DESENVOLVIMENTO E OPERAÇÕES UTILIZANDO ANSIBLE COMO PRINCIPAL FERRAMENTA DE IMPLANTAÇÃO: ESTUDO DE CASO NO TRIBUNAL DE JUSTIÇA DO DISTRITO FEDERAL (TJDFT)

Renato José da Silva Camões,
Jessé Alves da Silva

Resumo

O *DevOps* é uma resposta a uma desconexão entre as atividades de uma equipe de desenvolvimento e operacional na maioria das empresas. Essa desconexão muitas vezes se manifesta como conflito e a incompatibilidade entre ferramentas de desenvolvimento e operações. De acordo com os princípios do *DevOps*, a infraestrutura é código e deve ser desenvolvida e gerida. Processos automatizados se tornam essenciais, pois permitem executar tarefas mais rapidamente e diminui a possibilidade de erros humanos. O conceito de automação pode ser definido como a técnica de tornar um processo ou sistema automático e refere-se tanto a serviços executados como a produtos fabricados automaticamente e às tarefas de intercâmbio de informações. Desta forma, o objetivo do presente trabalho é mostrar a eficácia da utilização do *DevOps* através de ferramentas de automação de infraestrutura, em um estudo de caso feito no grupo de tecnologia do Tribunal de Justiça do Distrito Federal (TJDFT). Pretende-se demonstrar a mudança de cultura do setor com a possibilidade de automação dos processos, a eliminação de interrupções, evidenciando consistência, reprodutibilidade, continuidade, auto-teste e versionamento.

Palavras-chaves: DevOps. Infraestrutura. Tecnologia. Desenvolvimento.

Abstract

DevOps is a response to a disconnect between the activities of a development and operational team in most companies. This disconnect often manifests as conflict and the incompatibility between development tools and operations. According to the DevOps principles, the infrastructure is code and must be developed and managed. Automated processes become essential as they allow you to perform tasks faster and reduce the possibility of human error. The concept of automation can be defined as the technique of making an automatic process or system and refers to both executed services and automatically manufactured products and information exchange tasks. Thus, the objective of this work is to show the effectiveness of using DevOps through infrastructure automation tools, in a case study done in the technology group of the Federal District Court of Justice (TJDFT). It is intended to demonstrate the change of culture of the sector with the possibility of automation of processes, elimination of interruptions, evidencing consistency, reproducibility, continuity, self-test and versioning.

Keywords: *Devops. Infrastructure. Technology. Development.*

Introdução

É notório o desafio das organizações de T.I em lidar com um mercado cada vez mais competitivo e com mudanças rotineiras. Durante anos se desenvolvia *softwares* de formas prescritivas, práticas de desenvolvimento citadas por Roger Pressman (2010) como sequencial ou linear dos modelos cascata, incremental, evolucionários e espirais. Esses Modelos apresentam prescrições de um conjunto de atividades, tarefas, produtos e mecanismos de qualidade.

Portanto, desenvolver as ideias a partir de requisitos em tempo adequado pode influenciar o nível de concorrência das organizações de T.I. Alguns autores que participam do Manifesto Ágil, 2001, como Martin Fowler, Kent Beck, Paul Duvall, dentre outros, estudam práticas de desenvolvimento de *software* desde 1980 com o intuito de fornecer alternativas melhores que as tradicionais, como as dirigidas por documentação de modo rígido (VERHOEF, 2007).

Trabalhar em conjunto iguala-se a trabalhar em equipe. Nos últimos anos muito se tem ouvido sobre a tendência de uso *DevOps* como um método de desenvolvimento de *software* que enfatiza a comunicação, colaboração e integração entre os desenvolvedores de *software* e os profissionais de infraestrutura de TI. O método de desenvolvimento permiti aumentar o fluxo de trabalho completado,(maior frequência de *deploys*), ao mesmo tempo que aumenta a estabilidade e robustez do ambiente de produção (SATO, 2017).

Desta forma, o objetivo do presente trabalho é mostrar a eficácia da utilização Infraestrutura como código, a partir do *DevOps* através de ferramentas de automação das atividades manuais da equipe de infraestrutura em um estudo de caso feito em um grupo de tecnologia no âmbito do Tribunal de Justiça do Distrito Federal (TJDFT). Mais precisamente, a seção de Serviço de Suporte a sistemas Operacionais e Soluções de Armazenamento (SERSOP).

