Fase 1: apresento uma visão geral da teoria (diagramas) + instâncias

A ideia e objetivo da primeira fase é validar o diagrama e a teoria em geral para em seguida mostrar as instâncias.

Figma

Fase 2: Valida as proposições e hipóteses que são de alta prioridade

Aqui eu organizei as proposições que têm relação, assim eu discuto elas por temas e facilita a validação.

hypothesis_proposition_generated

1. Automation, Collaboration, and Responsibility

Pergunta (P3):

Em sua experiência, você já observou a relação entre **automated lifecycle management (ALM)** e a **colaboração entre times**? Existe uma influência entre eles?

- A implementação de ALM **aumentou ou diminuiu a necessidade** e a frequência de comunicação entre as equipes?
- Um aumento na colaboração **facilitou ou acelerou** a adoção e o aprimoramento das ferramentas de automação?
    - Pode fornecer exemplos específicos de como a automação se tornou um **facilitador para a colaboração** ou vice-versa?

P3. AUTOMATED APPLICATION LIFE-CYCLE MANAGEMENT IS ASSOCIATED WITH COLLABORATION. COLLABORATION IMPACTS AUTOMATED APPLICATION LIFE-CYCLE MANAGEMENT AND VICE VERSA. AUTOMATION AND COLLABORATION MUTUALLY FACILITATE THE ADOPTION OF THE OTHER, SO THEY ARE COMPLEMENTARY

Categoric relationship	Automation
	type

			Automated application life-cycle management	Automated infrastructure management
Collaboration	frequency	daily	H3.1-H3.1’	
		eventual		
	quality	high	H3.3-H3.3’	
		median	H3.5-H3.5’	
		low		

• H3.1 Teams using automated application life-cycle management are associated with daily collaboration
• H3.1’ Teams with daily collaboration are associated with automated application life-cycle management
• H3.3 Teams using automated application life-cycle management are associated with high collaboration
• H3.3’ Teams with high collaboration are associated with automated application life-cycle management
• H3.5 Teams using automated application life-cycle management are associated with median collaboration
• H3.5’ Teams with median collaboration are associated with automated application life-cycle management

Pergunta (P15):

Você acredita que ao ter AUTOMATED INFRASTRUCTURE MANAGEMENT facilita a ter uma ou melhorar responsibility/ownership sharing?

P15. AUTOMATED INFRASTRUCTURE MANAGEMENT ENABLES RESPONSIBILITY/OWNERSHIP SHARING

Categoric relationship			Automation	
			type	
			Automated Infrastructure Management	Automated Application Life Cycle Management
Team	responsibility/ ownership sharing	full sharing	H15.1	
		medium sharing	H15.2	
		Minimal or null sharing		

• H15.1: Teams relying on automated infrastructure management are associated with full responsibility/ownership sharing.
• H15.2: Teams relying on automated infrastructure management are associated with medium responsibility/ownership sharing.

Pergunta (P16):
E se eu expando isso para automatizar todo o ciclo de vida da aplicação, você acha que isso impacta no compartilhamento de responsabilidade entre os times?

P16. AUTOMATED APPLICATION LIFE-CYCLE MANAGEMENT ENABLES RESPONSIBILITY/OWNERSHIP SHARING

Categoric relationship	Automation	
	type	
	Automated Infrastructure Managem	Automated Application Life Cycle Management

			ent	
Team	responsibility/ ownership sharing	full sharing		H16.1
		medium sharing		H16.2
		Minimal or null sharing		

• H16.1: Teams relying on automated application life-cycle management are associated with full responsibility/ownership sharing.
• H16.2: Teams relying on automated application life-cycle management are associated with medium responsibility/ownership sharing.

Pergunta (P30):

Manoel e outros entrevistados me informaram que uma colaboração de alta qualidade entre dois times, foca em resolver ou automatizar um gargalo entre dois times. Você concorda que ao colaborar eles reduzem essa dependência e deixam o time mais autônomo?

