

Questions

Responses

30

Um panorama sobre experiências e práticas de profissionais em ambientes DevOps.

Me chamo Kim Agliardi e sou aluno do curso de Ciência da Computação na Universidade do Vale do Rio dos Sinos (Unisinos), e gostaria de convidá-lo a participar da pesquisa um panorama sobre experiências e práticas de profissionais em ambientes DevOps. O presente questionário tem o objetivo analisar as características dos profissionais (com enfoque técnico) que estão trabalhando inseridos em companhias que praticam a cultura DevOps, seus conhecimentos, habilidades, experiências, e prós e contras na adoção desta cultura de colaboração. Realizaremos algumas perguntas no início para entender melhor quais são as suas experiências no mercado de TI e após isso, sobre suas experiências no "lado DevOps da força" :) As informações de identificação coletadas nesta pesquisa não objetivam identificar as empresas ou profissionais pesquisados. Para esclarecimento de dúvidas ou sugestões, estou disponível através do e-mail: kagliardi@edu.unisinos.br Tempo estimado para preenchê-lo: 15 minutos

1.Qual o seu endereço de e-mail?

2.Em que cidade/estado você trabalha?

3Informe a quantidade de colaboradores da empresa:

☐ Até 19 colaboradores

- ☐ De 20 a 99 colaboradores
- ☐ De 100 a 499 colaboradores
- ☐ Mais de 500 colaboradores

4. Se você fosse resumir suas experiências em TI, qual seria a área predominante?

- ☐ Operações de TI (Banco de dados, Middleware, Infraestrutura, Serviços, etc..)
- ☐ Segurança da informação
- ☐ Gestão administrativa
- ☐ Desenvolvimento de software (Front-End, Back-End, Testes, Mobile, etc..)
- ☐ Análise de negócios



5. Quanto tempo de experiência você tem em tecnologia da informação?

- ☐ Menos de um ano
- ☐ 1 a 2 anos
- ☐ 3 a 5 anos
- ☐ 6 a 10 anos
- ☐ Acima de 10 anos

6. Qual é o seu cargo atual?

7.Quanto tempo de experiência você tem utilizando ou gerenciando ferramentas de automação voltadas para ambientes que praticam DevOps?

- ☐ Não possuo experiência, mas estou pesquisando sobre DevOps.
- ☐ 1 a 2 anos
- ☐ 2 a 3 anos
- ☐ 3 a 5 anos
- ☐ Acima de 5 anos

8.Qual o ramo de atuação da companhia em que você trabalha?

- ☐ Tecnologia/Software
- ☐ Atividades de serviços financeiros
- ☐ Fabricação e Materiais
- ☐ Varejo/Comércio
- ☐ Saúde
- ☐ Educação
- ☐ Organizações sem fim lucrativo
- ☐ Órgão governamental
- ☐ Telecomunicações e serviços
- ☐ Transportes e logística
- ☐ Agronegócio

9.Sua organização trabalha com times cross-funcionais (composto por indivíduos com diferentes competências e perfis que atuem em diferentes atividades (Ex: Desenvolvedores, infra, UX designers, testadores, analistas de negócio...), aonde todos trabalham juntos e colaboram para atingir um objetivo em comum?

- ☐ Sim
- ☐ Somente em alguns projetos
- ☐ Não

10.Como você descreveria o compartilhamento de conhecimento entre os times de TI em sua companhia?

- ☐ As informações são abertas e estão disponíveis para todos, provendo informações técnicas, objetivos, planos e resultados, em formatos dinâmicos, como wikis e salas de bate-papo.
- ☐ As informações são abertas e estão disponíveis apenas para equipes cross-funcionais em formatos dinâmicos, como wikis e salas de bate-papo.
- ☐ As informações são abertas e estão disponíveis apenas para equipes cross-funcionais em formatos estáticos, como páginas da Web ou documentos.
- ☐ As informações são compartilhadas entre os times somente quando requisitado.
- ☐ O compartilhamento de conhecimento é baixo mesmo entre integrantes do mesmo time.