Na seção SERSOP que são aplicados os conceitos de *DevOps* para integração entre equipes de desenvolvimento e operações, apresentando os procedimentos realizados para automatizar a preparação de ambientes de produção através de programação de *scripts* e utilização de servidores virtuais na nuvem, como forma de padronizar e manter continuidade dos serviços, resultando em agilidade e eficiência da equipe no geral.

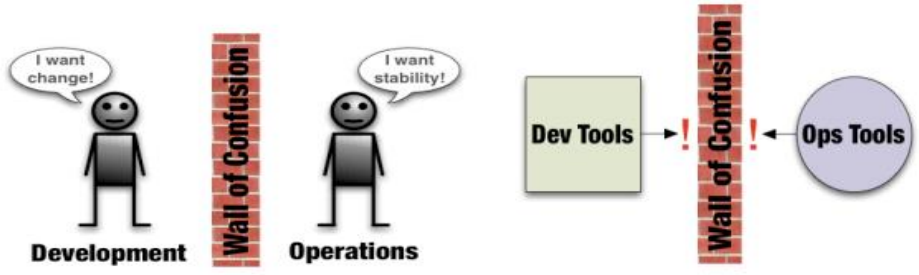
Devops

O termo *DevOps* é resultado da junção das palavras Desenvolvimento e Operações, que do inglês "*Software Development*" e "*Information Tecnology Operantions*", sendo uma boa imagem mental do conceito mais amplo quando se traz Dev e Ops juntos, obtendo *DevOps*. Houve outros termos para essa ideia, como *Agile Operations, Agile Infrastructure* (surgindo de um artigo publicado por Patrick Debois intitulado "*Agile and Operations Infrastrucuture: How Infra-gile Are You*?" (DEBOIS, 2008).

Porém, de acordo com Damon Edwards (2010), da página Dev2ops.org, durante a primeira conferência chamada "*DevOps Days*", na Bélgica 2009, foi popularizado o termo atual *DevOps* por Patrick Debois, propondo uma nova abordagem de trabalho que busca automatizar ao máximo possível processos e o aumento da colaboração entre as equipes.

O *DevOps* é uma resposta a uma desconexão entre as atividades de uma equipe de desenvolvimento e operacional na maioria das empresas. Essa desconexão, chamada por Lee Thompson (2010) de "*Wall Of Confusion*" (Muro da Confusão), ver figura 1, muitas vezes se manifesta como conflito e a incompatibilidade entre ferramentas de desenvolvimento e operações, já que as implementações serão diferentes em cada grupo, mas com algumas exceções notáveis, como ferramentas rastreadores de bugs.

Figura 1: *What is DevOps? "Wall of Confusion*" (Muro da confusão)



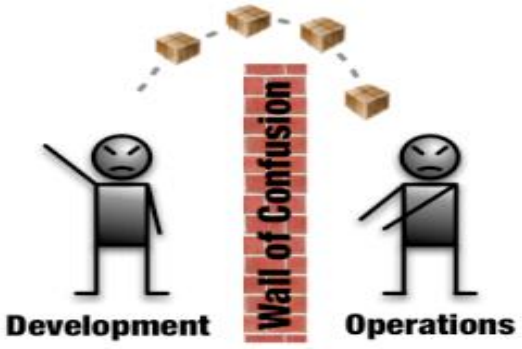
Fonte: Damon Edwards (2010) dev2ops.org

De acordo com a imagem acima, pessoas centradas no desenvolvimento tendem a uma mentalidade de mudanças como produtos resultantes do foram pagos para produzir. O negócio depende deles para responder às necessidades em

mudança e são frequentemente incentivados a criar o máximo de mudanças possíveis. Já os colaboradores e servidores de operações demonstram resistência a mudanças, tendendo a mentalidade resistente, considerando mudanças como aspecto negativo. O negócio depende deles para fornecer os serviços que fazem o dinheiro como lucro e são motivadas a resistir à mudança, pois prejudica a estabilidade e a confiabilidade (EDWARDS, 2010).

O “Muro da Confusão” (*Wall of Confusion*) é mais perceptível quando da ocorrência de mudanças nos aplicativos ou uma versão de *software* ser enviada da equipe de desenvolvimento para a de operação (figura 1.1), esse lançamento é chamado de *Deploy* ou em português “Implantação”.

Figura 1.1: *What is DevOps? Deploy* (implantações)



Fonte: Damon Edwards (2010) dev2ops.org.

A equipe de Operações (*Operations*) recebe os artefatos de *release* e começa a preparar sua implementação. Retiram manualmente os *scripts* de implantação fornecidos pelos desenvolvedores ou criam seus próprios *scripts*. Na melhor das hipóteses eles estão duplicando o trabalho que já foi feito em ambientes anteriores, na pior das hipóteses eles estão prestes a introduzir ou descobrir novos *bugs* (EDWARDS, 2010).

