pois a obrigatoriedade de trabalhar com uma sequência determinada faz com que o foco seja mantido no rumo correto, evitando possíveis devaneios, que são muito comuns em entrevistas. Além disso, como cada prática DevOps aborda um tópico distinto, recomenda-se escolher a pessoa do time que atue especificamente naquele determinado item, pois desta forma é possível captar a visão de quem mais entende daquele assunto. Deve-se evitar, portanto, que uma única pessoa responda a todas as questões, pois seria algo monótono e cansativo.

4.1. Modelo de Avaliação da Maturidade DevOps

4.1.1. Avaliação da prática Pessoas e Cultura

Os quadros 1 e 2 do modelo abordam as questões referentes à práticaPessoas eCultura e visam medir os constructos Colaboração e Estratégia. Como esta parte é composta de perguntas genéricas, pode ser respondida por qualquer pessoa do time.

Quadro 1 — Avaliação do constructo Colaboração (Pessoas e Cultura)

#	Questões	Diretriz	
1	Seus processos são compartilhados entre desenvolvimento e operações?	Todos os processos são unificados entre Dev e Ops	5
		Dev e Ops usam o mesmo conjunto de processos, mas alguns pontos ainda não estão unificados	4
		Alguns elementos do processo são reaproveitados entre Dev e Ops	3
		Dev e Ops usam processos distintos, mas as interfaces são alinhadas e compartilhadas	2
		Dev e Ops usam processos separados e redundantes, com compartilhamento limitado	1
2	Você trabalha de forma integrada e compartilha suas ferramentas entre Dev e Ops?	Todas as ferramentas funcionam como uma única cadeia de ferramentas, totalmente integradas e automatizadas	5
		As ferramentas entre Dev, QA e Ops são integradas, mas não há um único fluxo entre elas	4
		As ferramentas são parcialmente compartilhadas entre Dev, QA e Ops	3
		Há pouca integração de ferramentas e algum compartilhamento de status, via painéis de controle	2
		Dev e Ops usam um conjunto de ferramentas distintas e redundantes	1
3	Vocês colaboram para a construção de conhecimento compartilhado?	A gestão de conhecimento e o repositório são unificados	5
		Acontecem retrospectivas de conhecimento regulares entre Dev e Ops	4
		O compartilhamento de conhecimento entre Dev e Ops é moderado	3
		O compartilhamento de conhecimento entre Dev e Ops é limitado	2
		Não há compartilhamento de conhecimento entre Dev e Ops	1
4	Vocês dão mais ênfase às tarefas do que aos papéis individuais?	Totalmente implementado	5
		N/A	
		Parcialmente implementado	3
		N/A	
		Não implementada	1

Fonte: adaptado de Automic (2017) e Incycle (2017)

Quadro 2 — Avaliação do constructo Estratégia (Pessoas e Cultura)

#	Questões	Diretriz	
5	Os seus funcionários conhecem os princípios básicos de DevOps e o seu benefício para a empresa?	Os princípios e as práticas DevOps são institucionalizados em toda a empresa	5
		Existe um consenso na empresa acerca dos princípios básicos de DevOps e tem-se construído uma forma de aplicá-los	4
		Existe um consenso sobre a criação de atividades a respeito dos princípios básicos de DevOps e como aplicá-los	3
		As equipes estão melhorando sua compreensão sobre DevOps, mas as atividades permanecem desalinhadas na empresa como um todo	2
		Poucas conversas e nenhuma educação interna sobre DevOps	1
6	A sua empresa está qualificada em métodos e práticas DevOps?	Todas as equipes de Dev, Ops e Infraestrutura têm total experiência e habilidades em DevOps	5
		Vários times têm total experiência e habilidades em DevOps	4
		Vários times têm alguma experiência e habilidades em DevOps	3
		Existem pessoas na empresa que possuem experiência e conhecimento relevantes sobre DevOps	2
		Não existe experiência nem conhecimento sobreDevOps na empresa	1
7	A sua equipe está incentivada e motivada para aplicar as práticas DevOps?	Dev, Ops e Infraestrutura totalmente alinhados por meio de incentivos e toda a empresa está entusiasmada com DevOps	5
		O alinhamento entre Dev e Ops está incorporado na estratégia — a maioria das pessoas são positivas sobre a mudança	4
		A organização tem alguns visionários DevOps e já se nota um sentimento positivo e tentativas de alinhamento entre Dev e Ops	3
		Nenhum incentivo especial para alinhar Dev e Ops e o sentimento é neutro	2
		Nenhum incentivo especial para alinhar Dev e Ops e, além disso, há uma postura defensiva quanto à mudança	1
8	A sua estrutura organizacional está adaptada para DevOps?	O conjunto Dev e Ops está totalmente estratificado por aplicação, plataforma e infraestrutura	5
		Dev e Ops parcialmente combinados em equipes de *release*, mas com algum trabalho ainda feito por fora	4
		Dev e Ops seguem um portfólio de aplicações, mas o funcionamento de algumas aplicações permanece separado	3
		Dev segue um portfólio de aplicações, enquanto Ops permanece guiado por infraestrutura e tecnologia	2
		A estrutura da empresa é guiada por tecnologia, com abordagem separada entre Dev e Ops para os *releases*	1
9	Qual é a sua política em relação ao desenvolvimento de aplicativos móveis?	Usamos *frameworks* para coordenar aplicativos em diferentes dispositivos móveis e na web	5
		Desenvolvemos primeiro os serviços de nuvem comuns e tentamos manter o mínimo possível de lógica única no dispositivo	4
		Desenvolvemos apenas aplicações web compatíveis com dispositivos móveis, não aplicações mobile nativas	3
		Nós terceirizamos o desenvolvimento de aplicativos móveis para especialistas	2
		Não desenvolvemos aplicativos móveis	1
10	A sua metodologia de entrega está bem alinhada com os princípios DevOps?	Agilidade fim-a-fim através de todas as equipes: aplicações, plataforma e infraestrutura.	5
		Todas as equipes de Dev e a maioria das equipes de Ops executam processo ágil fim-a-fim, até a produção	4
		Agilidade integrada nos processos de QA, mas o processo de *release* é separado e não ágil	3
		Há alguma metodologia ágil, mas os processos de QA e *release* são separados e não ágeis	2
		Usamos metodologia tradicional de desenvolvimento (*waterfall*)	1

Fonte: adaptado de Automic (2017) e Microsoft (2017)

4.1.2. Avaliação da prática Planejamento Contínuo

Os quadros 3, 4 e 5 do modelo permitem avaliar a prática Planejamento Contínuo (PC)por meio da medição dos constructos Requisitos, Planejamento e Painel de Controle. O ideal é que estas questões sejam revistas com alguém responsável pelo planejamento dos releases e pela interface com o usuário final.

Quadro 3 — Avaliação do constructo Requisitos (PC)

#	Questão	Diretriz	
1	Como os requisitos funcionais e de negócios são coletados junto ao cliente?	Requisitos são coletados usando uma ferramenta DevOps, integrada com asferramentas usadas para as demais funcionalidades do processo de Dev e Ops	5
		Requisitos são coletados usando uma ferramenta DevOps, como Jira	4
		Requisitos são coletados por meio de ferramentas como Remedy ou Service Now	3
		Requisitos são versionados em um sistema de SCM	2
		Requisitos coletados em documentos Word e comunicação feita apenas via e-mail	1
2	Como você captura o trabalho a ser feito com o time (dentro de casa)? Utiliza alguma ferramenta?	Por meio de um combinado de *backlogs* departamentais ou de todo o projeto e/ou quadros Kanban abrangendo várias equipes	5
		Os *backlogs* combinam funcionalidade e trabalho de *site* ao vivo e hipóteses continuamente refinadas com base em evidências de produção	4
		Por meio de um *backlog* do time, quadro Kanban físico ou virtual, ou fila dedicada de ticket / incidente	3
		Requisitos coletados em documentos Word e comunicação é feita apenas via e-mail	2
		Não há um processo definido	1
3	Qual é a granularidade dos requisitos de negócios?	Os requisitos são elaborados de forma nítida, o que pode ajudar a criar pequenos entregáveis de *software*	5
		Os requisitos são apresentados como grandes documentos, que são uma coletânea de vários requisitos menores	1
4	Como os requisitos são priorizados?	A priorização é feita de forma ágil, usando uma ferramenta de gerenciamento de requisitos, como Jira, e feita de forma integrada com as demais ferramentas	5
		A priorização é feita de forma ágil, usando uma ferramenta de gerenciamento de requisitos, como Jira, mas sem integração com as demais ferramentas	4
		A priorização é feita de forma ágil, mas sem o uso de uma ferramenta para o gerenciamento de requisitos	3
		A priorização é feita utilizando planilhas Excel e com a realização de reuniões com as partes interessadas	2
		A priorização é feita de forma pontual, para cada projeto, sem uso de ferramenta	1
5	Como o trabalho é priorizado?	De forma contínua, por meio de um *backlog* orientado por hipóteses, com base em experiências mensuráveis e coleta de dados de produção	5
		De forma consistente em toda a empresa	4
		Consistentemente dentro das equipes, mas prioridades entre equipes podem ser incompatíveis	3
		Há uma tentativa de priorização dentro das equipes, mas ainda é feito de forma inconsistente e irregular	2
		Não há um processo definido	1