11.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.
- ☐ As ferramentas entre Desenvolvimento (Dev), Qualidade (QA) e Operações (Ops) são integradas, mas não há um único fluxo entre elas.
- ☐ As ferramentas são parcialmente compartilhadas entre Dev, QA e Ops.
- ☐ Há pouca integração as ferramentas e algum compartilhamento de status, via ferramentas de monitoramento.

- ☐ Dev e Ops utilizam um conjunto de ferramentas distintas e redundantes.

12. Como o seu time colabora, divide riscos, inova e aprende?

- ☐ Existe um alto nível de colaboração entre os times, riscos são compartilhados. Falhas levam ao aprendizado e à investigação para ajudar a equipe a melhorar. A inovação é incentivada e implementada.
- ☐ A colaboração, inovação e aprendizado entre os times são bem-vindos. Encorajamos a inovação.
- ☐ A colaboração, inovação e aprendizado é uma prática estimulada e compartilhada dentro do time, existe alguma comunicação entre os times, porém, quando erros são cometidos, há justiça para as pessoas responsáveis.
- ☐ A colaboração, inovação e aprendizado é uma prática estimulada e compartilhada dentro do time, com a possibilidade de riscos.
- ☐ A colaboração, inovação e aprendizado é uma iniciativa exclusivamente individual, iniciativas não são estimuladas pelo time.

13. Como é planejado, priorizado e agendado o trabalho?

- ☐ De forma contínua, por meio de um backlog orientado por hipóteses, com base em experiências mensuráveis e coleta de dados em produção.
- ☐ De forma consistente, em todos os setores da empresa.
- ☐ De maneira consistente dentro das equipes, mas as prioridades entre os times podem ser incompatíveis.
- ☐ Há uma tentativa de priorização dentro das equipes, mas ainda é feito de forma inconsistente e irregular.
- ☐ Não há um processo de planejamento/priorização consistente.

14. O quão automatizado é o processo de provisionamento de um ambiente?

- ☐ Todas as tarefas de infraestrutura são geridas por código fonte e ferramentas de provisionamento. Máquinas virtuais, banco de dados, servidores de aplicação, servidores web e até mesmo redes (no modelo SDN - Software Defined Network) são configuradas por meio de códigos fonte. A camada de hardware é "abstraída" e se torna software.

- ☐ Utilizamos ferramentas de provisionamento (ex: Docker, Ansible, Chef e Puppet) e os scripts de provisionamento são totalmente versionados e distribuídos de forma que a equipe de desenvolvimento (Dev) possa provisionar ambientes de maneira self-service. O provisionamento de ambientes de produção é controlado por máquinas e com a mínima interferência humana. Para isso, são estabelecidos ambientes de nuvens públicas e/ou privadas para facilitar a implementação de políticas de escalabilidade e alocação de recursos no modelo pay-as-you-go. Existe uma redução notável no trabalho manual das equipes de operações.
- ☐ O time de infraestrutura (Ops) utiliza ferramentas de provisionamento (ex: Docker, Ansible, Chef e Puppet) para tornar o trabalho de provisionamento, instalação e configuração totalmente automatizados. Os scripts de provisionamento começam a ser escritos e organizados em repositórios de código. A criação de ambientes é facilitada e o tempo de atendimento para requisições deste tipo é reduzido.
- ☐ O time de Ops utiliza ferramentas de virtualização, como VMware, VirtualBox e Hyper-V, para facilitar o trabalho de criação e configuração de máquinas. Algumas tarefas são operacionalizadas por meio de scripts, mas o trabalho manual é dominante e gera altos tempos de atendimento para requisições de criação de ambientes.
- ☐ Todo trabalho de preparação de ambientes é realizado diretamente em hardwares reais, e isso é repetido para cada nova requisição realizada para o time de Ops. Os tempos de atendimento são longos.

15. A equipe de desenvolvimento considera requisitos não funcionais no processo de desenvolvimento de uma aplicação?

- ☐ Sim, todos os requisitos não funcionais são levantados em conjunto entre as equipes de desenvolvimento e operação.
- ☐ Sim, todos os requisitos não funcionais são levantados, mas a equipe de operações não é envolvida no processo.
- ☐ Considera que esta é uma responsabilidade de ambas as equipes, mas não considera no processo de desenvolvimento.
- ☐ Considera que esta é uma responsabilidade apenas do time de operação, ao qual é delegado o levantamento destes requisitos durante o processo de desenvolvimento.
- ☐ O time de desenvolvimento não leva os requisitos não funcionais em consideração ao longo do processo de desenvolvimento.