P30. A high quality collaboration reduces dependency, increasing team autonomy.

determinant			Collaboration		
			quality		
			hight	median	low
team	Autonomy	self organization	H30.1		
		dependent	H30.2		

H30.1, H30.2: A high quality collaboration focused on reducing dependency is associated with an increase in team autonomy.

2. Cross-Functionality and its Enablers

Pergunta (P5):

Você acha que se eu tenho um time cross-functional/skills em uma equipe influenciam o nível de **colaboração (frequência e qualidade)**?

P5. IF A TEAM IS CHARACTERIZED BY CROSS-FUNCTIONALITY/SKILLS THIS WILL INCREASE COLLABORATION

Categoric relationship			Team	
			cross-functionality/skills	
			true	false
Collaboration	frequency	daily	H5.1	
		eventual	H5.2	
	quality	high	H5.3	
		median	H5.5	
		low	H5.4	

- H5.1 Multidisciplinary/poly-skilled teams (i.e., teams with all the necessary skills such as development, infrastructure, etc.) are associated with a daily collaboration with other teams
- H5.2 Multidisciplinary/poly-skilled teams (i.e., teams with all the necessary skills such as development, infrastructure, etc.) are associated with an eventual collaboration with other teams
- H5.3 Multidisciplinary/poly-skilled teams (i.e., teams with all the necessary skills such as development, infrastructure, etc.) are associated with a high collaboration with other teams
- H5.4 Multidisciplinary/poly-skilled teams (i.e., teams with all the necessary skills such as development, infrastructure, etc.) are associated with a low collaboration with other teams
- H5.5 Multidisciplinary/poly-skilled teams (i.e., teams with all the necessary skills such as development, infrastructure, etc.) are associated with a median collaboration with other teams

Pergunta (P17):

Na sua experiência, você já identificou que equipes CROSS-FUNCTIONALITY/SKILLS tem mais facilidade em compartilhar conhecimento?

- Essas equipes se engajam mais nos compartilhamentos?

P17. SKILLS/KNOWLEDGE SHARING IS A PROPERTY OF TEAMS CHARACTERIZED BY CROSS- FUNCTIONALITY/SKILLS

Categoric relationship			Team	
			cross-functionalityskills	
			true	false
Team	skills/knowledge sharing	full sharing	H17.1	
		medium sharing	H17.2	
		Minimal or null sharing	H17.3	

• H17.1: Teams characterized by cross-functionality/skills are associated with full skills/knowledge sharing.
• H17.2: Teams characterized by cross-functionality/skills are associated with medium skills/knowledge sharing.
• H17.3: Teams characterized by cross-functionality/skills are associated with minimal or null skills/knowledge sharing.

Pergunta (P18):

Você diria que os times Enabler (Platform) é caracterizado também por ser um time cross-funcional/skills?

P18. CROSS-FUNCTIONALITY/SKILLS IS A PROPERTY OF ENABLER (PLATFORM) TEAM

categoric	Team	
	cross-functionality/skills	
	true	false

Team	Enabler Team	true	H18.1	
		false	H18.3	H18.4

• H18.1: Teams characterized as enabler (platform) teams are associated with being cross-functional.
• H18.3: Teams not characterized as enabler (platform) teams are associated with being cross-functional.
• H18.4: Teams not characterized as enabler (platform) teams are associated with not being cross-functional.

3. The Impact on Team Autonomy

Pergunta (P24):

Podemos afirmar que em uma estrutura de times que possui um Enabler (Platform) team, esses time promove self-organization e autonomia aos times de produto por exemplo?

P24. ENABLER (PLATFORM) TEAM ENABLES TEAM SELF-ORGANIZATION & AUTONOMY

Categoric relationship			Team	
			Autonomy	
			self organization	dependent
Team	Horizontal Enabler	true	H24.1	
		false		

• H24.1: Teams supported by an enabler (platform) team are associated with self-organization and autonomy.