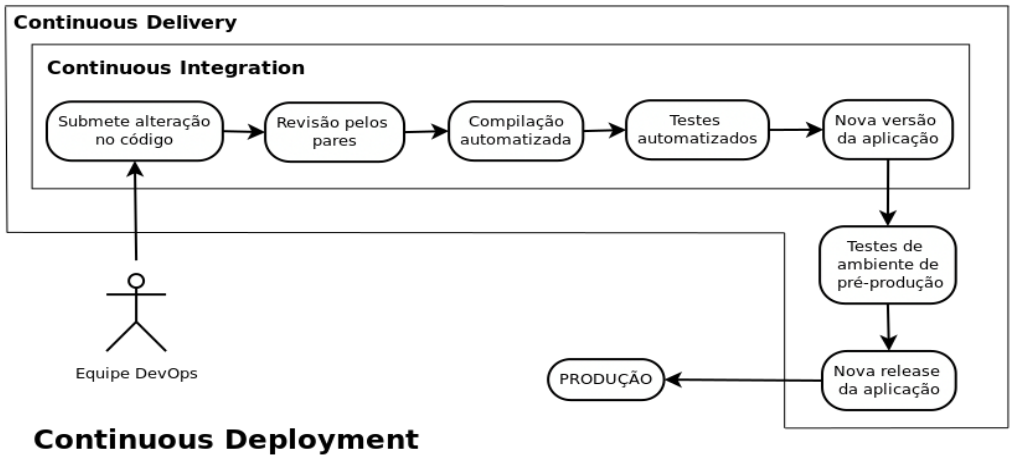
Em seguida, a equipe de operações avança para o processo de implantação, que neste ponto está essencialmente sendo executado pela primeira vez devido às diferenças de *script,* configuração, processo e ambiente entre Desenvolvimento e Operações. É claro que, em algum momento, ocorre um problema e os desenvolvedores são chamados para ajudar na solução, mas impasses prevalecem, por exemplo: a equipe de Operações (*Operations)* afirma que o *Development (*Equipe Desenvolvimento) deu a eles artefatos defeituosos. Os desenvolvedores

respondem apontando que funcionou muito bem em seus ambientes, então deve ser o caso da *Operations* ter feito algo errado. Os desenvolvedores encontram dificuldades para diagnosticar o problema, porque a configuração, os locais dos arquivos e o procedimento utilizado para entrar nesse estado (Implantação) são diferentes do que esperam.

Neste cenário, percebe-se que as equipes evoluem em ritmos diferentes e a falta de comunicação e contribuição entre elas impactam na produção e negócio. Conforme o autor Danilo Sato (2017), além da mudança cultural, o *DevOps* tem um enfoque nas práticas de automação de atividades rotineiras a fim de reduzir erros humanos e alcançar a entrega contínua dos serviços essenciais para o negócio.

Marcel Faria (2017), buscando uma visão integrada em *DevOps,* explica quais são os três *pipelines* principais de processos automatizados ou semi-automatizados. Primeiramente o uso de *Continuous Integration* ou Integração contínua, onde as alterações feitas pelos diferentes elementos do grupo são validadas, testadas e integradas ao repositório compartilhado do código fonte da aplicação. *Continuous Delivery* (Entrega Contínua) estendido até geração nova *release* de *software*. Por último, o *pipeline Continuous Deployment* (Implantação Contínua) onde a nova *release* é posta em produção, como mostra a figura 1.2.

Figura 1.2: Pipelines utilizados no DevOps



Fonte: Marcel Faria (2017).

Continuous Integration **(Integração contínua)** : é uma prática focada no desenvolvimento, em um processo de integrar continuamente o trabalho

desenvolvido em um repositório com o resto da equipe de desenvolvimento. Os testes do trabalho integrado a cada checkin no repositório, permitindo assim detectar e localizar erros rapidamente, como destacam Kent Beck e Cynthia Andreas (2000). Entre as atividades presentes nela estão o controle de versão com repositório único, build automatizado, commits diários no repositório principal, conjunto de testes automatizados e abrangentes, entre outros;

Continuous Delivery **(Entrega contínua)**: entrega contínua é uma prática que foca na entrega de software da equipe de desenvolvedores para o ambiente de produção em um processo confiável, previsível, visível e o mais automatizado, com riscos mitigados e bem entendidos, além de incluir as atividades da fase anterior, inclui-se as atividades de Teste de ambientes de pré-produção e lançamento de novas releases de aplicações. Essa linha de pensamento que tenta diminuir o tempo entre a criação de uma ideia e sua implementação em produção (HUMBLE; FARLEY, 2014).