Fonte: adaptado de IBM (2017) e Microsoft (2017)

Quadro 4 — Avaliação do constructo Planejamento (PC)

#	Questão	Diretriz	
6	Como as atividades de projeto são planejadas?	O planejamento do projeto é feito de forma ágil, usando ferramentas DevOps, como Jira ou RTC	5
		O planejamento do projeto é feito de forma ágil, usando ferramentas como Microsoft Project	4
		O planejamento do projeto é feito de forma ágil, usando planilhas Excel	3
		O planejamento do projeto é feito de forma cascata (*waterfall*), usando ferramentas como Microsoft Project	2
		Os planos do projeto são feitos caso a caso, contendo apenas as datas de entrega de alto nível	1
7	Os projetos têm um dono, que comunica ativamente as solicitações e as prioridades?	Totalmente implementado	5
		N/A	
		Parcialmente implementado	3
		N/A	
		Não implementada	1
8	Você usa iterações de duração fixa no seu planejamento?	Totalmente implementado	5
		N/A	
		Parcialmente implementado	3
		N/A	
		Não implementada	1
9	Os SLAs entre as Negócios, Desenvolvimento e Operações atendem aos objetivos de negócio?	Totalmente implementado	5
		N/A	
		Parcialmente implementado	3
		N/A	
		Não implementada	1
10	Existem métricas definidas?	As métricas são bem definidas e os dados são coletados e analisados com o uso de ferramentas DevOps	5
		As métricas para gerenciamento de testes/defeitos, produtividade, etc. são coletadas de forma parcial, com o uso de ferramentas DevOps	4
		As métricas são coletadas e analisadas usando métodos manuais em planilhase relatórios são gerados por meio de automação do Excel	3
		As métricas são definidas e coletadas caso a caso, podendo ser necessário esforço adicionalpara criar os relatórios de gestão	2
		Não há métricas definidas	1

Fonte: adaptado de IBM (2017) e Incycle (2017)

Quadro 5 — Avaliação do constructo Painel de Controle (PC)

#	Questão	Diretriz	
11	Como os requisitos são identificados e rastreados para o *release* de produção?	O Gerenciamento do Ciclo de Vida Colaborativo (CLM) do *software* está implementado usando ferramentas DevOps e com processos ágeis	5
		Rastreabilidade usando planilha Excel ou ferramentas caseiras, sendo os *releases* muito bem planejados e rastreados, mas as atualizações são manuais	4
		Rastreabilidade usando planilha Excel ou ferramentas caseiras, mas não há planos de dados disponíveis ou não há indicadores de dados suficientes disponíveis	3
		Rastreabilidade parcial, somente até o design de alto nível (HLD)	2
		Não há rastreabilidade	1
12	As partes interessadas conseguem visualizar os requisitos e o status do projeto?	Totalmente implementado	5
		N/A	
		Parcialmente implementado	3
		N/A	
		Não implementada	1
13	Os critérios de aceite são claramente definidos e obedecidos?	Totalmente implementado	5
		N/A	
		Parcialmente implementado	3
		N/A	
		Não implementada	1
14	As diferentes arquiteturas de sistemas são bem documentadas e mantidas?	Totalmente implementado	5
		N/A	
		Parcialmente implementado	3
		N/A	
		Não implementada	1
15	Você tem uma visão integrada do processo de promoção de código de Dev, para QA e para Operações?	Painel totalmente integrado disponível desde Dev até Ops, usando ferramentas DevOps de gerenciamento de *release*	5
		Uma visão parcial do processo de promoção do código está disponível, por meio de ferramentas DevOps	4
		Ferramenta caseira ou integrações de algumas ferramentas de código aberto que produzem relatórios de status, não relatórios de análise	3
		Relatórios gerados manualmente, dependente de pessoas, portanto sujeito a erros	2
		Não há disponibilidade de uma visão integrada	1

Fonte: adaptado de IBM (2017) e Incycle (2017)

4.1.3. Avaliação da prática Integração Contínua

Os quadros 6, 7 e 8 do modelo permitem avaliar a prática Integração Contínua (IC)por intermédio da medição dos constructosGerenciamento, Colaboração e Painel de Controle. Recomenda-se que estas questões sejam revistas junto aos desenvolvedores do time.

Quadro 6 — Avaliação do constructo Gerenciamento (IC)

#	Questões	Diretriz	
1	Como o seu código-fonte é gerenciado e como é feito o controle de versão?	Por meio de sistemas de compartilhamento de código-fonte	5
		Por meio de sistema de controle de versão distribuído	4
		Por meio de sistema de controle de versão instalado em um servidor	3
		Por meio de sistemas de arquivos, drives de rede compartilhados e backups	
		Nenhuma das opções acima	1
2	Como um *release* é planejado?	Feito de forma ágil. Monitorado continuamente e atualizado por meio de ferramentas DevOps	5
		Feito de forma ágil, mas executado desarticuladamente e às vezes alguns prazos são perdidos	4
		Há um planejamento iterativo, mas não há um planejamento ágil. Recursos precisam ser adicionados eventualmente, dependendo de onde haja uma crise	3
		Modelo *waterfall*. Os grandes problemas são descobertos muito tarde no ciclo de desenvolvimento	2
		Não há muito planejamento de *release*. As datas precisam ser modificadas frequentemente ou recursos precisam ser adicionados	1
3	O sistema de gerenciamento da configuração de *software* (SCM) está integrado comdesenvolvimento, teste e implantação?	O IDE, as fases de Teste, a Produção e o Sistema de Gerenciamento de Testes são integrados com o sistema de SCM	5
		O gerenciamento de testes não é integrado com o sistema de SCM	4
		As fases de Produção e Teste não são integradas com o sistema de SCM	3
		Apenas Dev IDE está integrado com o sistema de SCM	2
		Não há integração de nenhum sistema com o sistema de SCM	1
4	Com que frequência você mede o débito técnico?	De forma contínua, a cada check-in	5
		Medimos a cada *build*	4
		Medimos a cada *release* de produção	3
		Em um prazo de calendário (mensalmente, etc.) não vinculado aos *releases*	2
		Nós não medimos	1
5	Como a integração contínua é alcançada? Quanto dela é automatizada?	Implementamos, de forma automatizada: integração de código, gerenciamento de *build*,defeitos e configuração, alémde testes unitários	5
		A automação de *build* e a automação de teste unitário são alcançadas, mas o *build* não é desencadeado de forma automática	4
		O *build* é automatizado, mas não há teste unitário nem automação	3
		O *build* não é automatizado e o teste unitário é manual	2
		O *build* é manual, sem teste unitário. Não há integração de *build* e SCM	1

Fonte: adaptado de IBM (2017) e Microsoft (2017)

Quadro 7 — Avaliação do constructo Colaboração (IC)

#	Questões	Diretriz	
6	Como é feita a revisão do código?	Há revisões automatizadas de código usando ferramentas de revisão de código, integrado com o processo de gerenciamento de *build* (SonarQube)	5
		Há revisões automatizadas de código usando ferramentas de revisão de código, mas sem integração com o gerenciamento de *build*	4
		Há um processo manual de revisão do código e monitoramento via ferramenta	3
		Há um processo manual de revisão do código, mas não há um monitoramento	2
		Não há um processo de revisão do código	1
7	Vocês fazem testes durante o desenvolvimento? Existe algum tipo de teste unitário incorporado?	Teste unitário automatizado, com cobertura entre 80 e 90%	5
		Teste unitário automatizado, com cobertura entre 40 e 50%	4
		O teste unitário é manual e a sua cobertura é decidida de acordo com orientação política do cliente	3
		O teste unitário é manual e a sua quantidade depende da performance de desenvolvedores individuais	2
		Não há teste unitário em vigor	1
8	As equipes de desenvolvimento, operações e suporte se reúnem para discutir a melhoria do processo?	Totalmente implementado	5
		N/A	
		Parcialmente implementado	3
		N/A	
		Não implementada	1
9	Vocês trabalham com times dedicados para um projeto ou iniciativa?	Totalmente implementado	5
		N/A	
		Parcialmente implementado	3
		N/A	
		Não implementada	1