Pergunta (P29):

Você acredita que prover **Serviços de Plataforma** (ou a Equipe de Plataforma) impacta a **carga cognitiva** das equipes de desenvolvimento ou produto? Ela ajuda a reduzir?

P29. Platform Servicing reduce team cognitive load

categoric	Platform
	Platform Servicing

			True	False
team	Cognitive Load	hight	H29.1	
		median/ normal	H29.2	
		low	H29.3	

- H29.1: The presençe of a platform servicing is associated with high cognitive load.
- H29.2: The presençe of a platform servicing is associated with median/normal cognitive load.
- H29.3: The presençe of a platform servicing is associated with low cognitive load.

Pergunta (P31):

Ainda no tema das equipes mais “avançadas” do DevOps. Elas ajudam a reduzir a carga cognitiva das equipes?

P31. Horizontal Teams (Enabler and Bridge) reduce Product team cognitive load

categoric			Team	
			Horizontal Team (Bridge and/or Enabler)	
			True	False
Product team	Cognitive Load	hight	H31.1	
		median	H31.2	
		low	H31.3	

H31.1: The presence of Horizontal Teams is associated with a high Product_team cognitive load.
H31.2: The presence of Horizontal Teams is associated with a median Product_team cognitive load.
H31.3: The presence of Horizontal Teams is associated with a low Product_team cognitive load.

4. Hipoteses refutadas

H2.2	Teams with eventual collaboration are associated with fewer organizational silos.

H2.4	Teams with low-quality collaboration are associated with fewer organizational silos.

H4.10	A team culture based on full knowledge sharing are associated with low quality collaboration between team members

H9.2	Teams not characterized by cross-functionality/skills are associated with full responsibility/ownership sharing.

H10.1	Teams characterized by full responsibility/ownership sharing are associated with organizational silos.

Fase 3: Propostas de refinamento da teoria

3.1 Questões levantadas na instanciação do Leo e paulo:

Das instâncias que o Leo conseguiu, algumas ainda ficaram faltando. Apresento elas aqui para lembrar de discutir: Testing D2

Siloed departments

lack of proper test automation		The model did not fit. Obs: Team could point to Culture_Values_Best_Practices, and then we could say

		about a team with Practices without continuous_testing.

A proposta é ajustar a teoria para que tenha uma conexão entre times e cultute_values_best_practices, principalmente porque as práticas descritas também podem ser adotadas pelos times, como continuous testing.

Collaborating departments:

- Not enough to achieve high delivery performance (nossa teoria nao mostra e discute sobre delivery peformance, talvez seja interessante adicionar. Onde o Leo adicionaria isso?)

infra as development collaborator	Team.responsibility/ownership_sharing = full sharing	But it would be more precise with an additional optional "method" for Operation_team: contributes_to_the_product (or something like this)

Platform teams

Product teams are fully accountable for the non-functional requirements of their services	Product_Team with access_to_all_necessary_information()	But this is just part. We could have something as Product_Team with operations_responsibility
The platform itself handles many NFR concerns	Platform with continuous_monitoring and recovery_automation()	But not so precise. Maybe it could be something like Platform cooping_with_nfr

The infrastructure team is no longer requested for operational tasks	The Bridge_Team automates_infrastructure_creation/deployment/management.	But it do not create/deploy/manage_infrastructure as the Operation_Team would do.
avoids the need for product teams to have infrastructure specialists	Procut_Team with cross-functionality/skills = false	**here I would prefer to say "low" rather than "false"**

[organizations with] platform team structure are high performers		Nothing about delivery performance in the diagram
platform teams not suitable for small companies		Nothing about organization size in the model.

3.2 Questões e refinamentos identificados na rodada de teste anterior (Manoel + 2 specialists)

Team cognitive load limits cross-functionality

Um ponto crucial sobre a **carga cognitiva** e como ela pode sabotar a **multifuncionalidade (cross-functionality)** de uma equipe. A ideia de ter um especialista para cada área em um mesmo time pode levar a um anti-padrão, onde cada um opera em um silo, mesmo estando no mesmo grupo.

