Primeira entrevista

- Data da entrevista: 02/02/2023
- Duração da entrevista: 1:02:51
- Entrevistadores: Isaque Alves, Leonardo Gomes

Isaque Alves: Como Carla já mencionou, há uma pesquisa em andamento envolvendo a Universidade de Brasília, USP e a Universidade Politécnica de Madrid. Nosso objetivo é conectar os temas e informações que coletamos da academia e do mercado para validar nossa teoria. Os três primeiros, Jorge, Jéssica e Daniel, são professores da UPM.
Entrevistado: Eles são espanhóis?
Isaque Alves: Sim! Eu e Fábio somos da USP. Carla e Léo são da UnB. Identificamos um trabalho muito semelhante realizado pela UPM, a Universidade Politécnica de Madrid, e pela USP, que chegaram a resultados muito parecidos. A UPM identificou quatro tipos diferentes de estruturas de equipe; temos o DevOps, organizado por uma plataforma que facilita essa colaboração, onde desenvolvedores e operacionais trabalham juntos. Por outro lado, existem outros tipos de centros de excelência que auxiliam as equipes de desenvolvimento e também equipes de DevOps, que são assistidas por uma plataforma e têm um pouco mais de autonomia. Estes são os quatro tipos. Por outro lado, a USP também identificou quatro tipos de trabalho: departamentos, selos ou DevOps clássicos, equipes multifuncionais e a plataforma. Isso foi o ponto de partida para o meu trabalho.

Entrevistado: Pode voltar um pouquinho só para eu visualizar a caixa bem

Isaque Alves: se você quiser eu posso explicar um pouco melhor ou compartilhar esses trabalhos com você.

Entrevistado: Não, segue o planejamento, mas muito interessante.

Isaque Alves: Então em seguida a gente identificou que existia vários outros trabalhos que tratavam sobre isso que a gente chama de taxonomias, então a proposta foi harmonizar essas taxonomias; a gente já fez esse trabalho de harmonização foi um trabalho rígido, onde a gente parte de construtores que a gente leva a cultura automação os silos colaboração e os times e através disso a gente criou essa teoria essa taxonomia que a…

Entrevistado: Ontologia

Isaque Alves: Obrigado, essa ontologia, que a gente pode dividir em três, a gente tem para um software a gente tem os times que a gente consegue ver o product team, times de desenvolvimento, operacional, além dos times horizontais que são o enabler team e os bridge team; por outro lado, a gente tem que ter também o management, que a gente identificou nesses trabalhos product management, project management; além das atividades de Tecnologia de automação, da infraestrutura, do ciclo de vida, criar uma plataforma em infraestrutura como código e por aí, vai. Então essa é a nossa teoria que a gente quer validar agora através dessas entrevistas e pesquisas. Então nosso objetivo é colocar nossa Teoria em teste para identificar fraquezas, refinar e evoluir elas. E aí que a gente entra na nossa discussão e nas perguntas que a gente quer fazer para você. Primeiro a gente tem algumas dúvidas sobre a sua experiência pessoal e como é a estrutura do lugar onde você está trabalhando hoje? A gente já fez uma pesquisa básica, espero que não esteja errado e você já trabalhou no ministério da cultura, e eu queria saber um pouco sobre como foi a sua experiência lá e como você enxerga o departamento, ou melhor, vamos focar na empresa atual que você tá, qual o tamanho da equipe de TI? qual o papel que você desempenha lá? obrigado

Entrevistado: Você tá querendo que eu foque no Ministério da Cultura? É isso?

Isaque Alves: Não! Pode ser na empresa atual, não tem problema.

Entrevistado: O ministério da cultura já faz isso. 6, 7 anos; 5 a 7 2015/2018 então, acho que até o conceito DevOps era muito antigo. Iniciaram ainda naquela época e eu também não tava tão envolvido com os detalhes, né, do processo de desenvolvimento. enfim, faz muito tempo também, eu acho que não seria a melhor experiência, mas a experiência que eu tô hoje, eu tô no, eu tô no setor privado trabalhando na empresa. Como desenvolvedor, engenheiro de dados, engenheiro de software. Líder né, em vários aspectos e também fazendo um pouco papel de DevOps. Então a gente tem um lance do DevOps, na infraestrutura de dados. Então a gente tem o airflow, que é o orquestrador, a ferramenta que central para gente lá, e desenvolve em Python, né, tem um gitLab e tem uns pipelines para fazer Deploy Automático. Então a entrega continua. Então cada mudança que eu faço faço o commit e já cai lá na produção e já tá no ar, isso tanto para os, as pipelines né, que é cada pipeline é um projetinho Python arquivo Python. E para o também para infraestrutura do airflow. Então a infraestrutura também está codificada tem os dockerfiles do build. Tem uma parte também. Que é do Deploy no kubernetes. Que também tá separado no outro repositório, então eu diria que só nessa parte são três repositórios. Uma da, um que é o Build do airflow, da imagem, o outro que é o Deploy do airflow, que aí tem a ver com o kubernetes. E o outro que é os pipelines em si que é cada pequeno, cada pequena, cada pipeline para resolver um problema diferente que no final uso o airflow, depois a gente pode conversar sobre isso, tem um outro, uma outra, é um outro caso que nessa empresa também, que aí, é eu enquanto o engenheiro de software, não é muito mais desenvolvendo back-end de uma plataforma API back-end, fast api, enfim python, não vou entrar em detalhes sobre a steck, mas da mesma forma o gitlab com alguns pipelines, então todo commit acontece testes, ele dispara os testes se eu fizer uma tag ele faz Deploy. Faz os testes, build e o deploy na produção no kubernetes. Então a mesma forma tem o código, o código da aplicação e tem o código do Deploy. E com o kubernetes também tá no outro repositório, tem esses dois casos; até um ano e pouco atrás, eu ainda estava no governo. Tava trabalhando no ministério da economia e lá a gente também tinha uma airflow e tinha também uma infraestrutura que a gente mesmo mantinha e tinha alguma automação, mas acho que conversar sobre a minha experiência atual que eu acabei de explicar ela vai ser mais interessante. Tô mais e mais profissional. Tem outras pessoas, não foi só eu que fiz, então aprendi muita coisa do que já tava lá pronto, melhorei também. Enfim, acho que vai ser melhor.