Continious Deployment (implementação contínua): Por último, o pipeline de implementação contínua foca em colocar em produção a nova release de software gerada na fase anterior (FARIA, 2017).

A fim de facilitar a implementação do pipeline de Deploy Continuo, pode-se utilizar o padrão de projeto Infraestrutura como código, pois permite a execução automática, isolada, concorrente de testes de integração em ambientes de produção. Diversas ferramentas têm evoluído para tentar padronizar o gerenciamento automatizado de infraestrutura. Tais ferramentas permitem tratar infraestrutura da mesma forma que trata-se código, é a prática conhecida como IaC (SATO,2017).

IAC- Infraestrutura como código

Para a empresa AMAZON, a infraestrutura como código é uma prática em que a infraestrutura é provisionada e gerenciada utilizando técnicas de desenvolvimento de código e *software*. Adonai Medrado (2015) explica que funciona como controle de versão e integração contínua.

De acordo com os princípios do *DevOps*, Ernest Mueller (2010), a infraestrutura é código e deve ser desenvolvida e gerida. Processos automatizados se tornam essenciais, pois permite executar tarefas mais rapidamente e diminui a possibilidade de erros humanos.

Conforme Danilo Sato (2017), alguns administradores de sistema escrevem seus próprios scripts para automatizar as tarefas mais comuns, essas tarefas altera o estado do servidor, podendo causar problemas se a mudança for mal executada. Se cada administrador escreve sua própria versão dos scripts, fica difícil reutilizá-los em outras situações conforme cresce o número de servidores a serem administrados.

Portanto, a IaC se torna um pré-requisito para permitir a aplicação das práticas *DevOps* no contexto de infraestrutura de redes, como destaca Marcel Faria (2017). De acordo com Danilo Sato (2017) um processo automatizado é mais confiável e pode ser auditorado com mais facilidade.

O autor Kief Morris (2016) lista alguns princípios para por em prática o IaC em seu livro, Insfraestructure as Code. Por conta da extensão do assunto, será exposto aqui, de forma resumida, os mais relevantes para o entendimento:

- **Consistência:** Dados dois elementos de infraestrutura que fornecem um serviço semelhante - por exemplo, dois servidores de aplicativos em um cluster - os servidores devem ser quase idênticos. O *software* e a configuração do sistema devem ser exatamente os mesmos.
- **Reprodutibilidade:** Deve ser possível reconstruir, sem esforço e de forma confiável, qualquer elemento de uma infraestrutura. Decisões sobre qual software e versões instalar em um servidor, como escolher um nome de host e assim por diante.
- **Sistemas de auto-teste:** O teste automatizado eficaz é uma das práticas mais importantes da infraestrutura. As configurações são validadas e testadas antes de serem aplicadas, pelo uso de ferramentas de *Continuous Integration*.
- **Sistemas de auto-documentação:** No lugar de manter as configurações em documentos, o próprio código serve como documentação (templates de configurações, scripts, configurações aplicadas).
- **Versionamento:** O controle de versão da configuração de infraestrutura é a base da infraestrutura de código. Torna-se possível acompanhar problemas devido alterações que não saíram conforme o esperado.

Automatização de processos de infraestrutura

O conceito de automação é definido por J.T. Black (1998), como a técnica de tornar um processo ou sistema automático e refere-se tanto a serviços executados como a produtos fabricados automaticamente e às tarefas de intercâmbio de informações.

Para Paul Duvall (2012), automação da infraestrutura é o processo de criar scripts de ambientes — da instalação de um sistema operacional até a instalação e configuração de serviços em instâncias, a configuração de como as instâncias e softwares comunicam-se entre si, e muito mais. Criando scripts para ambientes, sendo possível aplicar a mesma configuração a um único nó ou a milhares.

Segundo a INSTRUCT, uma das primeiras empresas brasileiras especializada em Infraestrutura Ágil com foco em automação e gerência de configurações. A Infraestrutura Ágil tem o foco em transformar a forma de como o time de operação lida com a infraestrutura de sua organização. Sendo um ponto de partida para ganhar maturidade e caminhar em direção a cultura DevOps (INSTRUCT,2016).