Fonte: adaptado de IBM (2017) e Incycle (2017)

Quadro 8 — Avaliação do constructo Painel de Controle (IC)

#	Questões	Diretriz	
10	Como o trabalho é distribuído e controlado?	O trabalho é distribuído e controlado por meio de ferramentas DevOps	5
		O trabalho é distribuído e controlado por meio de ferramentas de planejamento de projetos	4
		O trabalho é distribuído e controlado por meio de planilhas Excel	3
		O trabalho é distribuído e controlado por e-mails	2
		O trabalho é distribuído e controlado verbalmente	1
11	Como você monitora os resultados do *build*? Vocês utilizam notificaçãoautomatizada?	Há uma ferramenta de gerenciamento de *build*, o time de Dev faz uma monitoração do log de *builds*, o log é enviado automaticamente, estão disponíveis painéis de controle e relatórios	5
		O log é gerado, mas não são enviados relatórios sobre os painéis de controle	4
		O log não é coletado, mas o fracasso ou o sucesso do *build* é notificado	3
		O log é monitorado manualmente	2
		Não há uma maneira automatizada para rastrear o log	1
12	Os itens de configuração possuem controle de código fonte?	Todos os artefatos do projeto estão sob controle de versão	5
		Os requisitos não são versionados, mas os demais artefatos são	4
		Os scripts de *build*, implantação e banco de dados não estão sob controle de versão	3
		Não há uma identificação clara dos Cis, mas existe um controle de versão dos sistemas	2
		Não há um critério definido para os Cis e não é usado controle de versão	1
13	Como as mudanças são gerenciadas durante o ciclo de desenvolvimento?	Mudanças gerenciadas por meio de ferramenta de gerenciamento de requisitos, integrada com ferramentas de ciclo de vida colaborativo (CLM)	5
		Mudanças gerenciadas por meio de ferramenta de gerenciamento de requisitos, mas sem integração com ferramentas de ciclo de vida de colaborativo	4
		Mudanças gerenciadas e monitoradas através de um processo manual de gerenciamento de requisitos	3
		Mudanças gerenciadas e monitoradas usando planilhas Excel separadas, de forma não integrada com o processo de gerenciamento de requisitos	2
		Mudanças gerenciadas por e-mail ou verbalmente	1
14	Existem painéis de controle e relatórios apresentados para monitorar o *build*, a revisão de código e os resultados dos testes unitários?	São usadas ferramentas DevOps e painéis de controle e relatórios são exibidos para *build*, revisão de código e testes unitários	5
		Apenas a funcionalidade da ferramenta de *build* é usada para mostrar painéis de controle e relatórios, mas não há um rastreamento completo	4
		É usada uma planilha Excel, que é atualizada manualmente	3
		Apenas os logs de resultados de *build* são mantidos, mas não sob a forma de relatórios ou painéis de controle	2
		É usada apenas a ferramenta SCM. *Build* e revisão de código são manuais, portanto não há painéis de controle	1

Fonte: adaptado de IBM (2017) e Microsoft (2017)

4.1.4. Avaliação da prática Testes Contínuos

Os quadros 9, 10 e 11 do modelo permitem avaliar a prática TestesContínuos (TC) por meio da medição dos constructosGerenciamento, Automação e Painel de Controle. O ideal é que alguém de Controle de Qualidade responda às questões sobre Testes Contínuos.

Quadro 9 — Avaliação do constructo Gerenciamento (TC)

#	Questões	Diretriz	
1	A configuração do seu ambiente de testes é parecida com a do seu ambiente de produção?	Os ambientes de teste são idênticos aos ambientes de produção, incluindo a configuração de hardware e de *software*	5
		Os ambientes de teste são parcialmente semelhantes ao ambiente de produção: a configuração de *software* é idêntica, mas a de hardware não	4
		Nem todos os ambientes de teste são idênticos aos de produção. Alguns são idênticos e outros são similares	3
		Os ambientes de testes possuem configuração similar, mas não idêntica ao ambiente de produção	2
		Os ambientes de teste não são nem parecidos com o ambiente de produção	1
2	O time de desenvolvimento compartilha os casos de testes unitários para cada *release* de teste?	Os casos de teste unitário são automatizados e um painel de controle da cobertura do código está disponível para todos os *releases*	5
		Os casos de teste unitário são executados manualmente pelos desenvolvedores individuais e um relatório consolidado é compartilhado antes de cada *release*	4
		Os casos de teste unitários são executados por desenvolvedores individuais e não há um relatório consolidado disponível	3
		Testes unitários são rodados parcialmente, quando solicitado, e não há casos formais de teste unitário	2
		Não são rodados testes unitários e não existem casos de teste unitário	1
3	Em que fase do ciclo de vida do *software* são criados o plano de teste, os casos de teste e os scripts de teste?	Os casos de teste, o plano de teste e os scripts de teste são escritos durante a fase de design de alto nível (HLD)	5
		Antes do desenvolvimento	4
		Em paralelo com o desenvolvimento	3
		Pouco antes do início dos testes	2
		Não há documentos de teste	1
4	Onde são armazenados o plano de teste, os cases de teste e os scripts de teste?	São usadas ferramentas DevOps para gerenciamento de testes	5
		Em ferramentas SCM com controle de versão	4
		Em repositórios, mas sem controle de versão	3
		São armazenados em um sistema de arquivo local ou em boxes de teste	2
		Não há documentos de teste	1
5	Com que frequência você testa as funcionalidades?	Testes contínuos, de forma ágil	5
		Os testes são feitos por *releases* iterativos	4
		Desafios para fazer testes contínuos devido à disponibilidade de recursos	3
		Os testes são feitos conforme instruções da equipe de desenvolvimento	2
		Todos os requisitos são testados no final do *release* (*waterfall)*	1
6	As equipes de teste se reúnem com Operações e Desenvolvimento para discutir métodos e estratégias?	Totalmente implementado	5
		N/A	
		Parcialmente implementado	3
		N/A	
		Não implementada	1

Fonte: adaptado de IBM (2017) e Incycle (2017)

Quadro 10 — Avaliação do constructo Automação (TC)

#	Questões	Diretriz	
7	Os testes são automatizados ou manuais? Se automatizados, quais são as ferramentas utilizadas?	Os testes e o mecanismo de *feedback* são totalmente automatizados e completamente integrados com as ferramentas de gerenciamento do ciclo de vida colaborativo (CLM)	5
		Os testes funcionais são parcialmente automatizados, pois nem tudo é coberto	4
		Os testes são manuais, mas usando ferramenta de gerenciamento de teste	3
		Os testes são manuais e os resultados registrados no sistema de monitoração de defeitos e / ou em planilhas	2
		Os testes são manuais e os defeitos são registrados em planilhas	1
8	Quais são os vários ambientes de teste?	Ambientes SIT, UAT e Pré-produção virtualizados	5
		Ambientes SIT, UAT e Pré-produção (não virtualizados)	4
		Ambientes SIT e UAT, mas sem Pré-produção	3
		Ambientes SIT e UAT na mesma box	2
		Testes nas boxes dos desenvolvedores	1
9	Qual o percentual de testes automatizados que a sua equipe utiliza nas fases do processo de implantação?	Acima de 80%	5
		Entre 61% e 80%	4
		Entre 41% e 60%	3
		Entre 21% e 40%	2
		Até 20%	1
10	Quando os requisitos de teste não-funcional são obtidos e quando esses testes são realizados?	Os requisitos são coletados bem cedo na fase de Requisito/Design e modificados durante o ciclo de vida do desenvolvimento. Os testes são realizados após o teste unitário e em cada ambiente de teste e são totalmente automatizados	5
		Os requisitos são coletados bem cedo na fase de Requisitos/Design e modificados durante o ciclo de vida do desenvolvimento. Os testes são realizados após o teste unitário e em cada ambiente de teste e não são totalmente automatizados	4
		Os requisitos são coletados apenas durante os testes, que são realizados ao final dos testes funcionais e antes da implantação e são parcialmente automatizados.	3
		Os requisitos são coletados com base na necessidade, os testes são realizados antes da implantação e não são totalmente automatizados	2
		Feito apenas com base na necessidade	1
11	Quanto tempo demora para resolver um*broken build* em um ambiente de pré-produção?	A implantação de *builds* quebrados é impedida por *gated check-in*, política de *pull-request* ou *rollback* automático imediato	5
		Conseguimos resolver em cerca de uma hora	4
		Normalmente resolvemos no mesmo dia	3
		Demora pouco mais que um dia	2
		Demora alguns dias	1