Isaque Alves: ótimo, e como você avalia a maturidade de entrega contínua da sua equipe hoje?

Entrevistado: Como eu avalio? tem uma escala assim para avaliar? você quer que eu fale sobre problemas que eu enxergo?

Isaque Alves: não se você acha que

Entrevistado: Tem que melhorar, tem que melhorar.

Isaque Alves: eu acreditei no mais alto nível.

Entrevistado: Tá então.

Isaque Alves: Eu acredito que seja um pouco mais, pode falar.

Entrevistado: Tá como eu avalio, eu avalio que funciona, atende, mas tem que melhorar em vários aspectos. Vamos lá, primeiro que foi eu que levei a cultura de testes. Para lá, então, não existia teste e eu comecei a colocar teste, comecei a botar mono código sem precisar fazer com testes, e aí tu tá desenvolvendo tdd, e aí naturalmente inclui isso no pipeline. Mas os testes como etapa do pipeline, mas falta teste para muitas partes da aplicação, né, então, no seu quanto que isso é só DevOps ou é mais dev, até curioso entender um pouco melhor, mas é algo que no final afeta a qualidade. Na verdade afeta a nossa confiabilidade, né de que tá tudo certinho, e é um ponto assim, que que precisa rapidamente ser colocado né com prioridade. Quanto ao deploy, eu diria que o acho que fala de um aspecto, né que é o conhecimento sobre, esse Deploy no kubernetes, ele, o Deploy, O Job. Foi desenvolvido por outra pessoa

que tá lá no time, eu comecei a colocar a mão por uma necessidade de ter agilidade. Então eu não tinha no início, eu não tinha permissão no kubernetes, tanto para debugar a gente tem um outro ambiente, né homologação. Então eu não tinha acesso para ver o que tava acontecendo, queria debugar, e aí eles me deram permissão, e aí eu também me deram permissão para mexer no nas configurações né pela interface gráfica lá no Rancher; e e isso foi aconteceu de forma meio que desorganizada. a doc. Eu que pedi por uma necessidade, isso não, ou seja, não existe uma gestão ali desse papel, né, tava pensando naqueles modelos lá que você mostrou, né os tipos de DevOps que existem, não lembro os nomes, mas enfim, existe essa pessoa que cuida de várias outras coisas lá na empresa, então ela é DevOps de muitas outras coisas, e para a gente para o nosso time ele não dá não dá uma atenção. Não é dedicado, a gente precisa evoluir alguma coisa nesse pipeline, isso entra numa fila e muitas vezes, nem é feito, né, geralmente o que é feito é na emergência enfim, o que é urgente né, que é urgente, que é crítico, que precisa acontecer porque tá impedindo a produção, tem que entrar alguma coisa no ar. Então, eu diria que esse, essa autonomia do time, ela tá limitada, né do time poder evoluir a ferramenta, isso tem uma, tem um motivo os principais são questões de permissão, né de segurança, de acesso, de mexer nesses recursos no cluster, né de poder por exemplo adicionar uma variável de ambiente, eu acabei de desenvolver uma funcionalidade que requer uma variável de ambiente, no container da aplicação vai precisar de uma variável de ambiente, e a forma hoje de fazer isso ainda é um pouco manual que é no Rancher, no painel do Rancher e adicionar vara de ambiente; outra forma melhor. Seria a gente alterar os deployers do kubernetes, os arquivos de ambos, né declarativos, só que eu não tenho essa, essa experiência toda. Até conheço um pouco, mas eu mexo direto no Rancher e ao mesmo tempo isso não foi estruturado no time. Então assim qual que é a metodologia, qual que é o processo tradicional variável de ambiente. Isso não foi? Não foi disseminado, não foi construído esse processo e não foi capacitado para fazer isso. Eu diria que a maturidade do time é limitada então. Sintetizando o time tem autonomia de mexer no código de disparar o Deploy mas não tem autonomia de alterar o Deploy. Não falo nem o job. Pipeline, eu falo os os ativos. Os Os Como é que chama? os recursos que descrevem. O Deploy, e aí eu falei um exemplo de varal de ambiente, mas pode ter muitas outras coisas. Tudo que tá mais ligado ali é a configuração do Deploy na infraestrutura. Aquele lance lá todo do kubernetes; então isso é um buraco. Uma falha que a gente também nem discute, até legal, né falar aqui, que isso não é discutido a gente, nesse lance eu diria que nesse nessa disciplina do DevOps a nossa evolução é muito pela demanda, a doc né, mas claro a gente preserva e luta, né discutir muito essa questão da Autonomia a questão da automação aqui. Fez uma vez manual e fez duas vezes manual, pô, tá errado. Então vamos automatizar isso, e isso tira a autonomia, porque no final isso acaba impactando na agilidade, produtos, na entrega, enfim tem tudo uma amarração aí.

