Universidade Federal do Paraná
Programa de Pós-Graduação Lato Sensu
Engenharia Ágil de Projetos

Instituto de Soluções Tecnológicas Aplicadas

ALEX SANTAANA DA SILVA
EUGÊNIO LUÍS MATHEDI
FÁBIO FURQUIM DA SILVA
MATEUS FAVORETTO MACHADO
RAUL DIAS NUNES PEREIRA

ANÁLISE DE IMPLEMENTAÇÃO DE METODOLOGIA ALM: ESTUDO DE CASO DA *STARTUP* VOEARES

CURITIBA
2024

ALEX SANTAANA DA SILVA
EUGENIO LUÍS MATHEDI
FABIO FURQUIM DA SILVA
MATEUS FAVORETTO MACHADO
RAUL DIAS NUNES PEREIRA

ANÁLISE DE IMPLEMENTAÇÃO DE METODOLOGIA ALM: ESTUDO DE CASO DA *STARTUP* VOEARES

Monografia apresentada como requisito parcial à obtenção do grau de Especialista em Engenharia Ágil de Projetos, Curso de Pós-graduação Lato Sensu, Setor de Tecnologia, Departamento de Engenharia Mecânica, Universidade Federal do Paraná.

Orientador: Prof. Dr. Fernando Deschamps e Prof. Dr. Alessandro Marques

CURITIBA
2024

LISTA DE SIGLAS

ALM	Gerenciamento do Ciclo de Vida de Aplicações
SLA	Acordo de Nível de Serviço
XP	Programação Extrema
ITIL	Biblioteca de Infraestrutura de Tecnologia da Informação
GLPI	Gerenciador Livre de Parque Informático
VUCA	Volátil, Incerto, Complexo e Ambíguo
JV	Empreendimento Conjunto
BPR	Reengenharia de Processos de Negócios
PO	Dono do Produto
SM	Mestre Scrum
WIP	Trabalho em Andamento

RESUMO

As *startup*s são conhecidas por sua agilidade e flexibilidade na adaptação às mudanças do mercado e às necessidades dos clientes. No entanto, gerenciar processos de desenvolvimento de *software* em empresas em estágio inicial pode ser um desafio, especialmente quando se trata de garantir qualidade, eficiência e colaboração entre os membros da equipe.

É aí que entra o ALM (*Application Lifecycle Management*, gerenciamento do ciclo de vida do aplicativo). ALM é um conjunto de práticas, ferramentas e metodologias que ajudam as organizações a gerenciar todo o ciclo de vida de desenvolvimento de *software*, desde a coleta de requisitos até a implantação e manutenção.

Nesse contexto, o ALM pode ser um divisor de águas para as *startup*s, permitindo que elas agilizem seus processos de desenvolvimento, reduzam custos e melhorem a qualidade do produto. Por enfrentarem desafios únicos quando se trata de desenvolvimento de *software*, geralmente possuem recursos limitados, prazos apertados e um grau de incerteza em relação à demanda do mercado e ao *feedback* dos clientes.

Ademais, o ALM pode proporcionar diversos benefícios, tais como: permitir uma maior colaboração entre os times, reduzir erros aumentando a qualidade final, melhorar o tempo de resposta ao cliente, reduzir custos de desenvolvimento e facilitar a escalabilidade.

No contexto de nosso estudo de caso, a *Startup* VOEARES apresentou grande deficiência no controle e gerenciamento da entrada de *features* de suporte e de novos desenvolvimentos. A implementação do processo se mostrou bastante promissor e trouxe diversos benefícios, como a redução da SLA de 30 dias para 24 horas úteis.

Desta forma o ALM pode ajudar as *startup*s a acelerar seus ciclos de desenvolvimento, automatizando tarefas repetitivas, reduzindo erros manuais e fornecendo *feedback* em tempo real. Isso pode ajudá-los a lançar produtos mais rapidamente e responder de forma mais assertiva às mudanças do mercado, se tornando um ativo valioso para *startup*s que buscam melhorar seus processos de desenvolvimento de *software*.

Palavras-chave: ALM, *Startup*, Eficiência operacional, Desenvolvimento de *software* e Colaboração.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

CONTEÚDO

1. INTRODUÇÃO E SITUAÇÃO-PROBLEMA

1.1. CONTEXTUALIZAÇÃO

O desenvolvimento de *software* desempenha um papel crucial no cenário empresarial contemporâneo. *Startups*, em particular, são reconhecidas por sua agilidade e capacidade de adaptação às mudanças rápidas do mercado e às demandas dos clientes. No entanto, gerenciar processos de desenvolvimento de *software* em *startup*s pode ser desafiador. É necessário conciliar a agilidade e a flexibilidade com a necessidade de qualidade, eficiência e colaboração eficaz entre os membros da equipe.

O ALM surgiu como uma abordagem abrangente que oferece práticas, ferramentas e metodologias para gerenciar todo o ciclo de vida do desenvolvimento de *software*, desde a coleta de requisitos até a implantação e manutenção. Esta abordagem busca integrar as diversas fases do desenvolvimento de *software*, proporcionando visibilidade, controle e rastreabilidade em todos os estágios do processo.

Em particular, o ALM pode ser um divisor de águas para *startup*s, permitindo que otimizem seus processos de desenvolvimento, reduzam custos, melhorem a qualidade do produto e respondam de forma ágil às mudanças do mercado. As *startups* enfrentam desafios únicos, muitas vezes operando com recursos limitados, prazos apertados e alto grau de incerteza em relação à demanda do mercado e ao *feedback* dos clientes.

1.2. SITUAÇÃO-PROBLEMA

Neste contexto, surge uma situação-problema que motiva este estudo. Muitas *startups* enfrentam dificuldades no controle e gerenciamento eficiente do ciclo de vida do desenvolvimento de *software*. Essas dificuldades se manifestam em várias formas, desde a falta de visibilidade sobre o progresso do desenvolvimento até a ineficiência na colaboração entre equipes multidisciplinares. Uma área crítica que muitas *startup*s identificam como problemática é o processo de entrada de novas funcionalidades de suporte e de novos desenvolvimentos. A situação atual muitas vezes resulta em atrasos, qualidade inconsistente e insatisfação dos clientes.

Esta situação-problema é central para este estudo, pois descreve um desafio específico que as *startup*s enfrentam em suas operações de desenvolvimento de *software*. Abordar efetivamente essa situação-problema requer uma abordagem abrangente que integre o ALM em todas as fases do ciclo de vida do aplicativo.

1.3. JUSTIFICATIVA

A justificativa para este estudo é respaldada pela necessidade de oferecer soluções eficazes para a situação-problema identificada. O ALM oferece uma abordagem que pode contribuir significativamente para a melhoria dos processos de desenvolvimento de *software* em *startup*s.

A implantação bem-sucedida do ALM nas *startup*s pode proporcionar uma série de benefícios, incluindo uma maior colaboração entre equipes, redução de erros, aumento na qualidade final do produto, melhoria no tempo de resposta ao cliente, redução de custos de desenvolvimento e facilidade de escalabilidade. Esta justificativa se baseia em evidências de que o ALM pode ser um ativo valioso para *startup*s que desejam melhorar seus processos de desenvolvimento de *software* e competir eficazmente em seus mercados-alvo.