A empresa lista os princípios para essa automação:

- **Gerencia de Configuração**: Nesta atividade são utilizadas ferramentas de gerência de configurações para manter a padronização e o compliance, conforme Brasil Endeavor (2015) significa agir em sintonia com as regras e normas de políticas, em sistemas e serviços, a partir do controle automatizado de estados do sistema.
- **Gerencia de Estados:** é a principal característica de ferramentas que gerenciam configurações: com elas, é possível definir os estados desejados para o sistema.
- **Orquestração:** Consiste em executar ações de forma paralela ou não, em tempo real, em um conjunto de nodes ou usando ferramentas onde se especifíca uma sequência de comandos ou etapas para cumprir um objetivo.
- **Provisionamento:** Provisionar significa ter uma camada de inteligência entre a necessidade e os recursos disponíveis na infraestrutura. A criação de novos ambientes deve ser um processo rápido e fácil. Levando no máximo alguns minutos para ser realizado e o usuário/consumidor deve ter acesso ao recurso de maneira direta.

Estudo de caso

A Infraestrutura de Tecnologia da Informação do TJDFT é a área que atua com o Data Center, sistemas operacionais, sistemas de bancos de dados, servidores de aplicação, *storages*, virtualização, segurança de dados, redes locais e metropolitanas, sistemas de telefonia e videoconferência, monitoramento.

A SERSOP compõe a SETEC e tem como atribuições, relacionadas na Resolução n. 02, de 12/12/2016 do TJDFT, monitorar e manter operacional as plataformas de servidores computacionais; virtualização; propor modernização das soluções utilizadas.

CENÁRIO ANTERIOR
A equipe SERSOP no cenário antigo, utilizando ITIL e *Scrum* para prover serviços, recebia todas as requisições de configurações e provisionamento em formulário, e manualmente criava e configurava cada máquina virtual ou servidor através de *scripts,* criadas de acordo com os requisitos solicitados por desenvolvedores de aplicação. Dessa forma existia risco de instruções serem esquecidas, devido a não padronização das configurações e falhas ocorrerem em servidores criados.
PROBLEMA
O setor recebia a solicitação da equipe de desenvolvimento, e então criavam-se máquinas virtuais através de um formulário da aplicação VMWare (gerenciador VMs) com base nas especificações dos desenvolvedores. Todo, esse processo dependendo da demanda do dia para o setor, resultava em atrasos na entrega de infraestrutura, com tempo de criação e configuração de até 3 horas para cada máquina virtual. Fornecê-las em tempo adequado aos desenvolvedores de aplicação, para uma migração prevista do sistema PJE (Processo judicial eletrônico) 1.7 para 2.0 se tornava inviável ao prazo de implantação, no segundo semestre de 2017.
SOLUÇÃO PROPOSTA
Implantar a cultura de comunicação/colaboração e automação no setor, integrando as equipes de desenvolvedores de aplicação e operações no trabalho diário, com

reuniões apontando a retrospectiva do dia, gerando empatia e criação de serviços compartilhados como ambientes de produção automatizados, automatizando atividades repetitivas de criação de máquinas virtuais e versionamento do código, mantendo uma padronização.

Instalação da ferramenta *Ansible*, no sistema *Linux* de distribuição integrando-a com as já existentes como VMWAre. A estrutura *Ansible* conta com inventários coordenando os *host's* a serem gerenciados pela ferramenta *Ansible*; Módulos para controle de serviços e outros recursos host's; Tarefas, também denominadas *Task's*, de execução nos *host's*; Bem como, *Playbook's* em YAML de execução nos *host's.*

Os módulos utilizados por *Ansible*, viabilizaram junto ao inventário a execução da maioria das atividades e tarefas, como instalação de softwares e cópias de arquivos. Para tarefas ou conjunto mais avançadas os *Playbook's* eram colocados execução, sendo que, para cada tarefa, um grupo-alvo era especificado, e as tarefas definidas nos *Playbook's* eram executados em todos os *host's.*

Resultados

A partir da problemática acima, os quadros abaixo mostram como foi possível implantar as soluções propostas para o setor.

QUADRO 1: Características de Soluções.

Característica do *DEVOPS*	**Característica desejada para SERSOP**	**Ferramenta Utilizada na Implementação**
Pipelines	Integração contínua; Entrega contínua; Implantação contínua.	***Jenkins*** (servidor de automação de fluxo)
Ambiente Técnico	Automatizar todo o processo de implantação, desde a criação, testes e produção, obtendo um *feedback* rápido a todos no fluxo de valor, colaborando com correções imediatas ao problema.	
Resultados Iniciais	Facilitou e acelerou o processo de implantação, fluxo de valor do setor e para equipe implantação; ganhou qualidade, agilidade. Comunicação entres equipes com o *feedback* rápido durante o pipeline. ; Ganhou de	

	tempo nas entregas; Menor número de erros.