Due to the team's cognitive load, it's not advisable to have a fully cross-functional and 100% independent team.

In such a setup, each member will end up focusing on their own area because they will be overloaded with tasks and improvements to make. This ultimately hinders the proper evolution of both the project and the team.

Paes interview: "Ellos dicen ok, para ser cross-functional tenemos en el equipo un diseñador, un frontend, un backend, un tester, PO, etc. y al final necesitas un equipo de 15 personas y no tienen responsabilidad compartida porque cada uno está en su silo dentro de equipo. Esa idea de cross functional puede llevar a este entendimiento que es un poco *naive*. Es mejor mantener un equipo pequeño e ir a buscar cross-funcionalidad con más responsabilidad compartida que es típicamente el modelo T shape (cada persona tiene una especialidad, pero pueden hacer varias cosas distintas)."

AUTOMATION brings from experience

A automação surge da capacidade de identificar e atuar em processos repetitivos, uma sensibilidade cultivada pela experiência direta e prática. A experiência adquirida em diferentes contextos introduz soluções inovadoras que aumentam a eficiência e abordam preventivamente potenciais problemas, melhorando assim a qualidade geral do sistema e otimizando os fluxos de trabalho.

O especialista menciona que a automação surge da experiência prática e da sensibilidade em notar repetições. Além da experiência técnica, quais outros tipos de diversidade (de perfil, de cultura, etc.) você acredita que influenciam a capacidade de uma equipe de identificar oportunidades de automação?

O especialista defende que a automação deve ser aplicada o mais cedo possível no ciclo de desenvolvimento (por exemplo, no ambiente local do desenvolvedor). **Como a adoção de automações precoces pode afetar a cultura de responsabilidade e propriedade (ownership) dos desenvolvedores sobre o seu próprio trabalho?** Como lidar e incentivar essa cultura de automações? **Ela é realmente importante para os contextos do DevOps?**

Quais seriam as principais barreiras que impedem as equipes de perceberem e implementarem automações? Seria responsibility/ownership sharing de autonomia?

Specialist 1: "Because automation comes from experience, right? People do it, there are many people who do something repeatedly without realizing that it is repeated. So the opportunity for automation comes from experience, right? I saw something like this somewhere else, I can do it here, I've already done something similar and in another scenario, I'll do it here. Or even the sensitivity of realizing that this is being very manual because it's taking a lot of time. And I realize that I'm repeating the code, I'm repeating commands, how can I automate? I think all of this is experience. Both life experience and diversity, right? Even if we're talking about people who automate things. And another person who automates in the backend, for example, in this team I'm on, there was a foreign collaborator, he was Hungarian, he created a hook in Git that for each commit, before finishing the commit he runs a Bash script, in Git you can do this. There's a page there about hooks where you can configure the hooks, what did he do? Did he create the hook to run lint before committing? So it's a file there that runs lint on the frontend and runs lint on the backend if it passes, if it doesn't pass, it prints the error there and then the developer has to go there, only commit if it passes lint, right, man, I think that's phenomenal. I didn't know, I didn't even know about this Git thing and I hadn't even had this idea of, Oh, you can put these lints inside, that is, automation in the local environment, that opened up a whole other bunch of things for me. Damn, I can use the local developer computer, because we have the local environment, there's a test environment. The repository and there's production, anyway, there are several moments that we can automate at various moments as soon as possible and that reminds me, right? Other theories. Other principles. That the sooner you identify the problem. That is, the faster the cycle, right, of making a change, identifying a problem and correcting it, the better, then damn. I thought, the best scenario is that everything that is local is faster. Everything I can do to avoid the person's location because if they only commit to lint in the repository, it's better to receive the email later. They're focused on something else, they've already missed the boat. Just giving the example of multidisciplinarity, how much his experience alone has already contributed to teaching me a lot of other things, and I think that's all I need to do."