16. O que a sua equipe considera como um bom nível de testes?

- ☐ Os testes são totalmente automatizados, as reversões de produção são raras e os defeitos são encontrados e corrigidos imediatamente.
- ☐ Métricas e tendências de qualidade são rastreadas. Requisitos não funcionais são definidos e medidos
- ☐ Automatizamos testes unitários e de aceitação. Testes são parte do processo de desenvolvimento. Os loops de feedback estão em vigor e a melhoria contínua é medida e gerenciada.
- ☐ Automatizamos gradualmente o teste de integração manual existente para um feedback mais rápido e testes de regressão mais abrangentes. Para testes precisos, o componente deve ser implantado e testado em um ambiente de produção com todas as dependências necessárias.
- ☐ Praticamos análise de código estático e testes unitários manualmente em um ou mais ambientes de testes separados das máquinas de desenvolvimento local.

17. Na sua organização, quando são realizados os testes de segurança?

- ☐ Testes de segurança são realizados a cada pull request e Code review, antes de serem "commitados".
- ☐ Testes de segurança são executados nos pipelines de entrega contínua e integração contínua.
- ☐ Antes de qualquer atualização no ambiente de produção.
- ☐ Antes do lançamento de uma grande release.
- ☐ Não temos uma prescrição clara sobre a aplicação de testes de segurança / Não realizamos testes de segurança.

18. Na sua organização, como é realizada a gestão de builds?

- ☐ A prática de integração contínua passa a ser realizada, o processo de build é disparado por alterações no código fonte. Em outros times, existem também políticas automatizadas que impedem o commit de código fonte que provoque quebra nos builds (padrão Gated Commit: <https://goo.gl/ArDPRH>).
- ☐ Os builds começam a ser executados várias vezes por dia e operam até mesmo em ambientes paralelos com objetivos distintos (ex. Ambiente de build para testes de performance e ambiente de Build para testes funcionais). A cobertura do código fonte é agora

expressiva e normalmente passa de 50% do código fonte e inclui testes funcionais e outros tipos de teste como performance, usabilidade ou segurança.

- Builds são executados em intervalos regulares em ambientes dedicados (ambientes de integração). Ferramentas como o Jenkins, GitLab, IBM Rational Team Concert, Microsoft VSTS(ou TFS) passam a ser utilizadas para compilar e testar os builds. Falhas nestas gerações geram defeitos automatizados para o time de desenvolvimento. Os testes são ampliados e passa a ser utilizada uma verificação automatizada de métricas de qualidade de código.

- A cultura de builds começa a ser praticada. Ferramentas como Make, Maven, Gradle, Rake, Nuget e MSBuild passam a ser utilizadas pelo time visando eliminar dependências de máquina durante o processo de compilação. Isso que o código seja compilado com facilidade em qualquer ambiente. Também são realizados smoke tests (testes de fumaça) para garantir uma mínima estabilidade do build.

- Não existe uma cultura de gestão de builds. A compilação é acoplada a IDEs como Eclipse ou Visual Studio e é totalmente dependente destes ambientes. É comum que códigos compilem ou rodem em máquinas específicas, gerando a famigerada frase "mas no meu computador funciona" ㄟ(ˊ)ㄏ. Também não são utilizadas suítes de automação de testes que visam garantir a qualidade dos builds.

19. Na sua organização, como é realizada a gestão de releases?

- As práticas de continuous deployment e delivery foram atingidas. A primeira, lida com a capacidade de publicar automaticamente builds nos ambientes de testes e homologação toda vez que um novo build for gerado. A segunda é prática lida com a capacidade de entrega de builds em ambientes de produção de forma automatizada e com governança, toda vez que um build for aprovado em um ambiente de homologação. Os times passam a experimentar testes e ambientes canários, também conhecidos como testes A/B (Link com uma breve explicação sobre testes canários <https://goo.gl/xXDLMo>).