Isaque Alves: Entendi.

Entrevistado: Quer fazer uma pergunta? Eu tô…

Isaque Alves: Eu queria aproveitar esse gancho que você falou de automação para a gente pular para alguns blocos, a gente tem um bloco que é só sobre automação, no passado a gente identificou que essa automação ela tem um papel fundamental para organizar essas equipes; então eu pegaria agora, qual a sua percepção sobre métricas indicadores e até compartilhar informações, para o quanto isso impacta para eu conseguir automatizar um ciclo de vida de um software; se tem isso na sua experiência, como que é?

Entrevistado: Perdão, desculpa chegar mensagem perdendo um pouco. Fala de novo.

Isaque Alves: Mais focado em métricas e compartilhar conhecimento entre as equipes; se você vê uma diferença e se isso na sua vida no seu dia a dia tem um impacto para automatizar um ciclo de vida de um software.

Entrevistado: A relação entre métricas e automação você fala? **Isaque Alves**: Isso.

Entrevistado: Olha isso que eu acabei de falar. Esse problema, essa dificuldade, essa limitação. daria para metrificar? Dá, mas eu não sei se você está se referindo a métricas automatizadas, né painéis com métricas de tempo de demora de deploy

Isaque Alves: também

Entrevistado: Tempo de… enfim; bem, tô aqui pensando o que que a gente tem, a gente tem pouca

métrica sobre isso, né; as métricas que a gente tem, o tempo do build no gitlab, né, às vezes me atrapalha muito. Porque a gente coloca lá um processo e ele às vezes parece burocrático, porque a gente trava a branch main e ninguém pode commitar diretamente na main e aí tem a branch develop, e aí quando você faz commits na branch develop roda o teste automaticamente, para fazer o merge na main, e aí precisa passar por esses testes, e aí depois para fazer o deploy final, para fazer uma tag, aí ele enfim, no final demora um monte de tempo, né, meia hora eu diria e a gente até já identificou algumas melhorias por exemplo, o build da imagem é feito umas três vezes do zero. Porque a gente não configurou corretamente que gitlab para compartilhar lá, para ter lá um cash, sei lá, não lembro dos termos certos, mas eu sei que é possível, né, para ter, usar o rash interno, enfim no próprio gitlab; e esse é uma métrica né, enfim tá lá o tempo. Me mostrando toda vez o tempo que demora, toda vez que eu preciso fazer um path de emergência, demora uma hora no mínimo, tempo de corrigir e fazer o deploy; e de certa maneira, eu não faço outra coisa enquanto não terminar isso, então a minha atenção fica presa ali e isso impacta.

Isaque Alves: Então na sua opinião métricas e ter essa visualização dessas informações ela acaba auxiliando.

Entrevistado: sim

Isaque Alves: Manter essa automação

Entrevistado: Perfeito isso exatamente, quanto mais indicadores da qualidade, da agilidade, da fluidez de todos esses processos, a gente enquanto desenvolvedor. Consegue enxergar e justificar. E até estudar onde que tá o gargalo. Que no final vai ser o ideal é que seja tudo muito fluido. Tem uma sensação de ser simples, de ser fluido, de ser ágil, com certeza eu afirmaria isso.

Isaque Alves: continuando nessa pegada ainda de automação, mas agora trazendo para o lado de compartilhar conhecimento, imagina que a gente tem um time de desenvolvimento e um time operacional e a gente vai automatizar o ciclo de vida que esses dois trabalham, você acha que ao automatizar o ao focar esforços nisso, me auxilia a compartilhar conhecimento entre essas duas equipes.

Entrevistado: sim, com certeza, com certeza e eu diria que, como eu dei o meu exemplo, né da limitação da forma como foi de eu pedir para poder ter permissão para poder fazer e ainda assim foi digamos uma passagem de conhecimento meia boca. Porque ninguém parou para me mostrar eu tive que ir lá dar uma pesquisada e tal. E aí assim pensando né sobre os conhecimentos necessários para isso tudo funcionar e quem detém e qual que é o fluxo de conhecimento que tem que existir aí eu entendo que existe um conhecimento que é exclusivo de desenvolvimento. Que é Dev Dev mesmo, existe esse conhecimento da operação que aí é do cara que é exclusivamente operação e exclusivamente DevOps. Chega até a fazer a parte DevOps porque ele está estudando assim como a galera de desenvolvimento tá estudando muito mais as questões do desenvolvimento do Framework escalabilidade ali no sentido. Entre a aplicação; do outro lado o especialista devops é ele que consegue inovar né nesse âmbito aí da infraestrutura é ele que tá preocupado com as tendências, ele que está estudando as novas ferramentas de infraestrutura, de enfim de organização e automação e eu acho que o DevOps é um encontro. Entre esses mundos. Porque por mais que a pessoa devops de operação, né especialista ele saiba fazer. A automação ele para cada caso, para cada aplicação, ele precisa entender os meados daquela aplicação, né até questionar e provocar o desenvolvedores para saber se tá tudo ali. Tem a questão da escalabilidade, vai ter que acessar uma outra rede, vai ter que ter uma para um certo serviço ter mais recursos de memória, sei lá, outro tipo de setup. E aí eu acho que, eu sou muito né, cara, me conhece muito defensor do compartilhamento, né da colaboração e eu consegui enxergar muito claramente em todo, na verdade eu conseguiria enxergar isso mesmo que não fosse um DevOps, né no próprio processo de desenvolvimento defendo muito isso então talvez eu seja até um pouco tendencioso para falar sobre isso, mas sem dúvida nenhuma pela minha própria experiência eu sinto falta de que houvessem algumas, primeiro algumas sessão para a gente corrigir construir junto essas melhorias que eu falei que às vezes são postergadas né de priorizadas, então a gente construir juntos, né o time de desenvolvimento junto com o especialista devops, para que tanto a gente consiga? A gente desenvolvedor, né, consiga garantir os requisitos finais do produto. Estejam sendo atendidos. Quanto para que a gente aprenda e tenha autonomia, né, para não depender dessa pessoa. Para poder ter agilidade no final, o objetivo é ter agilidade. Então eu acho que é natural, né essa esse movimento do desenvolvedor do Tech Lead. Os mais seniores eles irem se