1.4. OBJETIVOS

Neste estudo, definimos objetivos claros para direcionar nossa pesquisa e a implementação do ALM nas *startup*s. Os objetivos incluem:

1.4.1. Objetivo Geral

Analisar e demonstrar como a aplicação do ALM nas diferentes fases do ciclo de vida do desenvolvimento de *software* pode otimizar os processos de *startup*s, melhorando a eficiência, a qualidade e a capacidade de resposta às mudanças do mercado.

1.4.2. Objetivos Específicos

- Identificar as práticas, ferramentas e metodologias de ALM apropriadas para cada fase do ciclo de vida do aplicativo.
- Avaliar os benefícios da implementação do ALM em *startup*s, incluindo a redução de erros, aumento da colaboração e eficiência operacional.
- Documentar um estudo de caso de uma *startup* que implementou o ALM e destacar os desafios, soluções e resultados obtidos.

Este conjunto de objetivos orientará o desenvolvimento deste trabalho, fornecendo um caminho claro para atingir os resultados desejados.

Este capítulo de introdução e situação-problema estabelece a base para a pesquisa e o estudo que se seguirão. Ele destaca a relevância do ALM nas *startup*s, identifica a situação-problema que motiva este estudo, fornece justificativa para a pesquisa e estabelece objetivos específicos a serem alcançados. Nos capítulos subsequentes, a fundamentação teórica, a proposta de solução, os resultados e as conclusões aprofundaram ainda mais esse tema e sua aplicação nas *startup*s.

2. FUNDAMENTAÇÃO PARA A SOLUÇÃO

O propósito deste capítulo é apresentar conceitos essenciais para a compreensão aprofundada da pesquisa, destacando temas fundamentais e suas ramificações específicas que influenciaram diretamente nos resultados obtidos.

2.1. O *APPLICATION LIFECYCLE MANAGEMENT* (ALM)

Neste capítulo, a fundamentação teórica gira em torno do ALM. Exploraremos em detalhes o que é e como ele se tornou uma abordagem essencial no campo do desenvolvimento de *software*.

O ALM compreende um conjunto de práticas, ferramentas e metodologias que visam integrar e gerenciar todas as fases do ciclo de vida do desenvolvimento de *software*. Isso inclui desde a coleta de requisitos até a implantação, manutenção e melhoria contínua do aplicativo. Aiello e Sachs (2011) destacam a importância do ALM

como uma abordagem abrangente para modernizar o desenvolvimento de *software*, enfatizando a coordenação eficiente de todas as fases do ciclo de vida do aplicativo.

Para Mantle & Lichty (2012), o gerenciamento do ciclo de vida de aplicações (ALM) é um framework para gerenciar requisitos, arquitetura, desenvolvimento, teste, implantação e manutenção contínua de aplicações de software.

Analisaremos como o ALM proporciona visibilidade, controle e rastreabilidade em todo o processo de desenvolvimento, tornando-o particularmente valioso para *startup*s que buscam otimizar suas operações.

2.1.1. Gestão de requisitos

A gestão de requisitos é uma parte crucial do processo de ALM. Ela envolve a identificação, documentação, análise, verificação e rastreamento dos requisitos ao longo de todo o ciclo de vida do *software*. Uma gestão eficaz de requisitos contribui significativamente para o sucesso do projeto, assegurando que as necessidades do cliente sejam compreendidas e atendidas de maneira consistente. A seguir estão alguns aspectos relevantes da gestão de requisitos no contexto do ALM.

Para Molyneaux (2009), uma abordagem sistemática e bem gerenciada da gestão de requisitos no contexto do ALM é essencial para o desenvolvimento bem-sucedido de *software*, garantindo que as soluções entregues atendam às necessidades dos usuários e aos objetivos do negócio.

Identificação de requisitos

Inicialmente, é essencial identificar e capturar todos os requisitos relevantes do sistema. Isso pode envolver a interação com os *stakeholders*, como clientes, usuários finais e partes interessadas, para garantir uma compreensão abrangente das expectativas.

Documentação adequada

Os requisitos devem ser documentados de forma clara e compreensível. Isso inclui especificações detalhadas, casos de uso, diagramas e outros artefatos que ajudem a transmitir as necessidades e funcionalidades desejadas

Análise e priorização

A análise dos requisitos envolve a avaliação da viabilidade técnica, impacto no projeto e potencial retorno sobre o investimento. A priorização é crucial para determinar quais requisitos devem ser implementados primeiro, especialmente em projetos ágeis.

Rastreamento ao longo do ciclo de vida

Os requisitos devem ser rastreados desde a concepção até a implementação e manutenção. Isso ajuda a garantir que cada requisito seja considerado em todas as fases do desenvolvimento e que qualquer alteração seja gerenciada de forma controlada.

Comunicação efetiva com as partes interessadas

A gestão de requisitos exige uma comunicação efetiva com todas as partes interessadas. Isso inclui *feedback* constante para garantir que as expectativas dos *stakeholders* estejam sendo atendidas e que as mudanças nos requisitos sejam compreendidas por todos.

Gestão de mudanças

À medida que o projeto avança, é comum que os requisitos evoluam. A gestão de mudanças é vital para lidar com alterações nos requisitos, garantindo que as atualizações sejam documentadas, avaliadas e implementadas de maneira controlada.

Integração com outros processos do ALM

A gestão de requisitos está intimamente ligada a outras áreas do ALM, como desenvolvimento, teste e implementação. Uma integração eficiente entre esses processos assegura uma abordagem holística e coesa para o ciclo de vida do aplicativo.

2.1.2. Desenvolvimento e integração contínua

Segundo Rossberg (2014), o desenvolvimento e a integração contínua são componentes essenciais do ALM. Esses processos visam promover a entrega contínua e consistente de *software* de alta qualidade, reduzindo erros e acelerando o ciclo de vida do desenvolvimento. Alguns pontos chave relacionados ao desenvolvimento e à integração contínua no contexto serão abordados a seguir.

A abordagem de desenvolvimento e integração contínua no ALM promove a agilidade, a confiabilidade e a eficiência no ciclo de vida do *software*. Ao permitir entregas frequentes e automatizar processos-chave, as equipes conseguem responder rapidamente às mudanças nos requisitos e entregar *software* de alta qualidade de maneira consistente (Rossberg, 2014).

Desenvolvimento contínuo

O desenvolvimento contínuo no ALM refere-se à prática de criar código de maneira iterativa e incremental. Os desenvolvedores trabalham em pequenas unidades de funcionalidades ou melhorias, permitindo uma entrega mais frequente e uma resposta rápida às mudanças nos requisitos.

Integração contínua

A integração contínua é o processo de combinar regularmente o código desenvolvido por diferentes membros da equipe em um repositório compartilhado. Isso é feito automaticamente por meio de ferramentas de integração contínua, garantindo que o código seja integrado e testado frequentemente.

Automação de build e deploy

Ferramentas de automação desempenham um papel crucial no ALM, especialmente durante o desenvolvimento contínuo. A automação de *build* e deploy permite a criação automática do *software* a partir do código-fonte e sua implantação em ambientes de teste ou produção de forma eficiente.

Testes contínuos

A prática de testes contínuos é fundamental no ALM. Isso envolve a execução automatizada de testes unitários, testes de integração e outros tipos de testes a cada integração de código. Testes frequentes garantem a detecção precoce de bugs e a manutenção da qualidade do *software*.