- Os builds começam a ser implantados diariamente em ambientes de testes e homologação e as publicações em produção seguem calendários acordados áreas de negócio. É desenvolvida a capacidade de reverter publicações automaticamente em casos de instabilidade em ambientes produtivos, esta capacidade é controlada por ferramentas, evitando o risco de falhas humanas.

- Releases são publicadas em intervalos regulares em ambientes controlados. Ferramentas como Jenkins, GitLab, Microsoft VSTS(ou TFS) passam a ser utilizadas para movimentar as builds produzidas entre os ambientes de desenvolvimento, homologação e produção. Também são realizados smoke tests (testes de fumaça) automatizados que verificam a integridade dos ambientes após as implantações.

- A cultura de release começa a ser praticada. São utilizadas ferramentas de automação (Ex: Scripts em Powershell, Ansible, etc..) para automação de alguns passos de publicação. O processo ainda exige intervenção humana, mas por conta dos scripts, já é parcialmente documentado e pode ser executado sob demanda.
- Não existe uma cultura de gestão de releases. A entrega de um release nos ambientes de testes, homologação e produção é realizada de forma manual e sem o auxílio de ferramentas de automação. Geralmente desenvolvedores necessitam alterar (manualmente) apontamentos para bancos de dados e de configuração (ex: WebConfig em .Net ou web.xml em java). Existe uma tendência a erros por conta de falhas humanas.

20. Na sua organização, como é realizado o processo de monitoramento de aplicações?

- Possuímos ferramentas capazes de realizar a correlação de eventos de sistema e consumo de recursos, estas são capazes de prever quando um sistema entrará em um possível estado de falha. Quando necessário, a própria ferramenta pode realizar a auto escalabilidade do sistema/ação de correção, que é baseada em parâmetros configuráveis.
- Possuímos um ferramental capaz de monitorar recursos, serviços e realizar log de analytics (EX: Graylog, ElasticStack e DataDog) e que estão integradas com os processos e ferramentas de desenvolvimento. A partir disso, são gerados thresholds de desempenho mais assertivos, elaborados pelas equipes de Dev e Ops em conjunto. Ambas as equipes possuem acesso livre aos dados de monitoramento e são responsáveis pela melhoria no processo de monitoramento.
- Possuímos um ferramental capaz de monitorar recursos, serviços e realizar log de analytics (EX: Graylog, ElasticStack e DataDog). O monitoramento é realizado por uma equipe de operações dedicada à este propósito e quando necessário, ela aciona responsáveis por determinado serviço para verificação de problemas.
- Possuímos um ferramental básico para monitoramento de aplicações, que verifica sinais vitais de servidores como por exemplo: Consumo de recursos e status de serviços. O monitoramento é realizado pelo profissional responsável por determinada aplicação.
- Não possuímos um processo de monitoramento estabelecido, problemas são mapeados conforme reclamações de usuários das aplicações.

21. Na sua organização, como o feedback do usuário final é analisado e otimizado para alcançar a melhoria contínua?

- ☐ Coletamos este feedback a partir do monitoramento contínuo de mídias sociais e blogs, ou por meio de tickets encaminhados pela nossa central de serviços (Service Desk) e a partir destes dados, realizamos análises de sentimento dos usuários, visando a melhoria contínua de nossas aplicações.
- ☐ Utilizamos ferramentas padrões para coleta de feedback, e a partir de relatórios gerados por estas ferramentas, realizamos otimizações.
- ☐ Utilizamos técnicas como RCA (Análises de causa raiz), entrevistas, métricas e filtros de dados para coletas de feedback.
- ☐ Por meio de discussões de equipe, com base em dados de planilhas.
- ☐ Não há otimização para melhoria contínua.

22.No conjunto de ferramentas listadas abaixo, indique o grau de importância, considerando o contexto atual da utilização da cultura DevOps em sua organização

	Sem importância	Pouco importante	Razoavelmente importante	Importante	Muito importante
Ferramentas de controle de versão	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Ferramentas de gerência de configuração	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Ferramentas de gerência de build	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Ferramentas de gerenciamento de mudanças	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Ferramentas de testes	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

	Sem importância	Pouco importante	Razoavelmente importante	Importante	Muito importante
Ferramentas de integração	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Ferramentas de orquestração	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Ferramentas de provisionamento	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

23.No conjunto de atividades consideradas "core" para "DevOps/Release Engineers", indique o grau de importância das atividades abaixo, no contexto atual da utilização da cultura DevOps em sua organização.