Fonte: adaptado de IBM (2017) e Microsoft (2017)

Quadro 11 — Avaliação do constructo Painel de Controle (TC)

#	Questões	Diretriz	
12	Como o início da execução do teste, em várias fases, é comunicado e monitorado?	Comunicação automatizada com processo de não envolvimento e monitoração do status de teste	5
		Comunicação automatizada a partir das ferramentas, com uma lista de distribuição, mas sem painel de controle e sem relatório de monitoração	4
		Acionamento de e-mail automático, a partir da ferramenta de *build*, mas sem monitoração	3
		Comunicação manual através de e-mail	2
		Comunicação verbal em reuniões	1
13	Como você monitora quais *builds* são implantados no ambiente de teste?	O processo de gerenciamento de *release* controla as implantações automaticamente. Os *builds* podem ser acompanhados no painel de controle	5
		Builds e ambientes são controlados usando planilhas, com controle de versão e mantidas no repositório de SCM	4
		Builds e ambientes são monitorados via planilhas e/ou através de e-mail	3
		As implantações dos *builds* são comunicadas por e-mail	2
		Não existe um processo de gerenciamento de *release*	1
14	Qual ferramenta é usada para gerenciamento de defeitos? Como eles são capturados e monitorados até o seu encerramento?	Os defeitos são capturados utilizando ferramentas (processo automatizado) e as alterações são movidas para a fase seguinte, quando todos os defeitos são fechados	5
		Os defeitos são capturados utilizando ferramentas, mas sem métricas automatizadas, pois são geradas manualmente, usando planilhas	4
		Os defeitos são identificados em planilhas, mas não são geradas métricas	3
		Os defeitos são identificados individualmente pelos testadores em planilhas	2
		Não utiliza ferramenta nem planilha e os defeitos são comunicados por e-mail	1
15	Sua equipe consegue simular as solicitações de serviços da aplicação para testes de carga?	Totalmente implementado	5
		N/A	
		Parcialmente implementado	3
		N/A	
		Não implementada	1
16	Como você identifica os requisitos para os casos de teste?	Identificação usando uma ferramenta DevOps	5
		Identificação usando planilhas com métricas e painel de controle	4
		Identificação via planilhas, mas sem painel de controle ou métricas	3
		Há algum gerenciamento de requisitos, mas não há identificação	
		Não há gerenciamento de requisitos nem gerenciamento de testes	1
17	Quem fornece / cria as bases de dados de teste e o quão próximo elas estão do conjunto de dados de produção?	O time de desenvolvimento cria e fornece as bases de dados de teste, que são similares às de produção	5
		O time de desenvolvimento fornece a base de dados de teste, mas não é similar à de produção	4
		O time de testes cria a sua própria base de dados, que está em linha com os dados de produção	3
		O time de testes cria a sua própria base de dados, que não está em linha com os dados de produção	2
		Os testes são feitos caso a caso e a base de dados de teste é criada pelo testador, durante o teste	1

Fonte: adaptado de IBM (2017) e Incycle (2017)

4.1.5. Avaliação da prática Entrega Contínua

Os quadros 12, 13 e 14 do modelo permitem avaliar a prática EntregaContínua (EC)por intermédio da medição dos constructosGerenciamento, Automação e Painel de Controle. O ideal é que estas questõessejam respondidas por desenvolvedores e pessoas de Operações do time.

Quadro 12 — Avaliação do constructo Gerenciamento (EC)

#	Questões	Diretriz	
1	Como é a implantação é feita atualmente?	Implantação automática usando ferramentas DevOps como UrbanCode ou Chef	5
		Implantação usando ferramentas de *build* como Jenkins, masbaseada em scripts	4
		Implantação baseada em scripts,sem uma ferramenta de implantação de *build*	3
		Implantação semiautomatizada, usando procedimentos manuais e scripts	2
		Implantação manual	1
2	A sua equipe é capaz de atender rapidamente a demandas de mudanças do negócio?	Totalmente implementado	5
		Parcialmente implementado	3
		Não implementada	1
3	Como é o seu processo de gerenciamento de *release*?	O processo de gerenciamento de *release* é feito usando ferramentas (JIRA, RTC), com rastreabilidade e monitoração	5
		O processo de gerenciamento de *release* é feito usando ferramentas como (JIRA, RTC), mas sem rastreabilidade até a produção	4
		O processo de gerenciamento de *release* é feito usando planilha Excel, com rastreabilidade	3
		O processo de gerenciamento de *release* é manual, usando planilha Excel, feito sem rastreabilidade	2
		Não há um processo adequado de gerenciamento de *release*	1
4	Quais são os conteúdos da nota de *release* que são compartilhados com o time de operações?	A nota de *release* é automaticamente proveniente da ferramenta de *release*, o time de operações é envolvido e fica ciente desde o início	5
		A nota de *release* é criada em Word ou Excel com o número do *build*, instruções de implantação, detalhes do ambiente e números das histórias	3
		As notas de *release* com instruções de implantação são enviadas por e-mail com detalhes decididos por cada indivíduo	2
		Nenhuma nota de *release* é criada e as instruções de implantação são enviadas por e-mail	1
5	Com que frequência vocês implantam novas funcionalidades em produção?	É possível implantar uma ou mais vezes por dia	5
		Pelo menos uma vez por semana	4
		Pelo menos uma vez a cada duas semanas	3
		Pelo menos uma vez por mês	2
		Demora mais de um mês para conseguirmos implantar um novo *release*	1
6	Como você mantêm a compatibilidade entre sistema operacional, middleware, banco de dados e demais componentes?	A compatibilidade é mantida usando padrões e ferramentas DevOps, como Puppet ou Chef ou UrbanCode	5
		São usadas imagens virtualizadas para manter a compatibilidade, mas é demorado e caro	4
		A compatibilidade é mantida via planilha Excel e as instalações são manuais	2
		De forma manual e feita caso a caso	1

Fonte: adaptado de IBM (2017), Incycle (2017) e Microsoft (2017)

Quadro 13 — Avaliação do constructo Automação (EC)

#	Questões	Diretriz	
7	Quão replicável é o seu processo de *release*?	Os processos de implantação e de *release* são totalmente replicáveis, medidos e continuamente melhorados.	5
		Os processos de implantação e de *release* são totalmente replicáveis, para todos os tipos de *releases*	4
		Os processos de implantação e de *release* são replicáveis, exceto para grandes mudanças	3
		O processo de implantação é replicável, mas o processo de *release* é coordenado manualmente	2
		Release e implantação são gerenciados como atividades separadas, com reprodutibilidade muito limitada	1
8	Quando a equipe de Dev quer atualizar um *release*, o que eles fazem? (escolha todos que se aplicam)	Um pipeline de *release* automático garante que cada check-in pode ser enviado para *release* de forma segura	5
		O controle de exposição irá testar os novos recursos na produção e enviará progressivamente mais utilização para a nova versão, com base em seu sucesso	4
		Portais de qualidade manuais asseguram que ocorra a correta aprovação gerencial	3
		Segue o procedimento documentado de QA para aprovar alterações	2
		Nenhuma das opções acima	1
9	Quanto tempo leva para fazer uma mudança de uma linha e implantar o novo *software*, para corrigir um problema crítico de produção?	Menos de uma hora	5
		Algumas horas, mas menos que um dia	4
		Alguns dias, mas menos que uma semana	3
		Mais de uma semana	2
		Não se aplica	1
10	Vocês utilizam as técnicas de *Canary Release* ou *ToggledFeatures* no processo de implantação?	Usamos ambas as técnicas	5
		Usamos apenas *Canary Release*	4
		Usamos apenas *ToggledFeatures*	3
		Não usamos nenhum dos dois	2
		Não conheço estes conceitos	1
11	Vocês utilizam a técnica de *Blue-Green Deployment*?	Usamos em todos os nossos projetos	5
		Usamos em alguns projetos	4
		Estamos tentando usar, mas ainda está em fase bem inicial	3
		Não usamos	2
		Não conheço este conceito	1
12	Você é capaz de lidar facilmente com grandes implantações complexas em larga escala?	Totalmente implementado	5
		Parcialmente implementado	3
		Não implementada	1
13	Vocês conseguem implantar novas versões de aplicações para produção sem tempo de inatividade?	Totalmente implementado	5
		Parcialmente implementado	3
		Não implementada	1
14	As suas equipes estão preparadas para lidar com um *rollback* da implantação?	Totalmente implementado	5
		Parcialmente implementado	3
		Não implementada	1