Feedback imediato

A integração contínua fornece *feedback* imediato aos desenvolvedores. Se uma alteração no código quebrar alguma funcionalidade existente, os testes automatizados identificaram isso rapidamente, permitindo correções imediatas. Isso contribui para a estabilidade do código-base.

Gestão de versões e controle de configuração

A integração contínua requer um controle eficaz de versões e configurações. Ferramentas de controle de versão, como Git, permitem o gerenciamento eficiente de alterações no código-fonte, garantindo a rastreabilidade e a reversão de mudanças indesejadas.

Ambientes padronizados

A integração contínua preconiza ambientes padronizados para desenvolvimento, teste e produção. Isso minimiza a ocorrência de problemas

relacionados a diferenças de configuração entre ambientes e facilita a replicação do ambiente de produção.

2.1.3. Testes e qualidade

A área de testes e garantia de qualidade no contexto do ALM desempenha um papel crucial na entrega de *software* de alta qualidade. Esses processos visam identificar e corrigir defeitos, garantir a conformidade com requisitos e melhorar a confiabilidade do *software*.

A abordagem de testes e qualidade no ALM visa garantir que o *software* atenda aos requisitos, seja robusto e esteja livre de defeitos. Ao integrar testes de forma contínua ao longo do ciclo de vida, as equipes conseguem entregar produtos de alta qualidade de maneira consistente e eficiente.

A seguir estão alguns pontos chave relacionados aos testes e à qualidade no ALM.

Estratégia de teste abrangentes

O ALM promove estratégias de teste abrangentes que vão além dos testes unitários. Isso inclui testes de integração, testes de sistema, testes de aceitação do usuário e outros tipos de testes que abordam diferentes níveis e aspectos do *software*.

Testes automatizados

A automação de testes é uma prática fundamental no ALM. Testes automatizados garantem uma execução consistente e repetitiva, acelerando o processo de teste e permitindo uma resposta rápida às mudanças no código.

Testes contínuos

Assim como na fase de desenvolvimento, o ALM promove a prática de testes contínuos. Testes automatizados são executados a cada nova integração de código, proporcionando *feedback* imediato sobre possíveis problemas e mantendo a qualidade do *software* durante todo o ciclo de vida.

Gestão de defeitos

O ALM inclui uma abordagem sistemática para a gestão de defeitos. Cada defeito identificado durante os testes é documentado, rastreado e priorizado. Isso permite uma alocação eficiente de recursos para corrigir os defeitos mais críticos.

Rastreamento de requisitos

Os testes no ALM são alinhados aos requisitos do sistema. Cada teste é mapeado para um requisito específico, garantindo que todos os requisitos sejam testados e que a conformidade seja verificada.

Métricas de qualidade

O ALM utiliza métricas de qualidade para avaliar o desempenho do *software*. Isso inclui métricas de cobertura de teste, taxa de defeitos encontrados e corrigidos, tempo médio para resolver defeitos, entre outras métricas relevantes.

Padrões de codificação

Além dos testes automatizados, o ALM incentiva a adoção de padrões de codificação e a realização de revisões de código. Essas práticas contribuem para a prevenção de defeitos desde as fases iniciais do desenvolvimento.

Ambientes de teste controlados

O ALM preconiza ambientes de teste controlados e reproduzíveis. Isso garante que os testes sejam realizados em condições consistentes e que os resultados sejam confiáveis.

2.1.4. Implantação e monitoramento

A fase de implantação e monitoramento no contexto do ALM é crítica para garantir que o *software* seja entregue com sucesso aos usuários finais e que seu desempenho e integridade sejam continuamente avaliados.

A combinação eficiente de implantação e monitoramento no ALM assegura que o *software* seja entregue de maneira confiável e que sua operação seja sustentável ao longo do tempo. Ao priorizar a automação, a integração e a visibilidade contínua, as equipes conseguem manter a estabilidade do sistema e responder rapidamente a eventos adversos. Aqui estão alguns pontos chave relacionados à implantação e monitoramento no ALM.

Planejamento da implantação

Antes da implantação, é necessário um planejamento detalhado que inclua a seleção de ambientes de implantação, a definição de procedimentos e a consideração de possíveis contingências. O ALM facilita a coordenação desses esforços, garantindo uma transição suave para a produção.

Implantação contínua

O ALM promove a prática de implantação contínua, em que as atualizações de *software* são implementadas de forma incremental e frequente. Isso ajuda a reduzir o tempo de inatividade do sistema e a fornecer novas funcionalidades de maneira ágil.

Automação da implantação

Ferramentas de automação desempenham um papel crucial na implantação eficiente. O ALM integra ferramentas que automatizam a distribuição de versões, a configuração de ambientes e a aplicação de scripts de banco de dados, garantindo consistência e prevenindo erros humanos.

Gestão de configuração e versões

A implantação no ALM está integrada à gestão de configuração e versões. Isso significa que é possível rastrear quais versões específicas foram implantadas em ambientes de teste e produção, facilitando a identificação e correção de problemas.

Monitoramento contínuo

O ALM enfatiza o monitoramento contínuo do desempenho do sistema e a coleta de métricas relevantes. Isso permite identificar rapidamente possíveis problemas, otimizar o desempenho e garantir uma experiência eficiente para os usuários finais.

***Logs* e auditoria**

O ALM promove a geração e análise de logs detalhados durante a implantação e operação. Esses logs são valiosos para auditorias, resolução de problemas e compreensão do comportamento do sistema em diferentes circunstâncias.

***Rollback* eficiente**

O ALM facilita a implementação de estratégias de *rollback* eficientes em caso de problemas durante a implantação. Isso é fundamental para minimizar impactos quando uma versão nova é colocada em produção.

Integração com ferramentas de monitoramento

A integração com ferramentas de monitoramento externas é comum no ALM. Isso permite a utilização de soluções especializadas para monitorar a saúde do sistema, detectar anomalias e notificar a equipe de operações sobre possíveis problemas.

2.1.5. Gestão de mudanças e configuração

A gestão de mudanças e configuração desempenha um papel fundamental no ALM, garantindo que as alterações no *software* sejam controladas, rastreadas e gerenciadas de maneira eficaz para a estabilidade e a confiabilidade do *software*. Ao adotar boas práticas nessa área, as equipes podem gerenciar eficazmente a evolução do *software*, garantindo que as mudanças sejam controladas, auditadas e implementadas de maneira segura.

A seguir estão alguns aspectos importantes relacionados à gestão de mudanças e configuração no contexto do ALM.

Controle de versões

A gestão de mudanças no ALM inclui o controle de versões do código-fonte e de outros artefatos do projeto. Isso possibilita rastrear as alterações ao longo do tempo, facilita a colaboração entre membros da equipe e permite a recuperação de versões anteriores quando necessário.

Branching e merging

O ALM suporta estratégias avançadas de *branching* e *merging*, permitindo que equipes trabalhem em paralelo em diferentes funcionalidades ou correções de bugs. Essa capacidade é crucial para evitar conflitos e garantir a integridade do código durante o desenvolvimento.

Auditoria e rastreamento

A gestão de mudanças no ALM inclui recursos de auditoria e rastreamento, garantindo que cada alteração seja documentada. Isso é essencial para atender a requisitos regulatórios, entender o histórico do projeto e facilitar a resolução de problemas.

Solicitações de mudanças (*change requests*)

O ALM facilita a criação e o gerenciamento de solicitações de mudança, permitindo que as alterações sejam propostas, revisadas e aprovadas antes de serem implementadas. Isso ajuda a garantir que as mudanças estejam alinhadas aos objetivos do projeto e do negócio.