transformando em um certo DevOps às vezes mais especializado naquelas stack, naquele cenário ou como eu não conheço muito bem para usar os termos melhores, mas mais especializado no Deploy da aplicação do que na gestão de recurso sei lá do que e outras coisas do mundo Devops.

Leonardo Gomes: Muito bom (nome do entrevistador). Eu ia fazer uma pergunta exatamente disso que você acabou de falar que no final das contas então você acredita que tanto essa parte de compartilhamento de conhecimento o como você falou o Tech Manager o Tech, a pessoa que está comando na equipe ela ficar mais por dentro dessa parte de DevOps, tanto essas equipes multifuncionais no final das contas então elas são muito importantes para um alto e Médio nível de compartilhamento de conhecimento entre a própria equipe. Era isso que você queria dizer.

Entrevistado: Sem dúvida, precisa ter alguém que faça a ponte. Não dá para ter um time de plenos e juniores e do outro lado só e os Developers especialistas para… eu acho que se não tiver alguém que tenha um pouco mais de experiência para fazer a ponte não vai ter essa percepção do que eu tô narrando aqui. De que tá faltando isso de que enfim consiga dedicar tempo para ensinar gerir fazer a gestão desse conhecimento no time desenvolvimento. Por mais sênior o DevOps, o especialista DevOps seja, ele ta preocupado com o time dele, o time aqui tem outra questão, a lógica do time, a lógica da gestão do conhecimento, especialmente nessas questões mais complexas, né que o devops traz, que envolve a produção tem que ter experiência de produção, né conhecimentos mais amplos de ti. Redes, recursos, backups e Enfim, sem dúvida essa pessoa aí, ela é Ela é crucial.

Leonardo Gomes: então dando fim, você acredita que toda essa perspectiva é de multidisciplinaridade da equipe. No final das contas ela pode acabar tendo um costume. Pode trazer automatização do ciclo de vida, então você acredita que essa multidisciplinaridade tende a automatizar o ciclo de vida do software.

Entrevistado: Sim, sim. Multidisciplinaridade. Vamos lá. A gente fala muito sobre full stack, né desenvolvedor foi full stack que é o cara que faz tudo e aí hoje em dia tudo é cada vez mais coisa. E aí ser um full stack é cada vez mais questionável. Qual full stack? Que significa realmente full stack. Talvez seja muito mais na questão do desenvolvimento. Talvez não no DevOps que cuide dessa, que chega nesse nível. Mas sim, né sobre a pergunta se a multidisciplinaridade ela…

Isaque Alves: A gente pode expandir também para uma

Entrevistado: ela é fundamental.

Isaque Alves: Para uma equipe multidisciplinar, não precisa ser somente um indivíduo, mas que tem alguém que entende de operação, alguém que é sênior no desenvolvimento isso ajuda então na automação.

Entrevistado: Sem dúvida. Sem dúvida. Porque a automação surge da experiência, né. As pessoas fazem, tem muita gente que faz algo de uma forma repetida sem perceber que é repetido. então assim a oportunidade de automação ela surge da experiência, né de que ah eu vi algo assim em outro lugar dá para fazer aqui, eu já fiz algo parecido e outro cenário, eu vou fazer aqui. Ou até a sensibilidade de perceber que isso aqui tá sendo muito manual porque está tomando muito tempo. E eu percebo que eu tô repetindo o código, eu tô repetindo comandos, como automatizar? eu acho que tudo isso é experiência. Tanto experiência de tempo de vida quanto de diversidade né de disciplinas mesmo. Mesmo que a gente esteja falando de pessoas que automatizam as coisas. E outra que automatiza no backend por exemplo nesse time que eu tô teve uma pessoa colaborador estrangeiro ele, húngaro, que ele que ele criou um hook no Git que para cada commit ele antes de terminar o commit ele executa um script Bash, no git você consegue fazer isso. Tem uma página lá de hooks você pode configurar os hooks, o que que ele fez? Ele criou o hook para rodar o lint antes de commitar? Então é um arquivo um bath lá que roda lint no frontend e roda lint no backend se der, se não passar printa lá o erro e aí o desenvolvedor tem que ir lá, só faz o commit se passar no lint, né, pô, eu acho isso fenomenal. Eu não conhecia, não conhecia nem esse lance do Git e nem tinha tido essa ideia de, Ah dá para botar esses lints dentro do, ou seja, automação no ambiente local esse abriu para mim uma outra miria de coisa. porra, eu posso usar o computador local desenvolvedor, que a gente tem o ambiente local, tem um ambiente de teste. O repositório e tem a produção enfim tem vários momentos que a gente pode automatizar em vários momentos quanto antes e é isso me lembra, né de outras. Outras teorias. Outros princípios. Que quanto antes você identifica o problema. ou seja, quanto mais rápido o ciclo, né de fazer uma mudança identificar um problema e corrigir melhor então porra. Eu pensei, o melhor cenário é esse tudo que for local é mais rápido. Tudo que eu

puder fazer novamente local da pessoa para evitar porque se ela só commitar fazer o lint lá no repositório melhor, depois receber o e-mail. Ela tá focada em outra coisa, já perdeu o bonde. Só dando o exemplo da multidisciplinariedade, quanto que só essa experiência dele já contribuiu me ensinou um monte de outras coisas, e eu acho que é só isso que

Isaque Alves: E esse conhecimento dele se difundiu com o resto da equipe então, com essa automação que ele criou todo mundo passou a usar.