Fonte: adaptado de IBM (2017) e Incycle (2017)

Quadro 14 — Avaliação do constructo Painel de Controle (EC)

#	Questões	Diretriz	
15	Todas as partes interessadas têm visibilidade sobre asatividades de *release*?	Totalmente implementado	5
		Parcialmente implementado	3
		Não implementada	1
16	Qual é o processo de aprovação de implantação com os critérios de entrada e saída?	Processo automatizado usando ferramentas DevOps de gerenciamento de *release* e implantação	5
		Processo automatizado usando ferramenta de gerenciamento de incidentes	4
		Processo semiautomatizado	3
		Processo manual para copiar os artefatos para os ambientes; a equipe operações é informada verbalmente	2
		Sem processo de aprovação e colaboração manual	1
17	Como você coleta pró-ativamente os desvios do calendário de *release* e cria um *feedback* para trazê-los de volta ao caminho certo?	Usamos ferramentas como RTC ou JIRA: desvios mostrados em relatórios de forma dinâmica, notificações enviadas antecipadamente e ajustes feitos a tempo	5
		Usamos ferramentas como RTC ou JIRA e o relatório mostra os desvios	4
		O calendário de *release* é monitorado regularmente e os desvios são capturados com antecedência	3
		O calendário de *release* existe em planilha Excel e os desvios são discutidos durante a reunião de status	2
		Através de uma abordagem reativa	1
18	Como você monitora os itens de configuração? (por exemplo: número do *build* ou *release*, versão da implantação)	Temos relatórios de monitoração automáticos gerados usando integração de ferramentas DevOps	5
		N/A	
		A monitoração é feita com base em scripts e o processo é semiautomático	3
		A monitoração é feita através de planilha Excel, mas de forma manual	2
		Não é feita monitoração	1
19	Qual o nível de controle que a sua empresa tem sobre os processos de *release*?	O nosso processo de *release* é totalmente automatizado por ferramentas DevOps e conseguimos monitorar e garantir o cumprimento	5
		O nosso processo de *release* é em grande parte automatizado, com a capacidade de monitorar o cumprimento, mas ainda é necessário algum envolvimento manual	4
		N/A	
		Temos um processo documentado para as pessoas usarem e verificamos manualmente as suas condições	2
		Nossos processos de *release* são feitos caso a caso	1
20	Com relação à visibilidade do processo de *release*, como os dados de *release* da sua empresa são reunidos e disponibilizados para a equipe?	Automatizada e com métricas em todo o processo de *release*: a visibilidade de alto nível é possível através de um hub central com visão geral sobre análise de pipeline e com capacidade de detalhamento em pontos específicos	5
		Temos um processo definido para como a informação sobre todos os *releases* é coletada e organizada em um local central, com informações sobre os níveis de qualidade, status de *releases* e dependências	4
		N/A	
		De forma estruturada, mas manualmente e falta uma visão geral sobre o processo de *release*	2
		De forma desestruturada	1

Fonte: adaptado de IBM (2017), Incycle (2017) e Xebialabs (2017)

4.1.6. Avaliação da prática Infraestrutura como Código

Os quadros 15, 16 e 17 do modelo visam avaliar a prática Infraestrutura como Código (IcC) e permitem medir os constructosGerenciamento, Automação e Painel de Controle. Recomenda-se que estas questões sejam respondidas por pessoas de Operações do time.

Quadro 15 — Avaliação do constructo Gerenciamento (IcC)

#	Questões	Diretriz	
1	Qual é a sua política quanto ao uso de nuvem pública?	Escolhemos um fornecedor que nos dê segurança e qualidade de serviço; permitimos usar a nuvem, desde que os dados não sejam postos em risco	5
		N/A	
		As equipes são livres de decidir por conta própria	3
		N/A	
		Não permitimos o uso de nuvem pública	1
2	Vocês utilizam serviços de nuvem (PaaS ou SaaS)?	Usamos alguns SaaS	5
		Sim, usamos como PaaS	4
		Sim, usamos como IaaS	3
		N/A	
		Não usamos nuvem	1
3	Quem toma as decisões sobre como e onde as aplicações serão hospedadas?	Temos fornecedores preferenciais e políticas em torno das definições de rede, VMs e opções de IaaS ou PaaS	5
		Cada projeto ou unidade de negócio pode decidir por si mesmo	4
		N/A	
		Todas as decisões sobre hospedagem precisam de uma revisão de segurança e aprovação para cada aplicação	2
		Todos os projetos são hospedados no nosso datacenter	1
4	Qual é a sua política em relação ao *Software as a Service* (SaaS)?	Nós encorajamos SaaS, onde ele se adapte às necessidades de negócios e esteja em conformidade com as diretrizes da empresa	5
		Nós temos uma lista de fornecedores preferenciais de SaaS	4
		SaaS precisa passar por análise gerencial e aprovação, caso a caso	3
		Raramente usamos SaaS e não temos uma política definida quanto ao seu uso	2
		Não usamos SaaS	1
5	Suas equipes estão cientes de dependências de terceiros para garantir uma prestação de serviços de confiável?	Totalmente implementado	5
		N/A	4
		Parcialmente implementado	3
		N/A	2
		Não implementada	1
6	Vocês fazem implantação de *software* em máquinas virtuais para ambientes diferentes (testes, desenvolvimento) e com controle de versão dos ambientes?	Os ambientes são criados de forma flexível e controlado, sendo as versões de *software* iguais às de produção	5
		Os ambientes são criados de forma flexível, mas as versões do *software* nem sempre são controladas e nem sempre simulam o ambiente real de produção	4
		Sim, mas é preciso intervenção da equipe de operações para que possa acontecer uma implantação	3
		Sim, mas os ambientes criados são geralmente diferentes do ambiente de produção	2
		Não	1

Fonte: adaptado de Incycle (2017) e Microsoft (2017)

Quadro 16 — Avaliação do constructo Automação (IcC)

#	Questões	Diretriz	
7	Qual é o nível de automação do seu processo de *release* e implementação?	Processo de implantação e *release* totalmente automatizado	5
		Processo de implantação automatizado, mas não o de *release*	4
		N/A	3
		Implantação de novos *releases* por meio de scripts	2
		O provisionamento de novos *releases* é feito de forma manual	1
8	Qual é o nível de automação do seu processo de testes?	Os processos de negócio são testados de forma automatizada entre aplicativos e interfaces, de ponta a ponta	5
		Os testes de aplicação são amplamente automatizados	4
		Os testes são automatizados no nível de UI para aplicações, com cobertura média	3
		Os testes unitários de componentes individuais são bastante automatizados por desenvolvedores, mas os testes no nível de UI são feitos manualmente	2
		Todos os testes são executados manualmente. Oprocesso de teste é feito caso a caso e não é reproduzível	1
9	Qual é o nível de automação do seu processo de *build*?	*Builds* são totalmente automatizados, com testes unitários, análise de código estático, empacotamento e correções automatizados.	5
		Builds são totalmente automatizados, com testes unitários, análise de código estático e empacotamento automatizados	4
		Builds totalmente automatizados, mas a validação automática é limitada	3
		Builds noturnos são automatizados, mas não validados. O empacotamento para a implantação de teste e feito de forma automatizada.	2
		Builds são geralmente empacotados manualmente e validados. O empacotamento é feito manualmente.	1
10	Qual é o nível de automação do seu processo de provisionamento do ambiente?	O processo de provisionamento do ambiente é totalmente automatizado e orquestrado via ferramenta DevOps	5
		Provisionamento de infraestrutura básica, sistema operacional e *middleware* é feito de forma automatizada, mas não como um único processo orquestrado	4
		Provisionamento de infraestrutura básica e sistema operacional é feito de forma automatizada, mas a instalação do middleware e a configuração da aplicação exigem etapas manuais separadas	3
		Provisionamento de VM's é feito por meio de scripts, mas o encadeamento das tarefas exige coordenação manual	2
		Não é automatizado	1
11	Qual é o tempo necessário para disponibilizar o ambiente?	Ambiente em cloud e cerca de 2 a 4 horas	5
		Ambiente virtualizado e cerca de 4 a 6 horas	4
		Ambiente virtualizado e cerca de 8 horas	3
		Ambiente virtualizado (VM), mas demora cerca de 24 horas	2
		Ambiente em servidores físicos, por isso pode levar cerca de 4 a 6 semanas	1
12	Como você especifica os ambientes para desenvolvimento, teste, homologação e produção?	Usamos infraestrutura como código (Puppet, Chef)	4
		PaaS manuseia as imagens de produção, de modo que não precisamos controlar configurações individuais	3
		Laboratórios dedicados com ambientes virtualizados completos, incluindo papéis (roles) e rede (network)	2
		Máquinas virtuais individuais dedicadas	1
		Máquinas físicas individuais dedicadas	0