Impacto e análise de riscos

Antes da implementação de uma mudança, o ALM suporta a análise de impacto e risco. Isso envolve a avaliação de como a alteração afetará outros componentes do sistema e a identificação de possíveis consequências não desejadas.

Processos de aprovação

As mudanças no ALM geralmente passam por processos de aprovação. Isso inclui revisões de código, aprovação de solicitações de mudança e validação de testes. Processos estruturados de aprovação ajudam a garantir a qualidade e a consistência das mudanças implementadas.

Configuração de ambientes

A gestão de configuração no ALM abrange a configuração de ambientes de desenvolvimento, teste e produção. Isso assegura que todos os ambientes estejam alinhados e que as mudanças sejam implementadas de maneira consistente em todos os estágios do ciclo de vida do aplicativo.

Automatização de processos

Ferramentas de ALM frequentemente oferecem recursos de automação para facilitar a implementação de mudanças e garantir que os processos de configuração sejam executados de maneira eficiente e sem erros.

2.1.6. Colaboração e comunicação

A colaboração e comunicação eficazes são pilares fundamentais no ALM. Esses aspectos são cruciais para garantir que as equipes trabalhem de maneira coordenada, compartilhem informações relevantes e alcancem os objetivos do projeto. No contexto do ALM, onde várias equipes e stakeholders estão envolvidos em diferentes fases do ciclo de vida do aplicativo, a adoção de práticas que promovam a comunicação transparente e a colaboração é fundamental para garantir a entrega bem-sucedida de *software* de alta qualidade.

A seguir estão alguns pontos chave relacionados à colaboração e comunicação no ALM.

Plataformas colaborativas

Ferramentas de ALM frequentemente incluem plataformas colaborativas que permitem que membros da equipe compartilhem documentos, comentem código, atribuam tarefas e colaborem em tempo real. Isso promove a transparência e a comunicação eficaz entre os membros da equipe.

Reuniões de compartilhamento de informações

O ALM facilita a organização de reuniões e a troca de informações entre diferentes partes interessadas, incluindo desenvolvedores, testadores, gerentes de

projeto e clientes. Isso é essencial para garantir que todos estejam alinhados em relação aos objetivos e ao progresso do projeto.

Rastreamento de atividades

Ferramentas de ALM incluem recursos de rastreamento de atividades que permitem aos membros da equipe acompanhar o progresso das tarefas, identificar possíveis gargalos e colaborar de maneira mais eficiente para superar desafios.

Notificações e alertas

O ALM fornece funcionalidades de notificação e alertas, garantindo que os membros da equipe sejam informados sobre alterações relevantes, aprovações pendentes, e outros eventos importantes no ciclo de vida do aplicativo.

Integração com ferramentas de comunicação

Ferramentas de ALM muitas vezes oferecem integração com ferramentas de comunicação populares, como Slack, Microsoft Teams ou outras plataformas de mensagens. Isso permite uma comunicação contínua e em tempo real, especialmente em equipes distribuídas geograficamente.

Documentação colaborativa

O ALM promove a criação de documentação colaborativa, facilitando a compartilhamento de conhecimento e boas práticas entre os membros da equipe. Documentação clara e acessível é essencial para garantir a compreensão do código, processos e decisões de design.

Gestão de conflitos

Em ambientes de desenvolvimento de *software*, é comum surgirem conflitos. O ALM fornece ferramentas para gerenciar e resolver conflitos de maneira eficaz, garantindo que as divergências sejam tratadas de forma construtiva.

Acesso controlado à informação

O acesso controlado à informação no ALM garante que os membros da equipe tenham acesso às informações relevantes para suas funções, mantendo a segurança e a confidencialidade quando necessário.

2.2. PRÁTICAS ÁGEIS E METODOLOGIAS

O ALM envolve uma série de práticas ágeis e metodologias que visam melhorar a eficiência e a eficácia do desenvolvimento de *software*. Algumas práticas ágeis e metodologias comuns relacionadas ao ALM serão abordadas a seguir.

A escolha da abordagem depende das necessidades específicas do projeto, da equipe e do contexto organizacional. Muitas organizações também adaptam e combinam elementos de diferentes metodologias para atender às suas necessidades únicas.

Um componente fundamental do ALM são as práticas ágeis e metodologias associadas, como *Scrum* e *Extreme Programming* (XP).

> Sutherland, J. (2014), Scrum é como um jogo de xadrez: é fácil de entender, mas difícil de dominar. Todo mundo pode aprender as regras básicas, mas se você quer se tornar um grande jogador, precisa se dedicar constantemente ao aprimoramento de suas habilidades.

Essas abordagens enfatizam a colaboração, a flexibilidade e a capacidade de adaptação às mudanças.

> Beck, K. (2002), ao invés de se concentrar em encontrar maneiras de evitar mudanças, deveríamos nos concentrar em construir sistemas que tolerem mudanças - mesmo mudanças radicais.

Vamos explorar como as práticas ágeis se encaixam no ALM e como podem ser aplicadas em *startups*.

2.2.1. *Scrum*: uma abordagem ágil para excelência em desenvolvimento de *software*

O *Scrum*, uma metodologia ágil de gerenciamento de projetos, tem revolucionado a maneira como equipes desenvolvem produtos, proporcionando agilidade, flexibilidade e eficácia. Criado por Ken Schwaber e Jeff Sutherland, o *Scrum* é fundamentado em princípios que priorizam a colaboração, adaptação contínua e entrega de valor ao cliente.

> Sutherland, J. (2014), Scrum não é uma metodologia, é um framework. É uma forma de pensar e um conjunto de valores que ajudam equipes a entregar valor em um ambiente complexo.

Princípios fundamentais:

1. Iterativo e Incremental: O *Scrum* opera em ciclos chamados "*Sprint*s", geralmente de 2 a 4 semanas, nos quais incrementos de produto são entregues. Isso permite uma adaptação rápida às mudanças nos requisitos e uma entrega contínua de valor.
2. Papéis Claros e Colaboração: Papéis distintos, como *Scrum Master*, *Product owner* e a Equipe de Desenvolvimento, promovem uma