Entrevistado: todo mundo passou a usar porque tá automatizado, para todo mundo usar mesmo sem o pessoal saber. Então o que ele fez, ele criou um container novo, esqueci até, um container bem efêmero lá que roda no compose UP que ele simplesmente copia esse arquivo Shell para pastinha de hooks do Git. Né, então, toda vez que você fizer o Up, ele vai copiar a primeira vez ou atualizar o arquivo se ele fizer mudança nesse hook. E de repente a galera começou a, não tô conseguindo commitar, tá demorando aqui, porque demorando? Porque tá fazendo lint e depois não consegue commitar porque tá dando erro. E aí eu tive que explicar, não foi o processo da melhor forma. O cara fez lá falou para mim que eu sou o TechLead e… Enfim, eu deveria ter falado para galera, mas também ninguém pensou, ele nem podia ter falado não precisa passar por mim, né questão aí também de agilidade, mas a galera todo mundo ficou sabendo meio que dá força. porra, não consegui. Ah, entendeu? Aí eu mostrei só esse arquivo que dá uma olhada no arquivo. Entendeu? Lá e detalhe porra tô precisando comentar mas tem um lint aqui que outra pessoa commitou porque foi bem na transição, acabou mandando um código errado ou fez um commit pelo gitLab direto e aí não passa porque não roda o hook e aí como é que eu desligo para fazer um convite à força? Vai no arquivo aqui comenta essas linhas, roda de novo o up para ele copiar lá para a pastinha do hook do git hook. E aí tu consegue commitar se ele fizer o lint. E aí a pessoa acaba aprendendo mas muito pela necessidade ali. Eu acho que esse conhecimento, esse tipo de conhecimento precisa ter Dojo sessões assim de compartilhamento, eu sinceramente acho fundamental. E aí só expandindo um pouco essa reflexão, eu acho que a tendência. Dos times Hoje em Dia dos próximos cada vez mais é de ter essa autonomia então, é autonomia de conseguir fazer o deploy em produção ou seja conseguir cumprir o ciclo de DevOps completo, sem depender de um especialista terceiro, né de uma área terceira, então acho que essa disciplina ela vai cada vez ser mais naturalizada, né democratizada e que precisa que isso seja parte né dessa rotina de compartilhamento interno dos times. De que o por exemplo o onboarding Né? De novo desenvolvedor, passe por isso, né da sua pessoa saber o que que é aplicação, também saber o que acontece depois. todos os detalhes até chegar lá na produção de fato.

Isaque Alves: Perfeito mudando um pouco, mas é num a gente segue na mesma parte de automação, nos nossos estudos a gente identificou algumas empresas, que adotaram o que elas chamam de times de plataforma, que são times que prestam que criam uma plataforma por exemplo para fazer o Deploy ou para automatizar ou para solucionar algum problema de outras equipes?

Entrevistado: Eu tenho um exemplo.

Isaque Alves: No seu existe essas aqui.

Entrevistado: o airflow, eu vejo muito por aí o time, o time Central que mantém o airflow para os times de ciências de dados nas corporações. Então esse time é responsável por manter a infraestrutura de airflow e as várias instâncias de airflow rodando mas que os times que usam só fazem o uso não precisa se preocupar, né, para eles é paz, né, plataforma como serviço e tem o time Central que faz isso. Olha, eu entendo concordo, né, consigo até discutir um pouco sobre isso, porque eu já tinha enxergado isso, já pensei isso no ministério da economia a gente faz um pouco isso poderia a gente fazer. Eu falo meio que me sinto lá, interajo com o pessoal direto, mas nessa empresa agora não porque ela é até bem menor. Mas saindo do airflow, pensar aqui, né nessa empresa, times de plataforma, olha tem esse tem esse cara. Que era duas pessoas, agora só ele que é o cara DevOps. É ele que mantém tudo, ele é o último, a última fronteira ali da infraestrutura.

Isaque Alves: Ele mantém essa plataforma que auxilia os outros times então

e você age em cima?

Entrevistado: É… Rancher, cluster, jupyter hub pra gente, enfim, clusterdasc.

Isaque Alves: mas ele centraliza isso em uma ferramenta, em uma plataforma ou as pessoas têm acesso? Como as pessoas têm acesso a esse serviço?

Entrevistado: Isso centraliza nele enquanto o gestão, tudo roda no Kubernetes através do Rancher na interface gráfica ou Rancher, os deploys estão version ados no gitlab interno, eu tenho acesso porque eu pedi, mas não mexi muito porque eu prefiro mexer, assim inclusive tem até uma disparidade. Porque tem coisas que daria para fazer tudo só por ali como eu falei no começo. Eu faço muita coisa para interface gráfica do Rancher por exemplo adicionar uma variável de ambiente, mas daria para ser feito lá é né ficar persistido, inclusive rola um descompasso, porque o que foi feito diretamente no Rancher não volta para esses deploys. Então é uma certa bagunça ali na gestão; essa infraestrutura não, a priori não, eu sou uma exceção diria, mas ela não é compartilhada, os desenvolvedores usam, são clientes, usuários, não adentram nessa, digamos, programação da infraestrutura via regra.