Fonte: adaptado de Automic (2017) e Microsoft (2017)

Quadro 17 — Avaliação do constructo Painel de Controle (IcC)

#	Questões	Diretriz	
13	Durante o planejamento dos *releases*, a disponibilidade do ambiente precisa ser levada em consideração?	Não, porque os ambientes são baseados em nuvem e disponibilizados sob demanda	5
		As VMs são virtualizadas, portanto o planejamento de *release* é pouco afetado pela disponibilidade do ambiente	4
		Este problema é gerenciado por meio de boa coordenação entre Dev e Ops	3
		Sim, o planejamento de *release* pode ser afetado pela disponibilidade do ambiente	2
		Sim, o planejamento de *release* é bastante afetado pela disponibilidade do ambiente	1
14	Quais dados são coletados de aplicativos em execução em produção? (todos que se aplicam)	Teste A/B	5
		Utilização com relação à funcionalidade	4
		Disponibilidade	3
		Performance	2
		Não coletamos nenhum dos dados acima citados	1
15	As suas equipes são capazes de fornecer um provisionamento *self-service* e sob demanda para todos os recursos?	Totalmente implementado	5
		N/A	4
		Parcialmente implementado	3
		N/A	2
		Não implementada	1
16	Como você mantém iguais os ambientes de teste e de produção?	Ferramentas DevOps são usadas para preparação de dados e tanto o ambiente quanto os dados de teste são exatamente iguais aos de produção	5
		Os bancos de dados são copiados antes de cada execução de teste, mas possuem limitação de tamanho.	4
		Bancos de dados são sincronizados usando scripts	3
		Os ambientes são os mesmos, exceto bancos de dados, que são sincronizados de tempos em tempos	2
		Não há uma estratégia predefinida. Os ambientes não são mantidos iguais.	1
17	Como os problemas na produção são informados à equipe de desenvolvimento? (todos que se aplicam)	Por meio de ferramentas de acompanhamento de problemas, disponível para todas as equipes em tempo real	5
		A instrumentação gera alertas automatizados para todas as equipes afetadas	4
		Por meio de ferramentas de *servicedesk*, relatados separadamente para a equipe de Desenvolvimento	3
		Por solicitações/reclamações dos usuários finais (via telefone, e-mail, *help desk*, gerentes de conta)	2
		Nenhuma das acima	1

Fonte: adaptado de IBM (2017), Incycle (2017) e Microsoft (2017)

4.1.7. Avaliação da prática Monitoração Contínua

Os quadros 18, 19 e 20 do modelo visam avaliar a prática Monitoração Contínua (MC)e objetivam medir os constructosGerenciamento, Capacidade e Performance. Recomenda-se que estas questões sejam respondidas por pessoas de Operações do time.

Quadro 18 — Avaliação do constructo Gerenciamento (MC)

#	Questões	Diretriz	
1	Quais são as ferramentas usadas para a monitoração?	Usamos uma ferramenta DevOps para a monitoração, que está integrada com os processos e ferramentas de desenvolvimento	5
		N/A	
		Temos scripts que são capazes de criar painéis de controle	3
		N/A	
		Não usamos ferramentas de monitoração	1
2	Como é feito o gerenciamento de log?	Usando uma ferramenta de gerenciamento de log, que é integrada com aplicações e infraestrutura e permite analisar e gerar relatórios	5
		N/A	
		Temos hospedagem de log centralizada para várias aplicações e infraestrutura	3
		N/A	
		Não temos uma hospedagem de log centralizada	1
3	Como os defeitos são reportados para a equipe de desenvolvimento?	Por meio de vídeo de interação do usuário ou log de ações com gestos do usuário	5
		Por não passar nos casos de teste ou por não atender aos critérios de aceitação	4
		Por meio de dados relevantes de arquivo de log, incluídos automaticamente no relatório de incidente	3
		Á medida em que são reportados pelo usuário (ou pela parte interessada)	2
		Nenhuma das opções acima	1
4	As equipes têmvisão unificada, quepermita identificar rapidamente os incidentes e fazer uma triagem?	Totalmente implementado	5
		N/A	
		Parcialmente implementado	3
		N/A	
		Não implementada	1
5	Como você faz o gerenciamento de eventos e incidentes?	Gerenciamento de incidentes usando ferramentas DevOps, com integração com os processos de *release* e desenvolvimento	5
		Gerenciamento de incidentes usando ferramentas (Remedy ou CQ), com integração manual com os processos de *release* e desenvolvimento	4
		Gerenciamento de incidentes usando ferramentas como (Remedy ou CQ), mas sem integração automatizada com o processo de *release*	3
		Os incidentes são comunicados por e-mail e registrados em planilha	2
		Não há um processo de gerenciamento de incidentes	1
6	Vocês utilizam algum sistema degerenciamento do controle de mudanças, de forma a permitir rastreabilidade e auditoria?	Sim, os *releases* são gerados automaticamente a partir da ferramenta de controle de mudanças, com rastreabilidade total	5
		N/A	
		Não temos um sistema de rastreamento, mas usamos ferramentas de controle de mudanças	3
		N/A	
		Não usamos ferramentas de controle de mudanças	1

Fonte: adaptado de IBM (2017), Incycle (2017) e Microsoft (2017)

Quadro 19 — Avaliação do constructo Capacidade (MC)

#	Questões	Diretriz	
7	As equipes têm visibilidade sobre a capacidade do sistema, a fim de compreender o impacto de um aumento de carga?	Totalmente implementado	5
		N/A	
		Parcialmente implementado	3
		N/A	
		Não implementada	1
8	A equipe de operações é envolvida desde a fase de análise de requisitos para opinar sobreambientes,implantação eplanejamento de capacidade?	O time de operações é envolvido ativamente nos encontros de planejamento	5
		N/A	
		O time de operações é envolvido apenas para prover informações	3
		O time de operações é envolvido apenas quando necessário, caso a caso	2
		O time de operações não é envolvido na fase de análise dos requisitos	1
9	Qual é o tempo necessário para aumentar a capacidade do ambiente uma vez que o limite foi atingido?	Sob demanda, em menos de 2 horas	5
		Menos de uma semana	4
		De 2 a 4 semanas	3
		De 4 a 8 semanas	2
		De 8 a 12 semanas	1
10	Qual é o processo para planejamento de capacidade?	Usamos uma ferramenta de gestão de capacidade para monitorar a tendência atual e prever o tempo estimado em que podemos ficar sem capacidade	5
		N/A	
		Usamos scripts para coletar dados históricos e prever o tempo estimado em que podemos ficar sem capacidade	3
		N/A	
		Não há um processo definido	1
11	Sua infraestrutura é autoescalável?	Autoescalabilidade automática para expansão e redução da infraestrutura, com base em parâmetros configuráveis, como CPU/memória	5
		A infraestrutura é escalável, mas a expansão e a redução são feitas manualmente, com base em parâmetros de limite	4
		N/A	
		Não há espaço para a expansão. A Infraestrutura precisa ser implementada.	2
		A aplicação não pode escalar	1

Fonte: adaptado de IBM (2017) e Incycle (2017)

Quadro 20 — Avaliação do constructo Performance (MC)