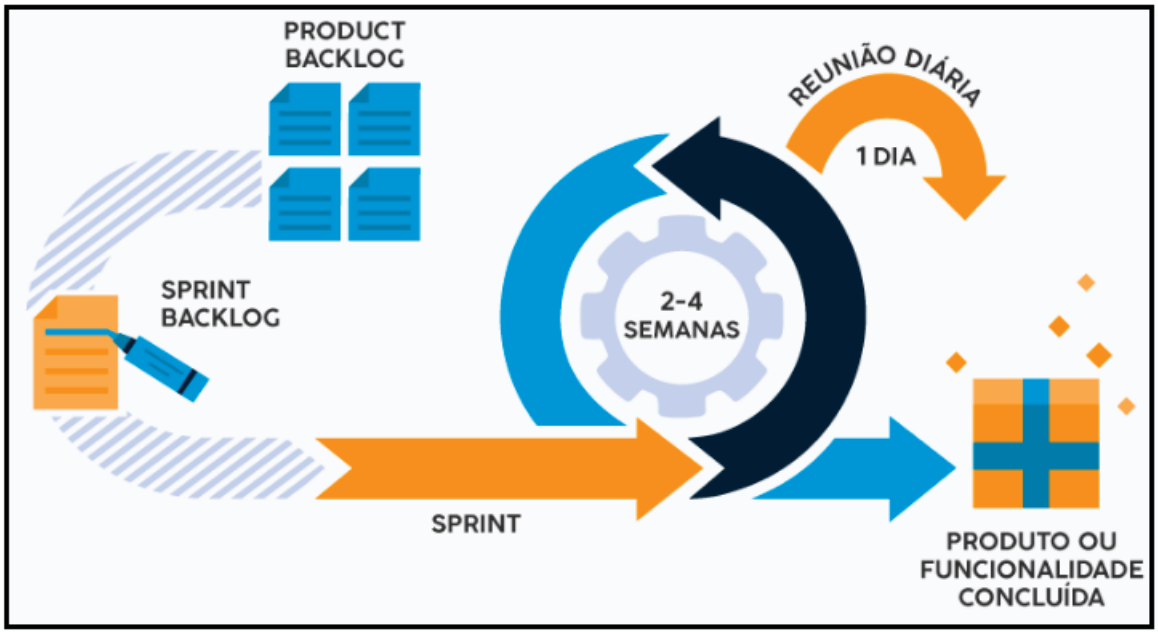
colaboração eficaz. O *Product owner* representa os interesses dos stakeholders, enquanto o *Scrum Master* facilita o processo e remove obstáculos.

3. Transparência e inspeção constante: Artefatos como o *Burndown* Chart e reuniões regulares, como a *Sprint Review* e a *Sprint Retrospective*, garantem transparência e permitem inspeções regulares do progresso do projeto

Na figura 1 podemos identificar o ciclo de vida do *scrum*:

1. *Sprint* Planning: Inicia cada *Sprint*, onde a equipe define as metas da *Sprint* e seleciona as funcionalidades a serem desenvolvidas.
2. Daily *Scrum*: Reuniões diárias curtas para sincronização da equipe, abordando desafios, progresso e planejamento para o dia seguinte.
3. *Sprint Review*: Avaliação do trabalho concluído durante a *Sprint*, proporcionando uma oportunidade para *feedback* e ajustes.
4. *Sprint Retrospective*: Reflexão sobre o processo e identificação de melhorias contínuas.

FIGURA 1 – SCRUM.



FONTE: TECNICON (2024).

Benefícios do *Scrum*:

- Adaptação Rápida: Capacidade de se adaptar a mudanças nos requisitos e no ambiente de negócios de forma rápida e eficiente.

- Foco no Valor: Priorização de funcionalidades com base no valor percebido pelo cliente, garantindo entregas incrementais significativas.
- Colaboração Intensa: Promoção da comunicação contínua entre os membros da equipe, partes interessadas e o *product owner*.
- Melhoria Contínua: As *sprint*s retrospectivas incentivam a reflexão e a implementação de melhorias contínuas no processo.

O *Scrum* oferece uma estrutura ágil eficaz para o desenvolvimento de *software*, destacando-se por sua abordagem colaborativa, iterativa e centrada no valor. Ao adotar o *Scrum*, as equipes podem transformar a complexidade do desenvolvimento em oportunidades de crescimento, entregando produtos de alta qualidade de maneira eficiente e adaptativa.

> Para Schwaber, K. (2002), o Scrum é como um jogo de rugby. É uma estratégia onde um time trabalha junto para ganhar, mesmo que possa parecer que a bola foi jogada em outra direção.

2.2.2. *Extreme programming* (XP)

O *extreme programming* (XP) é uma metodologia ágil de desenvolvimento de *software* que se concentra na produção de código de alta qualidade, adaptação contínua às mudanças nos requisitos e colaboração intensiva entre os membros da equipe. Criado por Kent Beck na década de 1990, o XP introduziu uma série de práticas e valores que revolucionaram a forma como as equipes desenvolvem *software*

Para Beck, K. (2004), *Extreme Programming* é sobre mudança social, e essa é a forma mais difícil de mudança que existe.

Princípios e Práticas fundamentais:

- Desenvolvimento orientado a testes (TDD): O TDD é uma prática central do XP, onde os testes automatizados são escritos antes do código de produção. Isso promove uma maior confiança na estabilidade do código e facilita mudanças futuras.
- Integração contínua: A integração contínua envolve a integração frequente do código desenvolvido pela equipe em um repositório compartilhado, seguida pela execução automatizada de testes. Isso reduz o risco de conflitos e problemas de integração.

- Programação em par: Os desenvolvedores trabalham em pares, colaborando na resolução de problemas e revisão de código. Isso promove a qualidade do código, compartilha conhecimento e reduz o tempo de revisão.
- Refatoração: A refatoração envolve a melhoria contínua do design do código sem alterar o comportamento observável externamente. Isso mantém o código limpo, modular e fácil de entender.
- Design simples: O XP preconiza um design simples e evolutivo, onde as soluções são implementadas para atender aos requisitos atuais, sem antecipar problemas futuros.

> Segundo Martin, R. C. (2002), XP tenta ser o mais leve possível, manter o overhead mínimo e trabalhar com todas as outras práticas que estão disponíveis.

Benefícios do XP:

- Qualidade de código superior: O foco em práticas como TDD, refatoração e design simples resulta em código mais limpo, testável e de alta qualidade.
- Adaptação rápida às mudanças: As práticas do XP, como refatoração e design simples, permitem que as equipes respondam rapidamente a mudanças nos requisitos do cliente.
- Colaboração intensa: A programação em par e a integração contínua promovem uma colaboração intensiva entre os membros da equipe, resultando em um ambiente de trabalho altamente colaborativo.
- Maior satisfação do cliente: O XP prioriza a entrega frequente de incrementos de produto de alta qualidade, resultando em maior satisfação do cliente e melhor adaptação às suas necessidades em constante mudança.

O XP oferece uma abordagem ágil única e altamente eficaz para o desenvolvimento de *software*, enfatizando a qualidade técnica, adaptação contínua e colaboração intensiva. Ao adotar as práticas e valores do XP, as equipes podem alcançar níveis mais altos de eficiência, qualidade e satisfação do cliente em seus projetos de desenvolvimento de *software*. Para Beck, K (2004), XP não é uma solução milagrosa, mas um ciclo de feedback que permite ajustar constantemente o seu rumo."

2.3. FERRAMENTA JIRA

O Jira é uma ferramenta amplamente reconhecida e utilizada para o gerenciamento de projetos ágeis, desenvolvida pela empresa Atlassian. Seu principal objetivo é facilitar a colaboração, o planejamento e a execução de projetos de *software* de forma eficiente. Com uma interface intuitiva e uma variedade de recursos poderosos, o Jira se tornou uma escolha popular para equipes de desenvolvimento em todo o mundo. Abaixo, discutiremos alguns dos principais aspectos do Jira como uma ferramenta de gerenciamento de projetos ágeis.

Possui uma estrutura flexível que pode ser adaptada para atender às necessidades específicas de diferentes equipes e metodologias de desenvolvimento. Seja utilizando *Scrum*, Kanban, XP ou uma abordagem personalizada, o Jira permite configurar fluxos de trabalho, quadros, campos e permissões de acordo com os requisitos do projeto.