Isaque Alves: Não, perfeito, a gente viu também, acho que é uma informação boa aqui isso ajuda a diminuir a carga cognitiva dos desenvolvedores, ao invés dele ter que focar na parte do produto, entender os requisitos, desenvolver… ele ainda tem que aprender também a parte da infraestrutura, isso é importante, só que a gente acaba diminuindo essa carga para ele poder focar no que a gente precisa para ele agora, em relação a essa plataforma e os princípios e os aspectos de DevOps, você acha que isso contribui que melhora?

Entrevistado: Qual a plataforma você fala?

Isaque Alves: Essa plataforma que vocês têm por exemplo ou esse conceito de ter uma plataforma que eu vou auxiliar e eu vou estar prestando serviço interno para os desenvolvedores dos times.

Entrevistado: É eu acho que isso depende do tamanho do time, né da maturidade do time e da pressão de entrega. Como o time? Eu acho que eu sou o único Sênior de backend. Tem vários Júniors e alguns plenos, tem outro sénior de frontend que também tá começando a mexer um pouco nesses, nessa parte DevOps. Eu acho que concordo com você essa questão da carga cognitiva para um Júnior, pleno aprender tantas coisas diferentes, seria excessivo e no cenário desse não existe uma pressão pela excelência na automação porque o time não precisa, não vai trazer tanto ganho para o time, Né? Então, eu acho que depende muito dessa conformação toda. Enfim, é isso, acho que depende.

Isaque Alves: não, perfeito sobre

Entrevistado: da maturidade, dos tamanhos, das praticidade do produto.

Isaque Alves: A gente ainda tem mais alguns minutinhos e uma série de perguntas. Léo, você quer, tem alguma coisa que queira fazer?

Leonardo Gomes: Tem uma em específico que eu gostaria de fazer, aí a gente é a parte para outra parte de outros blocos, mas é tá envolvendo tudo que você acabou de falar (nome do entrevistador) tá assim tanto nessa perspectiva de ter um time que são que utilizam a plataforma né utilizam. Essa infraestrutura que é criada. Então mas estão começando até essa integração dentro da parte de desenvolvimento manual commit quando há essa parte da em contato com o, que começa a entrar em contato com a parte de infraestrutura. Mas aí eu queria fazer uma pergunta mais em relação a sua visão em relação a esse compartilhamento de compartilhamento de conhecimento, mas a gente entra na perspectiva de que esse compartilhamento no final das contas, ele afeta a estrutura da equipe. Ela afeta toda a equipe tanto em questão de comunicação quanto na questão da criação de silos, entre outras perspectivas. E aí eu queria perguntar para você se esse compartilhamento de responsabilidade ele no final das contas ele reduz essa criação de Silos e os conflitos organizacionais tanto na perspectiva de burocratização de toda a estrutura, o segmento de devops, quanto também na parte de desenvolvimento também?

Entrevistado: Se o compartilhamento de responsabilidades, ele contribui. contribuir para que?

Leonardo Gomes: Se ele reduz na verdade os silos e os conflitos organizacionais.

Entrevistado: os silos e os conflitos, sem dúvida boa pergunta isso. Inclusive me lembra muito do Ministério da economia a estratégia que eu usei. Deixa eu contar essa história. Lá eu criei um time de engenharia de dados e migrei um monte de pipelines que estavam utilizando ferramentas proprietárias,