#	Questões	Diretriz	
12	Quais são os limites de desempenho definidos e como eles são monitorados?	Monitoração automatizada e rápido tempo de resposta para escalar o ambiente.	5
		Monitoração automatizada usando ferramenta, mas com limites definidos	4
		Limites definidos e monitoração baseada em scripts	3
		Sem limites, mas com monitoração manual proativa	2
		Não há limites definidos e a abordagem é reativa	1
13	Como você monitora suas aplicações quanto a erros, alertas e desempenho?	Monitoração usando ferramentas DevOps, com *feedback* automatizado para o time de desenvolvimento	5
		Monitoração usando ferramentas como o Nagios e Tivoli, mas sem integração com as ferramentas de desenvolvimento	4
		Monitoração através de scripts	3
		Monitoração manual do log e comunicação para o time de desenvolvimento de forma manual	2
		Não há monitoração	1
14	As equipes são capazes de analisar métricas de performance para solucionar problemas antes que eles afetem os usuários?	Totalmente implementado	5
		N/A	
		Parcialmente implementado	3
		N/A	
		Não implementada	1
15	Sua aplicação permite alta disponibilidade (HA)?	A aplicação está configurada para HA e o procedimento de contingência é automático, sem perda de dados	5
		N/A	
		A aplicação está configurada para HA, mas o procedimento de contingência é manual, com perda acordada de transações	3
		N/A	
		A aplicação não está configurada para HA	1

Fonte: adaptado de IBM (2017) e Incycle (2017)

4.1.8. Avaliação da prática Feedback Contínuo

Os quadros 21 e 22 do modelo permitem avaliar a prática Feedback Contínuo (FC) e visam medir os constructosFeedback e Otimização. Recomenda-se que estas questões sejam respondidas por alguém responsável pela interface com o usuário final.

Quadro 21 — Avaliação do constructo Feedback (FC)

#	Questões	Diretriz	
1	Existe alguma análise, orientação ou relatório produzido para criar um *feedback* para a equipe de desenvolvimento?	Usamos uma ferramenta analítica e também provemos *feedback* para melhoria contínua	5
		São usadas ferramentas de análise, mas o *feedback* não é usado para a melhoria contínua.	4
		A geração de alertas é baseada em scripts, mas não há análises	3
		Tanto a alimentação de dados quanto a criação de relatórios são manuais	2
		Não há relatórios	1

#	Questões	Diretriz	
2	Como você coleta o *feedback* do usuário interno?	É utilizada uma ferramenta DevOps de *feedback* com processos de planejamento contínuo de negócios	5
		Ferramentas como Remedy são usadas para coletar *feedback* online do usuário	4
		É usada planilha Excel com acompanhamento do histórico	3
		O *feedback* do usuário é comunicado por e-mail pelas partes interessadas	2
		Não há um processo formal. O *feedback* do usuário é transmitido pelo cliente para a equipe de gerenciamento	1
3	Como você coleta o *feedback* cliente final?	Pela monitoração de mídias sociais e blogs ou através de tickets (*servicedesk)*	5
		Abordamos o cliente via UserVoice, reuniões webou sala de bate-papo	4
		Telemetria de utilização	3
		Através de contato direto ou através da equipe de vendas	2
		Nenhuma das opções acima	1
4	Quais são as ferramentas utilizadas para coletar *feedback*?	Serviço de coleta de *feedbackon-line* integrado, juntamente com ferramentas	5
		Ferramentas como Remedy, RTC ou CQ usadas independentemente do usuário final	4
		Baseado em planilha, com o histórico monitorado na forma de incidentes	3
		Baseado em planilha e com manutenção manual	2
		Os incidentes são gerenciados através de comunicação por e-mail	1
5	Quando o *feedback* é coletado?	Existe a opção de prover *feedback* a qualquer momento, durante todas as fases.	5
		Durante crises, pós-implantação, testes e desenvolvimento	4
		Durante crises, pós-implantação e testes	3
		Apenas durante crises e pós-implantação.	2
		Somente quando há uma crise	1
6	Como você mede a aceitação / adequação ao propósito? (todas que se adequam)	Através da coleta de telemetria da utilização, com o monitoramento de usuários reais	5
		Através de *feedback* qualitativo do produto (por exemplo: "me mande um sorriso")	4
		Através de laboratórios de usabilidade ou testes exploratórios	3
		Através de testes manuais ou critérios gerais de aceitação	2
		Nenhuma das opções acima	1
7	O que você faz com os dados de utilização de suas aplicações? (todasque se adequam)	Os dados de utilização são combinados com os dados de performance e avaliados para cada *release*	5
		Os dados de utilização ajudam a esclarecer sobre os próximos passos no *backlog* de produtos	4
		Os dados de utilização são monitorados pelo departamento de Marketing	3
		Os dados de utilização são inferidos por meio de pedidos de funcionalidades e relatos de erros	2
		Os dados de utilização não são monitorados	1
8	Como você garante que a qualidade do serviço ou os requisitos não funcionais são atendidos? (todas que se adequam)	Nós automatizamos todos os testes para: desempenho, segurança, localização e outras qualidades de serviço	5
		Nós temos uma definição consistente de feito (*done*)	4
		Testamos na produção contra os nossos serviços em execução para garantir que eles funcionam para os usuários reais e não podem ser violados	3
		Nós anotamos todos os requisitos	2
		Nenhuma das opções acima	1
9	Como você valida designs de experiência do usuário? (todas que se adequam)	Através de testes A/B em produção	5
		Através de revisão via Storyboard com os usuários internos (negócio)	4
		Através de revisão via Storyboard com o proprietário do produto e/ou com as partes interessadas	3
		N/A	
		Através de revisão escrita da especificação ou do documento	1

Fonte: adaptado de IBM (2017), Incycle (2017) e Microsoft (2017)

Quadro 22 — Avaliação do constructo Otimização (FC)

#	Questões	Diretriz	
10	Como o *feedback* do usuário é analisado e otimizado para melhoria contínua?	São usadas ferramentas de análise e *feedback* otimizado para melhoria contínua	5
		São usados relatórios padrão de ferramentas de *feedback* e o resultado é otimizado	4
		São usadas técnicas como RCA, entrevistas, métricas, filtros de dados	3
		Através de discussões da equipe, com base em dados de planilha Excel	2
		Não há otimização para melhoria contínua	1
11	Como os pontos críticos obtidos do *feedback* são convertidos em itens de ação?	Via ferramenta DevOps, integrada com o planejamento de negócios	5
		N/A	
		Usando planilhas Excel	3
		N/A	
		Não coletamos *feedback* de forma consistente	1
12	As suas equipes conseguem acessar facilmente dados analíticos de comportamento e utilização?	Totalmente implementado	5
		N/A	
		Parcialmente implementado	3
		N/A	
		Não implementada	1
13	Você automatiza o *feedback* de performance crítica para otimizar ainda mais aplicações e serviços?	Totalmente implementado	5
		N/A	
		Parcialmente implementado	3
		N/A	
		Não implementada	1
14	Existe um mecanismo de *feedback* entre equipes? Como isto é praticado?	*Feedbacks* são compartilhados durante os *releases*, e são monitorados	5
		Feedbacks são compartilhados durante os *releases*, mas não são monitorados	4
		Feedbacks são compartilhados periodicamente	3
		Feedbacks são compartilhados apenas quando ocorrem problemas	2
		Os *feedbacks* não são compartilhados	1

Fonte: adaptado de IBM (2017) e Incycle (2017)

5. Conclusão

Este trabalho objetivou propor um modelo de avaliação da maturidade DevOps nas empresas. Para tanto, foi estruturada uma teoria, com base na revisão da literatura, que buscou relacionar as principais práticas DevOps, a partir das quais foi criado um modelo para avaliação da maturidade DevOps nas organizações. Esta proposta partiu dos constructos da pesquisa, extraídos do referencial teórico, que são justamente os objetos a serem mensurados por este modelo; isto ocorre por meio de uma pontuação, que indica se a unidade de análise avaliada pode ser considerada como mais ou menos aderente a uma abordagem DevOps no desenvolvimento e implementação de aplicações.Ao final de um trabalho de campo, não

descrito neste artigo, foi possível verificar que o modelo proposto para avaliação da maturidade DevOps mostrou-se viável e de fácil replicação, desde que se tenha um conhecimento básico quanto às práticas e técnicas utilizadas nesta abordagem de desenvolvimento de software. Foi possível concluir que este modelo consegue abordar os pontos mais críticos, por meio de uma análise detalhada, cuja estruturação em formato de perguntas com respostas graduadas permite facilitar o trabalho, servindo como um guia para manter o pesquisador no caminho correto.Os constructos escolhidos também se mostraram adequados ao propósito de viabilizar uma medição operacional de conceitos relativamente subjetivos. Portanto, a principal contribuição deste trabalho consiste no modelo de avaliação da maturidade DevOps, que pode ser utilizado por diversas empresas, para mapear pontos de melhoria na implementação desta abordagem de desenvolvimento de software.