Para equipes que seguem o framework *Scrum*, o Jira oferece recursos como quadros de planejamento de *sprint*, gráficos de *burndown* e relatórios de velocidade. Isso permite que as equipes planejem e executem suas iterações de forma eficiente, acompanhando o progresso e ajustando o plano conforme necessário. Sobretudo, as equipes que preferem o método *Kanban*, o Jira oferece quadros visuais que representam o fluxo de trabalho das tarefas. Os membros da equipe podem mover cartões entre as colunas para indicar o progresso, facilitando a visualização do trabalho em andamento e a identificação de possíveis gargalos.

O Jira facilita a gestão do *backlog* do produto, permitindo que as equipes criem, priorizem e refinem as histórias de usuário e as tarefas de forma colaborativa. Os itens do *backlog* podem ser detalhados com descrições, critérios de aceitação, estimativas de esforço e dependências, proporcionando uma visão clara do que precisa ser feito e em que ordem.

Uma das principais funcionalidades do Jira é o rastreamento de problemas e tarefas. Os membros da equipe podem criar, atribuir, comentar e atualizar problemas diretamente na plataforma, mantendo todos os envolvidos atualizados sobre o status e os próximos passos. Além disso, é possível anexar arquivos, vincular problemas relacionados e definir datas de vencimento para garantir que nada seja esquecido. O Jira oferece uma variedade de relatórios e métricas que ajudam as equipes a avaliar o desempenho do projeto e identificar áreas de melhoria. Isso inclui gráficos de

burndown, relatórios de velocidade, análises de fluxo de valor e muito mais. Com essas informações, as equipes podem tomar decisões informadas e ajustar sua abordagem para maximizar a eficiência e a qualidade.

Por fim, essa é uma ferramenta poderosa e versátil que oferece suporte ao gerenciamento de projetos ágeis em todas as suas etapas, desde o planejamento até a entrega. Com sua ampla gama de recursos e flexibilidade, o Jira ajuda as equipes a colaborar de forma mais eficaz, melhorar a transparência, aumentar a produtividade e entregar *software* de alta qualidade dentro do prazo e do orçamento estabelecidos.

O Jira Software é construído para todos os membros da equipe, de desenvolvedores a gerentes. Ele ajuda a planejar, rastrear e gerenciar todos os trabalhos da sua equipe, de maneira clara e eficiente."

2.4. DESAFIOS ÚNICOS DAS *STARTUP*S

O termo "*startup*", originário dos Estados Unidos, pode ser entendido em nosso idioma como o ato de iniciar algo ou dar início a um empreendimento. No entanto, sua definição transcende a tradução literal, tornando-se associado a empresas que experimentam um crescimento rápido a partir de ideias inovadoras, muitas vezes impulsionadas pela tecnologia, embora não exclusivamente. Uma *startup* é caracterizada como um grupo de pessoas ou empresas que buscam desenvolver um modelo de negócio escalável, ou seja, capaz de aumentar sua receita sem a necessidade de aumentar proporcionalmente seus custos, permitindo que suas novas ideias atendam às demandas da sociedade.

> Segundo Eric Ries (2011, p. 26), uma *startup* é uma instituição humana projetada para criar novos produtos e serviços sob condições de extrema incerteza.

Para alcançar esse objetivo, as *startup*s dependem de investidores que desempenham o papel de catalisadores financeiros. Esses investimentos fornecem o impulso necessário para o crescimento acelerado da empresa. No entanto, surgem desafios, como a dinâmica do cenário, pois o sucesso desse crescimento requer um sólido planejamento financeiro e uma estrutura hierárquica bem definida. Esta estrutura é caracterizada por uma série de atributos:

Inovação: Como essas empresas não seguem fórmulas tradicionais de negócios, elas se apoiam em ideias inovadoras para desenvolver produtos ou soluções que abordem desafios e necessidades da sociedade.

Incerteza: Decorrente da entrada em mercados pouco ou nunca explorados, as *startup*s enfrentam riscos consideráveis em comparação com negócios tradicionais.

Escalabilidade: Esse elemento possibilita que as *startup*s aumentem sua produção e distribuição sem aumentar proporcionalmente os custos, permitindo um retorno rápido sobre o investimento inicial.

Rentabilidade: O modelo de negócios pode ser replicado em larga escala para diversos clientes.

Flexibilidade: Caso a ideia inicial não gere os resultados esperados, as *startup*s têm a capacidade de ajustar sua estratégia ou adaptar-se às necessidades dos clientes de forma ágil.

Iniciar uma empresa sempre envolve dois lados: o otimista e o pessimista, especialmente no contexto das *startup*s. Sem um preparo adequado, alcançar o sucesso se torna uma tarefa árdua

> Eric Ries reforça essa situação (2011, p. 10), o sucesso de uma *startup* não é consequência de bons genes ou de estar no lugar certo na hora certa. O sucesso de uma *startup* pode ser construído seguindo o processo correto, que pode ser aprendido, e, portanto, ensinado.

A implementação eficaz do ALM em uma *startup* pode resolver muitos dos desafios comuns, promovendo uma abordagem estruturada e eficiente para o desenvolvimento de *software*, desde a concepção até a entrega e manutenção. Aqui estão algumas maneiras como o ALM pode ajudar a resolver problemas em uma *startup*:

Alinhamento Estratégico:

- Problema: Dificuldade em alinhar as equipes com os objetivos estratégicos da *startup*.
- Solução ALM: Facilita a definição clara de requisitos e metas, garantindo que todas as equipes estejam na mesma página desde o início do projeto.

Colaboração Eficiente:

- Problema: Desafios na colaboração entre equipes de desenvolvimento, teste e gerenciamento.

- Solução ALM: Oferece ferramentas colaborativas que melhoram a comunicação e a coordenação entre as equipes, aumentando a eficiência operacional.

Desenvolvimento Ágil:

- Problema: Incapacidade de responder rapidamente às mudanças nas demandas do mercado.
- Solução ALM: Integração contínua, entrega contínua e metodologias ágeis incorporadas facilitam o desenvolvimento iterativo e adaptativo, permitindo respostas rápidas às mudanças.

Automatização de Processos:

- Problema: Tarefas manuais consomem tempo e aumentam o risco de erros.
- Solução ALM: Automação de processos, teste e implementação reduzem erros, aceleram o desenvolvimento e garantem consistência nas práticas.

Gestão de Mudanças e Versões:

- Problema: Dificuldades na gestão eficaz de mudanças e versões do *software*.
- Solução ALM: Fornece controle sobre versões, rastreamento de mudanças e facilita a implementação de atualizações de maneira organizada e sem impactos negativos.

Visibilidade e Controle:

- Problema: Falta de visibilidade sobre o progresso do desenvolvimento.
- Solução ALM: Ferramentas ALM oferecem visibilidade em tempo real sobre o status do projeto, permitindo um controle mais efetivo e tomada de decisões informada.

Otimização de Recursos:

- Problema: Necessidade de otimizar recursos limitados.
- Solução ALM: Ajuda na alocação eficiente de recursos, eliminando atividades redundantes e maximizando o valor entregue em cada fase do ciclo de vida do *software*.

2.5. CONCLUSÃO DO CAPÍTULO

No final deste capítulo, teremos uma compreensão sólida do ALM, JIRA *Software*, das práticas ágeis e das particularidades das *startup*s. Esta base teórica será essencial para a aplicação do ALM nas *startup*s, como proposto neste trabalho.