Fonte: O Autor.

O quadro 2 traz características relacionadas a implantação de *DevOps* utilizando *Ansible* como ferramenta viabilizadora dessa implantação.

QUADRO 2: Caracteristicas de Soluções.

Característica do *DevOps*	**Característica desejada para SERSOP**	**Ferramenta Utilizada na Implementação**
IaC – Infraestrutura como código	Tratar a infraestrutura como código, provisionar e gerenciar com técnicas de desenvolvimento de código e *software.*	***Ansible* (Gerencia Configuração) - *Vargran*t – *Docker* (Montar ambiente)**
Ambiente Técnico	Código e configurações dentro do controle de versões. Criação automatizada e sob demanda de ambientes, evitando criação manual. Infraestrutura imutável, recriar ao invés de alterar ambiente feito, pois tudo é mais rápido.	
Resultados Iniciais	Garantiu automação dos pipelines, a criação de ambientes e configuração.	

Fonte: O Autor.

O quadro 3 traz os resultados alcançados através da implantação de *DevOps* utilizando a ferramenta *Ansible* no ambiente do TJDFT.

QUADRO 3: Resultados da implementação de DEVOPS através do Ansible

Característica do *DevOps*	**Característica desejada para SERSOP**	**Ferramenta Utilizada na Implementação**
Automatização de processos de Infraestrutura	criar *scripts* de instalação de sistemas operacionais e instalação e configuração de serviços	***Ansible, VMWare***
Ambiente Técnico	Foram criadas *playbook's* no ansible para automatizar a criação e configuração de máquinas e servidores virtuais na rede do tribunal. Uma *playbook* basicamente	

	é uma sequência de tarefas criada a partir dos módulos do ansible e de acordo com o problema que se pretende automatizar, um tipo de manual de instruções. Para colocar uma VM em atividade, a SERSOP realizava requisição dos Desenvolvedores e transformavam-nas em código, scripts e disparavam no *Ansible*, determinando a quantidade de máquinas ou tamanhos de storages necessários, e o *Ansible* cuidava de rodar as etapas de configurações codificadas:
Resultados Iniciais	Com essa automatização no provisionamento as entregas se tornaram contínuas, reduzindo o tempo de entrega de 3h para 15min, tanto em máquinas virtuais quanto em disponibilização de *storage* para o PJE 2.0.

Ansible como principal ferramenta da implantação no SERSOP

Ansible trata-se de uma ferramenta de automação de código aberto, inicialmente desenvolvida por Michael DeHaan e atualmente mantida pela comunidade e pela *Red Hat*. O *Ansible* é de fácil aprendizagem e utiliza por padrão SSH para se comunicar com os clientes (*nodes*) (ALMEIDA, 2017).

Através da criação de *playbook's* no *Ansible* a solicitação de máquinas em documentos oficiais passaram para transformação em código, oferecendo provisionamento, gerência de configuração e reprodutibilidade, como descreve o quadro 2.

QUADRO 4: Possibilidades com Ansible

	Possibilidades ofertadas	**Descrições**
ANSIBLE	Provisionamento	Remoto via SSH em diversos servidores, dinamismo e eficiência.
	Gerência de configuração	Possibilidade de observar o progresso e promover mudanças, ou fixar aspectos positivos do processo, criando

		modelos de configuração.
	Reprodutibilidade	Através de templates através de arquivos de configurações utilizando variáveis e determinadas, ofertando modelos a serem copiados.

Fonte: O Autor.

Provisionamento no *Ansible*

O provisionamento em *Ansible* demonstrou possibilitar comunicação entre servidores via SSH, coordenando ***host's*** através de ***playbook's***, condicionando que o *Ansible* configure e altere o processo de produção de códigos, permitindo reprodutibilidade a partir de moldes ou templates dos códigos criados, permitindo a continuidade dos serviços. O *Ansible* possui módulos para containers (*Docker)*, Virtualização (*VMware,* AWS, *OpenStack, Azure, Ovirt*) e podem facilmente se integrar com outras tarefas.

Gerencia de Configuração no *Ansible*

Código, ciclo de vida e mudanças poderam ser criados através de módulos, inventário e ***playbook's*** no *Ansible*, assim como a gerencia dos estados desejados e idempotência nativamente nas tarefas que serão executadas. Tudo de forma muito simples e robusta.