	Sem importância	Pouco importante	Razoavelmente importante	Importante	Muito importante
(Integração) - Controle de código Fonte (SCM), incluindo estratégias de branch e merging, geramente realizado por ferramentas como GIT e Subversion (SVN)	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
(Sistemas de build) - Técnicas e ferramentas para realizar build e empacotamento de código-fonte e outros arquivos em um entregável (por exemplo, Ant, Maven, Makefile)	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
(Integração contínua) - Gerenciar ferramentas automatizadas de build e QA, como Jenkins, Bamboo e MsBuild. Também realizando a marcação/armazenamento de artefados em repositórios como JFROG e Nexus.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

	Sem importância	Pouco importante	Razoavelmente importante	Muito importante	Muito importante
(Ambiente Infra) - Definir e gerenciar ambientes de infra (Servers, VMs, Containers e ambiente cloud) para atividades como desenvolvimento, testes, pré-produção, e produção.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
(Testes) - Construir ambientes de testes/escalar execução de testes para grandes sistemas de build	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
(Delivery / Release) - Configurar e manter os pipelines de deploy e release para novas versões de projetos de software	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
(Monitoramento) - Monitorar ambientes de produção	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
(Upgrades de versão) - Atualizar ambientes de produção, possivelmente utilizando técnicas como canary release e rollback	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
(Otimização de pipelines) - Otimização de diferentes tarefas do processo de release, permitindo o avanço para a entrega contínua	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
(Scripts) - Automação de tarefas via scripts, em linguagens como Bash, Python e PowerShell	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

24.Quais são as fontes que você tem utilizado para adquirir conhecimento sobre ferramentas / processos sobre DevOps?

	Nunca	Raramente	Às vezes	Muitas vezes	Sempre
Treinamentos online (YouTube, Udemy, CloudGuru, LinuxAcademy, etc..)	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Fóruns (StackOverflow, Reddit, grupos em redes sociais, etc..)	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Blogs pessoais com conteúdo técnico	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Faculdade	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Cursos em escolas especializadas (Presenciais)	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Meetups / Grupos de estudo	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Livros	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

25. Gostaríamos que você classificasse (Em ordem decrescente, da mais importante (1) para a menos importante (7)) os principais benefícios que você entende que são gerados para as organizações que adotam e praticam a cultura DevOps.

A entrega contínua possibilita uma melhoria na eficiência dos deploys realizados

Melhoria na habilidade de monitorar, alertar e auditar mudanças no ambiente de produção

Obter transparência entre qualidade das aplicações e performance da infraestrutura de rede

Os conceitos de infraestrutura-como-código propiciam um melhor conhecimento para os desenvolvedores sobre os ambientes (desenvolvimento, testes e produção)

Aumento na resiliência e recuperações de desastres, dado que os ambientes são altamente reproduzíveis

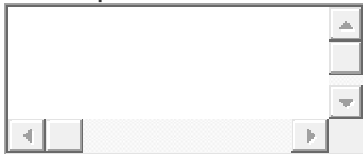
Automação pervasiva

Quebra de barreiras entre equipes de desenvolvimento e operações, pois um passa a entender melhor os desafios do outro, a eliminação destes silos gera uma maior empatia entre os times

26.Quais foram as suas principais dificuldades ao iniciar suas atividades em um ambiente DevOps? (A resposta neste ponto é livre, você pode expressar sua opinião sobre dificuldades culturais ou técnicas, por exemplo..)

A rectangular text input field with a light gray border. On the right side, there are three small square icons stacked vertically: a triangle pointing up, a square, and a triangle pointing down. On the bottom left, there is a small square icon with a left-pointing arrow. On the bottom right, there is a small square icon with a right-pointing arrow.

27.O que faria você não adotar a cultura DevOps em sua organização?

A rectangular text input field with a light gray border. On the right side, there are three small square icons stacked vertically: a triangle pointing up, a square, and a triangle pointing down. On the bottom left, there is a small square icon with a left-pointing arrow. On the bottom right, there is a small square icon with a right-pointing arrow.

Add new