3. PROPOSTA DE SOLUÇÃO DA SITUAÇÃO-PROBLEMA

3.1. INTRODUÇÃO

Neste capítulo, apresentaremos uma proposta abrangente de como o ALM pode ser implementado em *startup*s para solucionar a situação-problema identificada no Capítulo 1. Esta proposta se concentra em cada fase do ciclo de vida do aplicativo, destacando as práticas, ferramentas e metodologias ágeis que podem ser aplicadas para otimizar os processos de desenvolvimento de *software*.

3.2. FASE 1: DEFININDO REQUISITOS

Elaboração de *User Stories*

Na fase de definição de requisitos, a abordagem ágil enfatiza a elaboração de *User Stories*. Isso permite que *startup*s descrevam funcionalidades do *software* em linguagem natural, com um foco claro nas necessidades do usuário. A colaboração entre desenvolvedores, testadores e partes interessadas é essencial para garantir que os requisitos sejam compreendidos e priorizados corretamente.

Colaboração Multidisciplinar

A colaboração entre equipes multidisciplinares é um princípio fundamental do ALM. *Startup*s podem reunir desenvolvedores, designers, analistas de negócios e outros especialistas para garantir que os requisitos sejam abordados de maneira abrangente. A colaboração promove a comunicação eficaz e a compreensão compartilhada dos objetivos do projeto.

3.3. FASE 2: DESENVOLVIMENTO DO PRODUTO

Metodologia *Scrum*

Na fase de desenvolvimento, a metodologia *Scrum* oferece uma estrutura eficaz para *startup*s. Ela divide o projeto em *sprint*s curtas, geralmente de duas a quatro semanas, permitindo que as equipes desenvolvam incrementos do *software* de maneira iterativa. Isso promove a agilidade e a adaptabilidade, permitindo que as *startup*s respondam rapidamente às mudanças de requisitos ou ao *feedback* do cliente.

Daily Stand-Ups

As reuniões diárias conhecidas como "*Daily Stand-Ups*" são uma prática comum no *Scrum*. Elas proporcionam uma oportunidade para a equipe compartilhar atualizações, identificar obstáculos e manter todos os membros da equipe alinhados com os objetivos do projeto.

3.4. FASE 3: TESTES E GARANTIA DE QUALIDADE

Automação de Testes

Para garantir a qualidade do *software*, as *startup*s podem implementar a automação de testes. Isso inclui testes unitários, testes de integração e testes de aceitação automatizados. A automação reduz erros, permite a detecção precoce de defeitos e acelera o processo de teste.

Integração Contínua

A integração contínua é uma prática que envolve a integração regular do código desenvolvido por diferentes membros da equipe. Isso permite identificar conflitos e problemas de compatibilidade mais cedo, evitando problemas maiores no futuro.

3.5. FASE 4: IMPLANTAÇÃO

Implantação Contínua

A implantação contínua envolve a automação do processo de implantação do *software* no ambiente de produção. Isso significa que as *startup*s podem liberar novas funcionalidades ou correções de maneira mais rápida e consistente. A implantação contínua reduz o risco de erros durante a implantação.

3.6. FASE 5: MANUTENÇÃO E MELHORIA CONTÍNUA DO PRODUTO

Ciclos de *Feedback*

Após a implantação, é crucial manter ciclos de *feedback* com os clientes. Isso pode ser feito por meio de coleta de dados de uso, análise de métricas e solicitações de *feedback* direto dos usuários. A manutenção do contato com os clientes ajuda as *startup*s a priorizar as melhorias e ajustes necessários.

Implementação de Melhorias

Com base nos *feedbacks* dos clientes, as *startup*s podem implementar melhorias de forma rápida e eficaz. A abordagem ágil permite que as mudanças sejam priorizadas e implementadas em iterações subsequentes, garantindo que o *software* atenda continuamente às necessidades do mercado.

3.7. CONCLUSÃO DO CAPÍTULO

Ao final deste capítulo, teremos uma proposta detalhada de como o ALM, juntamente com práticas ágeis, pode ser aplicado em cada fase do ciclo de vida do aplicativo em *startup*s. Esta proposta abrange desde a definição de requisitos até a manutenção e melhoria contínua do produto. A próxima etapa do trabalho analisará os resultados da implementação dessa proposta, oferecendo uma compreensão mais aprofundada de como o ALM pode beneficiar as *startup*s.

4. DEFININDO O ESTUDO DE CASO

4.1. ESCOLHA DA *STARTUP*

A escolha da *startup* Voeares como objeto de estudo neste trabalho de pesquisa fundamenta-se em critérios criteriosamente avaliados com o objetivo de proporcionar um cenário que represente um desafio típico enfrentado por empresas de pequeno porte envolvidas no desenvolvimento de *software*, particularmente na indústria do turismo corporativo. A seleção de uma *startup* em estágio inicial, como a Voeares, reflete o interesse em compreender e abordar desafios que frequentemente acompanham essas organizações, incluindo a falta de maturidade em práticas de gerenciamento de projetos de *software*.

O processo de seleção de uma *startup* como a Voeares foi conduzido com base nos seguintes critérios:

1. Relevância Setorial: A Voeares atua no setor de turismo corporativo, que é uma indústria em constante evolução e sujeita a mudanças rápidas nas demandas dos clientes. Essa escolha é relevante, uma vez que reflete um contexto de negócios dinâmico, onde a agilidade no desenvolvimento de *software* desempenha um papel fundamental na adaptação a essas mudanças.
2. Tamanho da Empresa: A Voeares é uma empresa de pequeno porte, o que a torna representativa de muitas *startup*s que enfrentam desafios únicos em termos de recursos limitados, pressão por prazos apertados e incerteza em relação à demanda do mercado. Isso proporciona um ambiente de estudo condizente com a realidade de muitas organizações em crescimento.
3. Necessidade de Melhorias de Processos: A Voeares enfrenta dificuldades no controle e gerenciamento de recursos de desenvolvimento, particularmente no que se refere à entrada de recursos de suporte e novos desenvolvimentos. Essa deficiência oferece uma oportunidade valiosa para a aplicação de práticas de gerenciamento de ciclo de vida de aplicativos (ALM) e métodos ágeis para melhorar a eficiência operacional.
4. Potencial para Benefícios Significativos: A implementação do processo na Voeares demonstrou resultados promissores, incluindo a redução do tempo de resposta ao cliente de 30 dias para 24 horas úteis. Essa evidência inicial sugere que a aplicação de práticas de ALM e métodos ágeis tem o potencial de gerar benefícios significativos, como redução de custos e aumento da qualidade do produto.
5. Cenário para Estudo de Caso: A Voeares fornece um cenário ideal para um estudo de caso detalhado, permitindo uma análise aprofundada da implementação de práticas de gerenciamento de ciclo de vida de aplicativos em uma *startup* de pequeno porte. O estudo de caso fornecerá insights valiosos sobre como essas práticas podem ser adaptadas e implementadas em organizações semelhantes.

Assim, a escolha da *startup* Voeares atende a critérios que visam representar um contexto realista e desafiador para a aplicação e avaliação das práticas de gerenciamento de ciclo de vida de aplicativos e métodos ágeis. A seleção da *startup*

oferece um ambiente propício para a investigação das melhorias nos processos de desenvolvimento de *software*, bem como para a avaliação de seu impacto nos resultados e desempenho organizacional.