Orquestração no *Ansible*

Nesse quesito, o *Ansible* demonstrou se integrar com quase todas as áreas da infraestrutura, desde o provisionamento de VMs até liberação de regras no firewall. Também focando nas áreas onde outras ferramentas que deixavam lacunas, tais como, integrações contínuas *(Continuous Integration*) para aplicações multi-camadas em toda a infraestrutura.

Resultados gerais

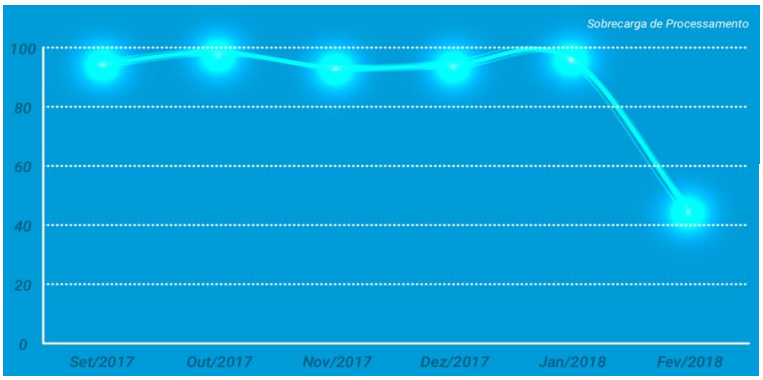
Seguindo o exposto no quadro 1, os *pipelines* de Integração, Entrega e Implantação continua facilitaram a automação da Infraestrutura, desfazendo as confusões, comparadas pela literatura como um muro que impede a comunicação eficaz entre setores possibilitando que as equipes envolvidas no processo obtivessem visibilidade do progresso, identificar as mudanças necessárias e quais aspectos puderam ser mantidos.

A Infraestrutura como código configura requisito para implantação da cultura *DevOps* no sentido de poder ser tratada como software, facilitando processos em rede integradas. A partir disso, em uso no TJDFT, o *Ansible j*unto com a IaC permitiu automatização das máquinas virtuais, agilizando e diminuindo mão de obra no setor, além fornecer ambientes para testes aos desenvolvedores, próximo ao de produção. Tal funcionalidade foi possível devido a criação de *playbook's* transformando documentos oficiais em códigos.

Dessa forma com a mudança de cultura do setor para *DevOps* com automação, possibilitou processo sem interrupções, evidenciando consistência, reprodutibilidade, continuidade, auto-teste e versionamento.

A implantação ocorreu no segundo semestre de 2017 e graças as mudanças feitas foi possível ampliar e otimizar a infraestrutura do tribunal, tornando-a escalável, com maior disponibilidade e capacidade de processamento, aumentando a performance das transações do sistema, como se vê no gráfico abaixo:

Figura 2: Sobrecarga Processamento



Sobrecarga de Processamento

MÊS	MÁXIMA
Set/2017	93,84
Out/2017	97,88
Nov/2017	92,65
Dez/2017	93,63
Jan/2018	95,71
Fev/2018	44,01

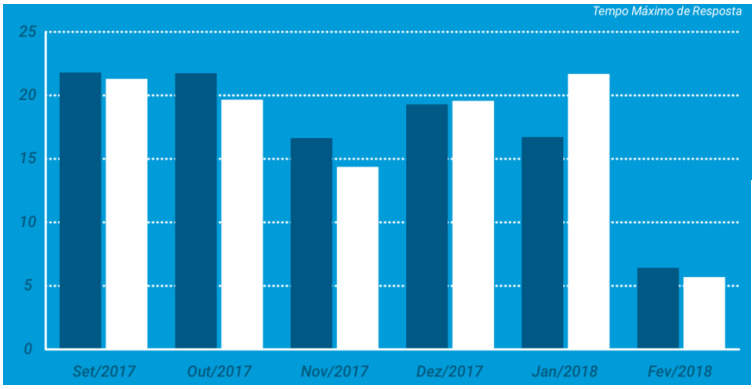
Fonte: relatório de gestão SETEC biênio 2016/2018 PJE resultados

De acordo com gráfico é perceptível que o nível de sobrecarga no sistema do início da implantação até o ano de 2018 foi reduzido, do pico em Out/2017 de

93,88% para 44,01% em Fev/2018, isso devido a escalabilidade e a capacidade de provisionamento rápido da infraestrutura com as máquinas virtuais e *storage* ao sistema.

Outro gráfico que mostra os resultados da infraestrutura automatizada é o tempo de resposta do sistema as requisições de acesso aos servidores.