antiquadas, o airflow. Então pipeline de leitura de banco, transformação de dados, carregar… tipo ETL né mais tradicionais e naquele, naquela situação eu estava lotado na Coordenação Geral direto no gabinete, mas que nesta secretaria no gabinete da secretaria, então a gente fazia a gente produzia painéis para o secretário, mas nessa secretaria existem vários outros departamentos que tem seus painéis, tem suas bases de dados que tem os mesmos problemas mesmo desafios. E aí eu iniciei uma estratégia de… vamos oferecer o serviço, primeiro vamos oferecer o serviço de construção de ETLs, nas áreas existem pessoas especializadas que sabem programar SQL, mas para fazer um pipeline no airflow precisa saber de Python né, mexer com o airflow e entender um pouco sobre o Deploy na nossa arquitetura lá, né, no repositório e tal. E aí eu defini, inicialmente, que a gente iria rodar o airflow a gente ia manter a infraestrutura airflow rodando, que assim eu pensei que a gente pudesse fazer esse processo de maneira gradual, né de compartilhar a responsabilidade com eles. Inicialmente. Fazer de uma forma com que entregasse o resultado final é o valor para que a necessidade dele dentro, a responsabilidade deles, a responsabilidade final dele dentro da secretaria, do departamento dele fosse atingida. Então o cara precisa resolver o problema, o chefe não quer saber se ele vai resolver com airflow, sem airflow ele precisa resolver então a gente vai ajudar ele a resolver para que essa estratégia toda, né de transição e de criar uma comunidade de dados não fosse impedida por casos mal sucedidos. Um experimento que deu errado então a ideia inicial foi: vamos fazer junto com eles. Então a gente, eu mesmo dei várias aulas e ensinei airflow, ensinei programar, mas no início eu tava fazendo para eles os pipelines, no segundo período, no segundo momento eu dei a permissão para eles escreverem os arquivos Python os scripts, ensinei a rodar o airflow localmente e mostrei para ela se vocês fizeram deploy, um commit no repositório já vai cair em produção; expliquei para ele e falei agora você tem responsabilidade. A autonomia de mexer em produção, fez local, rodou, fez um commit, fez push, resolveu, massa, mas existe um outro nível depois desse. E aí assim ficou por aí porque é complexo. Não, a minha ideia é avançar mais, mas é um processo exatamente porque a gente não pode chegar e falar olha, vocês vão fazer tudo aí vai rodar airflow, vai fazer o negócio e essa é a diretriz, não vai funcionar. E ao mesmo tempo que não dá para gente centralizar tudo e eles ficarem dizendo só eu quero que ler essa tabela roda, cria essa view aqui nesse banco. Também não ia funcionar, não ia escalar. Então o próximo passo. Seria ensinar eles a rodar o airflow na infraestrutura deles, né configurar tunar o airflow e tal, mas não conseguimos chegar nesse ponto. Na verdade o segundo passo nem ficou tão, tão concluído que é eles terem domínio do airflow no ambiente local e saber fazer o deploy em produção, não é nem rodar o airflow em produção, né que a gente tá, continua rodando. Então esse foi meio que um projeto né, que relativamente foi bem sucedido, mas que ele ainda tem etapas para serem avançadas que dependem da maturidade das pessoas do outro lado. Aí tem a questão da carga cognitiva aí, e no final das contas qual que é o interesse meu do meu time que é o core. da plataforma lá o que tá mantendo, aí nesse tá rodando, tá mantendo a plataforma produção para todos os outros times e tá fazendo esse processo né de disseminação desse conhecimento e na verdade de transmissão e responsabilidade né transferência de responsabilidade, o ideal é que a gente não precisa rodar a plataforma para todos para que a gente possa, que rodar da manutenção. Sempre tem uma coisinha ali para que a gente possa focar em outras coisas, sempre na minha cabeça a ideia é essa, automatizar para ter liberdade de fazer coisas diferentes novas. Então mas isso depende de fatores, da mesma forma, olha que doido, da mesma forma existe essa relação entre nós, na secretaria e no ministério como um todo, que tem um departamento de TI próprio. sacou? a gente não é departamento de TI, a gente é um time especializado na secretaria que dá suporte para outros times menos especializados na secretaria, mas existe uma infraestrutura de TI do ministério que faz um monte de outras coisas, por exemplo rodar o setor de e-mail, manter a internet no ar, as fábricas de software, e para gente prover máquina virtual, para a gente rodar o airflow e fazer a manutenção a gente gostaria que eles também provém outras coisas, por exemplo, o airflow fosse responsabilidade deles e não nossa. Né para que?

Isaque Alves: Entendi.

Entrevistado: Para que a gente pudesse continuar meio que representando a secretaria nessas questões de airflow. Então assim, a gente continua sendo os que mais entende de airflow na secretaria, mas quem roda hoje não é mais a gente é a DTI do ministério, e quem tá usando somos nós e os outros times, os outros times não estão usando o airflow que não é gente que mantém beleza, mas a gente consegue dar

manutenção diretamente, a gente né, com as configurações do deploy. O problema é que a gente roda as máquinas virtuais, aí teve uma pane, e aí ele não tinha o backup, enfim, tem coisas que só eles podem prover. Que eles têm recursos para isso, então assim da mesma maneira a DTI não tem capacidade de absorver, por limitações que eles têm muitos problemas precisam de um especialista lá que entendesse de airflow minimamente, né, fosse estudando para poder manter, e a decisão é pragmática? É de que não vale a pena por enquanto. Então existe um custo aí que não vale a pena a responsabilidade que não vale a pena transferir por o risco da gente ficar seu recurso que vai botar para lá para eles vai botar uma pessoa que não sabe vai cair a pessoa não vai saber fazer, e aí afetou todo, toda essa estrutura.

Isaque Alves: Entendi então, achei interessante sim, então você diria que quando a gente não tem essa, não compartilha essa responsabilidade, ou seja, as equipes têm papéis muito bem definidos acaba tendo uma transferência de tarefa? Um exemplo é o time de desenvolvimento e o time de operação ele desenvolve ele transfere a atividade ou responsabilidade para o time de operação garantir a produção disso.

Entrevistado: A sua pergunta foi se não tem compartilhamento de responsabilidade…

Isaque Alves: quando não tem esse compartilhamento de responsabilidade ocorre essa transferência de tarefas?

Entrevistado: Sim, sem dúvidas, a gente fala responsabilidade, mas é trabalho né, trabalho de manter aquilo funcionando.

Isaque Alves: Também, quando a gente, isso!

Entrevistado: É que pode estar bem feito, nunca vai dar problema, mas você é, você tá atento, você tá monitorando, você é o responsável por aquilo. Mas e aí claro. depende de quão bem efeito estiver a responsabilidade ela é leve, não é algo que pesa.

Isaque Alves: A gente já vai partir para finalizar também porque a gente definiu um time box de uma hora, então acredito que pelo pelos minutos a gente tem tempo para mais uma pergunta, Leo quer fazer? tem alguma aí?