Referências

AUTOMIC. *DevOps Maturity Assessment*. Disponível em: <https://automic.com/devops-maturity-assessment>. Acesso em: 5 nov. 2017.

AZOFF, Michael. Ovum Decision Matrix: Selecting a DevOps Release Management Solution, 2016–17. *Ovum Software Solutions*, mar. 2016. Disponível em: <https://www-01.ibm.com/marketing/iwm/dre/signup?source=ibm-cloud-weborganic&S_PKG=ov47603&dynform=22003>. Acesso em: 5nov. 2017.

BRAGA, Filipe. *Um Panorama sobre o uso de Práticas DevOps nas Indústrias de Software*. 2015. 123 f. Dissertação (Mestrado)—Curso de Pós-graduação em Ciência da Computação, Centro de Informática da UFPE, Universidade Federal de Pernambuco, Recife, 2015. Disponível em: <http://repositorio.ufpe.br/handle/123456789/15989>. Acesso em: 5nov. 2017.

DUVALL, Paul; MATIAS, Stephen; GLOVER, Andrew. *Continuous Integration:* improving software quality and reducing risk. New Jersey: Addison-Wesley, 2007.

ERICH, Floris; AMRIT, Chintan; DANEVA, Maya. *Report*: DevOps Literature Review. University of Twente, out. 2014. Disponível em:

<https://www.researchgate.net/profile/Chintan_Amrit/publication/267330992_Report_DevOps_Literature_Review/links/544ba33f0cf2bcc9b1d6bd8a.pdf>. Acesso em: 5 nov. 2017.

FARROHA, Bassam; FARROHA, Debora. A Framework for Managing Mission Needs, Compliance and Trust in the DevOps Environment. *IEEE Military Communications Conference*, p. 288–293, out. 2014. Disponível em: <http://ieeexplore.ieee.org/document/6956773/references>. Acesso em: 5 nov. 2017.

FEITELSON, Dror; FRACHTENBERG, Eitan; BECK, Kent. Development and Deployment at Facebook. *IEEE Internet Computing*, v. 17, n. 4, p. 8–17, jul. 2013. Disponível em: <http://ieeexplore.ieee.org/abstract/document/6449236/>. Acesso em: 5nov. 2017.

FRANÇA, Breno de; JERÔNIMO JUNIOR, Hélvio; TRAVASSOS, Guilherme. Characterizing DevOps by Hearing Multiple Voices. *Procedings of the 30th Brazilian Symposium on Software Engineering*, p. 53–62, set. 2016. Disponível em: <http://dl.acm.org/citation.cfm?id=2973845>. Acesso em: 5 nov. 2017.

GOTTESHEIM, Wolfgang. Challenges, Benefits and Best Practices of Performance Focused DevOps. *Proceedings of the 4th International Workshop on Large-Scale Testing*, p. 3, fev. 2015. Disponível em: <http://dl.acm.org/citation.cfm?id=2693187>. Acesso em: 5 nov. 2017.

HAMUNEN, Joonas. *Challenges in Adopting a Devops Approach to Software Development and Opertions*. 2016. 69 f. Dissertação (Mestrado)—MSc program in Information and Service Management, Aalto University, Helsinki, 2016. Disponível em: <https://aaltodoc.aalto.fi/handle/123456789/20766>. Acesso em: 5nov. 2017.

HERNANTES, Josune; GALLARDO, Gorka; SERRANO, Nicolás. IT Infrastructure-Monitoring Tools. *IEEE Software*, v. 32, n. 4, p. 88–93, ago. 2015. Disponível em: <https://www.computer.org/csdl/mags/so/2015/04/mso2015040088.pdf>. Acesso em: 5 nov. 2017.

HUMBLE, Jez; FARLEY, David. *Continuous Delivery*: reliable software releases through build, test and deployment automation. New Jersey: Addison-Wesley, 2010.

HUMBLE, Jez; MOLESKY, Joanne. Why enterprises must adopt DevOps to enable continuous delivery. *Cutter IT Journal*, v. 24, n. 8, p. 6–12, ago. 2011. Disponível em: <https://www.cutter.com/sites/default/files/itjournal/fulltext/2011/08/itj1108.pdf>. Acesso em: 5 nov. 2017.

HÜTTERMANN, Michael. *DevOps for developers*: integrate development and operations, the agile way. New York: Apress, 2012.

IBM. *IBM DevOps Practices Self-assessment*. Disponível em: <https://www-01.ibm.com/marketing/iwm/iwm/web/signup.do?source=swg-rtl-sd-calc&S_PKG=ov24988>. Acesso em: 5nov. 2017.

INCYCLE. *Online DevOps Assessment*. Disponível em: <http://incyclesoftware.com/devops-assessment/>. Acesso em: 5 nov. 2017.

JABBARI, Ramtin et al. What is DevOps? A systematic mapping study on definitions and practices. *ACM Digital Library*, maio 2016. Disponível em: <http://dl.acm.org/citation.cfm?id=2962707>. Acesso em: 5 nov. 2017.

LIU, Yuhong; LI, Chengbo; LIU, Wei. Integrated Solution for Timely Delivery of Customer Change Requests: a case study of using DevOps approach. *International Journal of u- and e-Service, Science and Technology*, v. 7, n. 2, p. 41–50, abr. 2014. Disponível em: <http://www.sersc.org/journals/IJUNESST/vol7_no2/4.pdf>. Acesso em: 5 nov. 2017.

MCCARTHY, Matthew et al. Composable DevOps: automated ontology based DevOps maturity analysis. *IEEE International Conference on Services Computing*, p. 600–607, jun. 2015. Disponível em: <http://ieeexplore.ieee.org/document/7207405/>. Acesso em: 5nov. 2017.

MICROSOFT. *DevOps Self Assessment*: moving you to the second decade of agile. Disponível em: <https://devopsassessment.azurewebsites.net/>. Acesso em: 5nov. 2017.

PARÉ, Guy et al. Synthesizing information systems knowledge: A typology of literature reviews. *Information & Management*,v. 52, n. 2, p. 183–199, mar. 2015. Disponível em:

<http://www.sciencedirect.com/science/article/pii/S0378720614001116>. Acesso em: 5 nov. 2017.

PENNERS, Ralf; DYCK, Andrej. Release engineering vs. DevOps: an approach to define both terms. *Full-scale Software Engineering*, fev. 2015. Disponível em: <https://www2.swc.rwth-aachen.de/docs/teaching/seminar2015/FsSE2015papers.pdf#page=53>. Acesso em: 5 nov. 2017.

SHARMA, Sanjeev; COYNE, Bernie. *DevOps for dummies*. 2. ed. New Jersey: John Wiley & Sons, 2015.

SKELTON, Matthew. Joined-Up Thinking. *Oxford Academic ITNow*, v. 58, n. 1, p. 40–41, fev. 2016. Disponível em: <https://academic.oup.com/itnow/article-abstract/58/1/40/2392008/Joined-Up-Thinking>. Acesso em: 5nov. 2017.

SPAFFORD, George; HAIGHT, Cameron. Apply Gartner research for a DevOps perspective when implementing a bimodal strategy. *Gartner*, out. 2014. Disponível em: <https://www.gartner.com/doc/2893418/apply-gartner-research-devops-perspective>. Acesso em: 5 nov. 2017

SPINELLIS, Diomidis. Don't Install Software by Hand. *IEEE Software*, v. 29, n. 4, p. 86–87, ago. 2012. Disponível em: <http://ieeexplore.ieee.org/document/6265084>. Acesso em: 5 nov. 2017.

VIRMANI, Manish. Understanding DevOps & bridging the gap from continuous integration to continuous delivery. In:INTERNATIONAL CONFERENCE ON INNOVATIVE COMPUTING TECHNOLOGY, 5., p. 78–82, maio 2015. Disponível em: <http://ieeexplore.ieee.org/abstract/document/7173368/>. Acesso em: 5nov. 2017.

XEBIALABS. *DevOps Maturity Quiz*. Disponível em: <https://xebialabs.com/devops-maturity-quiz>. Acesso em: 5 nov. 2017.