Figura 2.1: Tempo máximo resposta PJE



Tempo Máximo de Resposta

MÊS	INTRANET	INTERNET
Set/2017	21,8	21,3
Out/2017	21,74	19,66
Nov/2017	16,64	14,37
Dez/2017	19,29	19,57
Jan/2018	16,72	21,69
Fev/2018	6,42	5,69

Fonte: relatório de gestão SETEC biênio 2016/2018 PJE resultados

O tempo de resposta no ano de 2018 caiu de 21,3 segundos para 5,69. Sistemas que possuem infraestrutura alicerçadas na virtualização como PJe, possibilitando a configuração de múltiplos servidores virtuais hospedados em um mesmo servidor físico, proporcionando crescimento automatizado dos recursos.

Segundo relatório interno de gestão da SETEC, o grau de efetividade no atendimento a demandas por infraestrutura de T.I do tribunal aumentou significativamente entre os anos 2016 e 2017.

REFERÊNCIAS

About Devopsdays. 2009. Disponível em: <https://www.devopsdays.org/about/>. Acesso em: 2018.

ALMEIDA, Tiago. **Automação e provisionamento ágil com Ansible**. 2017.Disponível em: <https://imasters.com.br/desenvolvimento/devops/automacao-e-provisionamento-agil-com-ansible/?trace=1519021197&source=single>. Acesso em: abril 2018.

AMAZON. O que é DevOps - **What is DevOps**. 2018 <https://aws.amazon.com/pt/devops/what-is-devops/>. Acesso em : 2018

BECK, Kent; ANDREAS, Cynthia. **Extreme Programming Explained.** Boston: Addison-Wesley, 2000.

BLACK, J.T. **O Projeto da Fábrica com Futuro**. Bookman. Porto Alegre. 1998

EDWARDS, Damon,. **What is Devops –O que é Devops Tradução Adaptada**. 2010 . Disponível: < http://dev2ops.org/2010/02/what-is-devops/>. Acesso em: 2018.

DEBOIS, P. **Agile infrastructure and operations: how infra-gile are you?.** 2008. Disponível : <http://www.jedi.be/presentations/IEEE-Agile-Infrastructure.pdf>.

DUVALL, Paul. **Automação de Infraestrutura -Trate a infraestrutura como código com CHEF e PUPPET**. 2012. Disponível: <https://www.ibm.com/developerworks/br/library/a-devops2/index.html>. Acessado em: 2018.

Ebook Gratuito, **Guia Definitivo Infraestrutura Ágil**. 2016. INSTRUCTION. [S.l.:s.n].2017.Disponível em: <http://instruct.com.br/ebook-infraestrutura-agil/>.Acesso em: Abril 2018.

ENDEAVOR, Brasil. **Prevenindo com o Compliance para não remediar com o caixa**. 2015. Disponível em: <https://endeavor.org.br/compliance/> . Acesso em: abril 2018.

FARIA, Marcel R.. **DevOps para Infraestrutura de Rede**.2017. Disponível em: <https://www.rnp.br/sites/default/files/ger-propspeccao-devops_v8.0.pdf>. Acesso em: 2018.

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

HUMBLE, Jez; FARLEY, David. **Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation**. Boston: Addison-Wesley Professional, 2010.

MEDRADO, Adonai. **O que é DevOps? Colaboração como caminho para entregar valor ao negócio**. Rio de Janeiro: AVM, 2015.

MORRIS, Kief. **Infrastructure as Code - Managing Servers in the Cloud**. First Edition. O´Reilly Media .United States of America. 2016.

MUELLER, Ernest. **A DevOps Manifesto**. 2010. Disponível em: < https://theagileadmin.com/2010/10/15/a-devops-manifesto/>. Acesso em: abril 2018.

PRESSMAN, Roger S. Engenharia de Software. Mc Graw Hill, 6 ed, Porto Alegre, 2010.

SATO, Danilo. **DevOps na prática: entrega de software confiável e automatizada**. São Paulo: Ed. Caso do Código, 2017.

THOMPSON, Lee. **ex-chefe de tecnologia da E * TRADE Financial**. 2009 Disponível em: <http://dev2ops.org/2009/09/qa-lee-thompson-former-chief-technologist-of-etrade-financial/>. Acesso em: 2018.

VERHOEF, Peter C. **Quantifying the effects of IT-governance rules. Science of Computer Programming.** 2007. Disponível em: <https://ac.elscdn.com/S0167642307000780/1-s2.0-S0167642307000780-main.pdf?_tid=676b5762-a39d-4c9a-902c62f937db1dda&acdnat=1532113551_0ed88de8dd15d5eb04568771614ae20e>. Acesso em: 2018.