Leonardo Gomes: Eu quero fazer e eu acredito que vai ser para a gente conseguir agora concretizar alguns pensamentos nessa parte de compartilhamento de responsabilidade e eu acho que, eu acho que a pergunta que eu tava mais esperando. Quando eu comecei, quando eu trouxe esse, essa discussão que no final das contas o que você acredita que possa promover essas responsabilidades entre as equipes? E o que que você acha, qual que é o grau que isso afeta, maior ou menor, é qual, se esse grau no final das contas ele afeta promovendo o compartilhamento. Então, por exemplo, tanto a questão do que Isaque falou da segregação, mas do outro aspecto do compartilhamento de responsabilidade, você acredita que é isso? Posso promover? Ainda mais por exemplo uma comunicação frequente, você acredita que no final das contas é a gente consegue promover também é uma melhor uma melhor divisão desses aspectos e a assim que também promover também uma um compartilhamento de conhecimento também de tudo que a gente já falou aqui e aí eu queria finalizar também já que é para a última pergunta é para você detalhar essa sua visão responsabilidade, desse compartilhamento de responsabilidade.

Entrevistado: Olha eu acho que existe aí. O princípio da autonomia. Eu acho que todo time ele deveria ter isso como Horizonte, ter autonomia. Tanto o time de devops quanto o time de desenvolvimento. o tipo de operação quanto o time de desenvolvimento. Eles precisam ter autonomia um do outro, então é. E aí como eu dei o exemplo. Existe ali? Dependendo das capacidades, das maturidades esse equilíbrio ele tá mais para lá ou mais para cá. Quanto dessa responsabilidade de manter isso aqui vai ficar aqui ou vai para lá pro outro time? Não é isso, depende sempre dessa, dessa realidade material mesmo concreta. Que como é que a gente queira não vai fazer mágica, né, as pessoas são limitadas e a gente não pode transferir esse conhecimento porque aí tem um risco associado. Então assim esse equilíbrio eu acho que ele é dinâmico, ele precisa ser trabalhado em revista o tempo todo, não só pelo conhecimento das pessoas serem dinâmicos pela maturidade das tecnologias também. Serem dinâmicas. Eu acho que precisa ter uma interação sempre né, então esse isolamento. O isolamento de conhecimento eu acho que ele sempre é um problema. Então eu defendo muito práticas, né de manter uma, de manter canais de comunicação eventos, né de integração, ações mesmo de intercâmbio continuamente de forma recorrente para que esse equilíbrio seja repensado. Seja revisto seja, para que o até o ajuste fino seja, seja

trabalhado, né para aperfeiçoar quem tá falando assim, a gente vai se reunir. Deve haver operações para corrigir bug para evoluir, para repensar essa divisão de responsabilidade. Agora ele vai transferir um pouco mais responsabilidade para vocês, a gente vai capacitar vocês. E enfim tem que ter um projeto para isso. Pensar nesse projeto. Então eu acho que primeiro é isso. Tem que ter essa recorrência da, como se fosse uma retrospectiva, né uma recorrência dessa autonomia né dos dois times. Se o time de operações tá gastando muito tempo com a operação desse projeto aqui, que não precisaria; ao contrário também. O time de desenvolvimento está limitado tá precisando tá recorrendo muito ao time de operações também não precisaria. Como reduzir. Essa, essa dependência; reduzir a dependência de certa maneira é reduzir a interação eventual, mas para suplementar isso é que tem que ter uma interação recorrente planejada intencional, né para corrigir. Esses eventos, essas eventualidades e pensar no futuro e identificar ali quem que é a pessoa? No time de operações de desenvolvimento, é que vai fazer a ponte com operações pelo menos para esse projeto aqui em específico, transferência daquele conhecimento, daquela responsabilidade. Enfim, eu entendo que são domínios diferentes, eu entendo que naturalmente estão tão umbilicalmente ligados. E que precisa ter uma disciplina de refinamento de evolução. E sim. A ideia também não é transformar o desenvolvedores em Developers, né, Engenheiros de operação e nem os engenheiros de operação especialistas naquele projeto. E sei lá até desenvolvedores daquele projeto, mas é encontrar o equilíbrio. Perfeito aí entre essas pessoas, conhecimento e essas responsabilidade, legal a conversa.

Isaque Alves: Para você é importante ter essas equipes, mesmo que essas equipes, dependendo da estrutura da empresa dos objetivos da empresa, essas equipes trabalham junto para atingir mais frequência de Deploy ou se não precisa menos frequência de Deploy então.

Entrevistado: Sim, sim, é responsabilidade sempre existir. O grau maior ou menor, nem que seja fazer o commit certinho. Ou que fazer o pull request certinho, né que saber fazer a tag que vai disparar o Deploy né, e que tenha teste para não quebrar, enfim, tem tantos níveis de responsabilidade que acho que sim, tem que ter essa, essa interação.

Isaque Alves: Não, perfeito, a gente tem várias hipóteses e eu tenho certeza que a gente conseguiu validar algumas delas e colocar em jogo outras. Então eu só tenho a agradecer pela sua disponibilidade, por falar com a gente. Eu queria pedir o seu e-mail, se você puder me mandar depois porque através do e-mail a gente pode te mandar a transcrição dessa entrevista que agora a gente vai fazer a transcrição dela e através de vários métodos de análise e de coleta a gente vai tentar chocar elas com a nossa as nossas hipóteses para evoluir a nossa teoria, então ao fim de todo esse trabalho, eu gostaria de enviar e compartilhar as informações que a gente, e o conhecimento que a gente adquiriu com você. Pode ser?

Entrevistado: Claro agradeço Claro, por favor, agradeço muito, se você fizer isso.

Isaque Alves: Então, Obrigado,

Entrevistado: Te mandei lá o e-mail.

Isaque Alves: Obrigado, tenha um bom dia. Qualquer coisa a gente entra em contato.

Entrevistado: Beleza, obrigado Isaque. Obrigado Leonardo. Valeu pela conversa.

Isaque Alves: Tchau!