DevOps (bridge) team

Q: DevOps [bridge] team' atua como intermediário entre desenvolvedores e a equipe de operações de TI. Na sua experiência, esse modelo, por natureza, cria um novo silo, ou ele realmente quebra os silos originais? Ele pode ser uma solução de longo prazo ou é mais adequado para ser uma fase de transição?

Q: O texto menciona que as equipes de DevOps são clientes da equipe de Operações de TI e prestadoras de serviços para os desenvolvedores. Essa dupla função pode causar um conflito de interesses, ou ela capacita a equipe de DevOps a impulsionar as mudanças necessárias em ambos os departamentos?

“Developers provide business solution, leaving the creation, deployment, and management of both the cloud infrastructure, virtual systems, and deployment pipelines to the DevOps teams. These teams are provided services of on-premises infrastructure by the Ops team. **Essentially, the DevOps teams are customers to the Ops team, and service providers to developers.** It is important to note that in this approach to DevOps, everyone is responsible for their actions”

DevOps teams in these organisations understand the codes of developers and help out where necessary. Developers are also made aware of how the automated infrastructure works, though not directly involved in its creation or maintenance. According to some practitioners, a level of confidence is brought about by the basic understanding of other aspects of the process and familiarity with the other actors. Intra-team collaboration is reported as brainstorming and coding together when issues are encountered. Collaboration in FinCo2 involves the DevOps team creating users’ stories from requirements, breaking them into manageable tasks and delegating these tasks to developers through Azure DevOps.

Some of the interviewees had the job title of ‘DevOps Engineer’ and worked in distinct DevOps teams or departments. “We don’t actually have developers in our team. So, in our case. . . it’s just DevOps” [Finco1 DOps1]. They further described their team as “platform builders” for developers, “who support them and host their applications on our platform.” Here, we see DevOps being presented as a job description, with DevOps Engineers responsible for carrying out “DevOps functions”,

Fig. 2c depicts the Developers-DevOps mode where DevOps teams creates, deploys, and manages both the cloud infrastructure and deployment pipelines. Developers applications are also deployed and maintained by the DevOps team.

Here, developers are not responsible for application deployment and management. Completed applications or features are handed over to the DevOps teams for deployment and management, who are the DevOps practices facilitators.

DevOps bridge team mode was the mode widely used in our study. This mode (shown in Fig. 2d) was found in hybrid environment of cloud and onpremises deployment. Here, DevOps teams interface with both developers and IT Ops to drive the practices of DevOps like conffguration management, continuous integration and continuous delivery, automated testing, deployment, monitoring, and metrics collection.

Developers provide business solution, leaving the creation, deployment, and management of both the cloud infrastructure, virtual systems, and deployment pipelines to the DevOps teams. These teams are provided services of on-premises infrastructure by the Ops team. Essentially, the DevOps teams are customers to the Ops team, and service providers to developers. It is important to note that in this approach to DevOps, everyone is responsible for their actions.

Using automation tools, DevOps engineers create pipelines to enable continuous practices such as continuous integration/continuous deployment, continuous testing etc.

The DevOps teams under study are tasked with migration from existing platforms to either cloud based or an automated on-premises environment, and its subsequent maintenance. Generally, they act as intermediary between IT Ops and developers, providing the means to an end in software development (SD) by creating automated pipelines on both physical and virtualized servers to enable continuous integration and continuous delivery.

our ffndings show DevOps being described by practitioners in the study as not just a culture and speciffc job description, but also distinct teams separate from both developers and IT Ops teams. Although members of these teams have backgrounds in either software development or IT Ops, the nomenclature “DevOps” now separates them from their original silos and classiffes them as a unique team of “platform builders”

Bridge team: create a separate DevOps team that functions as a bridge between Devs and Ops