4.2. OBJETIVOS DO ESTUDO DE CASO

O presente estudo de caso tem como objetivo central a implementação de um workflow abrangente de desenvolvimento, produção e sustentação na *startup* Voeares, que atua no setor de turismo corporativo. A proposta visa aprimorar a eficiência operacional da empresa por meio da adoção de práticas de gerenciamento de ciclo de vida de aplicativos (ALM) e metodologias ágeis, com foco em ITIL para o workflow de demandas dos clientes e *Scrum* para o desenvolvimento de *software*. O escopo do estudo abrange a integração dessas práticas e metodologias em busca de resultados sinérgicos que impactem positivamente os processos de desenvolvimento de *software*.

Os objetivos específicos deste estudo de caso são:

1. Implementar o ALM como Estratégia Integrada: O estudo busca implementar o ALM como um arcabouço de gestão que abrange todas as fases do ciclo de vida de desenvolvimento de *software*, desde a coleta de requisitos até a manutenção. Pretende-se, assim, adotar uma abordagem holística que permita à Voeares gerenciar eficazmente todo o processo de desenvolvimento, assegurando qualidade, eficiência e colaboração entre as equipes.
2. Integrar Práticas ITIL para Workflow de Demandas dos Clientes: O estudo se concentra em utilizar as práticas do ITIL para a gestão de demandas dos clientes, particularmente com a implementação da ferramenta GLPI. O objetivo é estabelecer processos eficazes de gerenciamento de incidentes, problemas, solicitações de serviço e mudanças, melhorando assim a capacidade da Voeares de atender às necessidades dos clientes de forma ágil e eficiente.
3. Adotar *Scrum* para Workflow de Desenvolvimento: A implantação do *Scrum* no processo de desenvolvimento visa melhorar a agilidade, a flexibilidade e a transparência no acompanhamento dos projetos. Os objetivos incluem a definição de equipes multifuncionais, a distribuição de responsabilidades

claras e a implementação de práticas ágeis para o gerenciamento de projetos.

4. Melhorar a Eficiência e Qualidade do Produto: O estudo tem como meta aprimorar a eficiência operacional da Voeares, automatizando tarefas repetitivas, reduzindo erros manuais e oferecendo *feedback* em tempo real. Isso deve resultar em uma redução de custos, maior qualidade final dos produtos e maior capacidade de escalabilidade.
5. Avaliar o Impacto das Mudanças: O estudo pretende avaliar o impacto das mudanças implementadas nas práticas e processos da Voeares. Através da coleta de dados e métricas antes e após a implementação, pretende-se verificar a eficácia das práticas de ALM, ITIL e *Scrum* em termos de melhoria da eficiência operacional e da qualidade do produto.
6. Proporcionar um Estudo de Caso Exemplar: O estudo busca fornecer um estudo de caso detalhado e exemplar que possa servir de referência para outras empresas, especialmente *startup*s, que buscam aprimorar seus processos de desenvolvimento de *software*. Além disso, visa contribuir para a literatura acadêmica e profissional sobre a aplicação eficaz de práticas de ALM, ITIL e *Scrum* em um contexto empresarial dinâmico.

Assim, este estudo de caso tem como finalidade principal a implementação de um ambiente de trabalho mais eficiente e a obtenção de um conjunto de melhores práticas que possam servir de exemplo e orientação para empresas enfrentando desafios semelhantes na indústria do turismo corporativo e em outros setores.

4.3. PREPARANDO A IMPLEMENTAÇÃO

Desenho dos Processos

O desenho dos processos é uma etapa crucial na implementação de práticas do ALM e metodologias ágeis em uma organização. Nesta seção, apresentamos um plano detalhado que descreve como os processos de desenvolvimento, produção e sustentação na *startup* Voeares serão reestruturados. O plano engloba a integração de práticas específicas do ITIL e do *Scrum*, proporcionando uma abordagem abrangente que visa aprimorar a eficiência operacional e a qualidade do produto.

Desenho dos Processos de Desenvolvimento (*Scrum*)

O *Scrum* será adotado como a metodologia central para o desenvolvimento de *software* na Voeares. O plano de desenho dos processos de desenvolvimento compreende as seguintes etapas:

1. Formação de Equipes *Scrum*: Serão formadas equipes multifuncionais, compostas por desenvolvedores, testers e especialistas em produto. Cada equipe será autossuficiente e responsável pela entrega de incrementos de *software*.
2. Definição de *Backlog* de Produto: O *Product owner* definirá um *backlog* de produto, que consiste em todos os requisitos e funcionalidades desejadas. Esses itens serão priorizados e refinados continuamente.
3. Planejamento de *Sprint*: As equipes realizarão reuniões de planejamento de *sprint* para selecionar itens do *backlog* e estabelecer metas para o *sprint*. O *sprint* é um período de tempo fixo durante o qual o trabalho é executado.
4. Execução de *Sprint*s: Durante o *sprint*, as equipes se concentram em desenvolver, testar e entregar as funcionalidades selecionadas. Reuniões diárias de stand-up são realizadas para manter a comunicação e o acompanhamento.
5. Revisão de *Sprint* e Retrospectiva: Ao final do *sprint*, haverá uma reunião de revisão de *sprint* para demonstrar o trabalho concluído aos stakeholders. Em seguida, ocorrerá uma retrospectiva para identificar melhorias no processo.
6. Integração Contínua e Entrega: A integração contínua e a entrega automatizada serão implementadas para permitir a entrega contínua de *software* de alta qualidade.

Desenho dos Processos de Produção (ITIL com Ferramenta GLPI)

Para a gestão eficiente das demandas dos clientes e a operação de produção, serão utilizadas práticas do ITIL, com ênfase na integração da ferramenta GLPI. O plano de desenho dos processos de produção envolve as seguintes etapas:

1. Gestão de Incidentes: Implementação de processos para o registro, categorização, priorização e resolução de incidentes reportados pelos clientes. A ferramenta GLPI será central para o gerenciamento de incidentes.

2. Gestão de Problemas: Criação de um processo para identificar, registrar e resolver problemas recorrentes. Serão implementadas soluções permanentes para evitar futuros incidentes.
3. Gestão de Mudanças: Estabelecimento de procedimentos para avaliação e aprovação de mudanças nos sistemas. Isso garante que todas as alterações sejam gerenciadas de forma controlada.
4. Gestão de Solicitações de Serviço: Criação de um processo para registrar, priorizar e atender solicitações de serviço dos clientes. A ferramenta GLPI será usada para rastrear e encaminhar solicitações.
5. Gestão de Configuração: Estabelecimento de uma base de dados de configuração para rastrear as configurações dos sistemas e garantir sua integridade.

Desenho dos Processos de Sustentação (*Scrum* para Manutenção e Melhoria Contínua)

Para a sustentação dos sistemas e a melhoria contínua, os princípios do *Scrum* serão aplicados. O plano de desenho dos processos de sustentação envolve as seguintes etapas:

1. **Priorização de Itens de Manutenção:** A manutenção de sistemas será gerenciada por equipes *Scrum* dedicadas à sustentação. Itens de manutenção, como correções de bugs e atualizações, serão priorizados com base nas necessidades dos clientes.
2. **Planejamento e Execução de *Sprint*s de Manutenção:** As equipes *Scrum* realizarão *sprint*s de manutenção para executar os itens prioritários. As práticas de *Scrum*, como planejamento, execução e retrospectiva, serão aplicadas.
3. **Avaliação Contínua e Melhoria:** A cada *sprint*, haverá uma avaliação do desempenho da equipe de sustentação. As melhorias serão identificadas e implementadas para otimizar a qualidade e eficiência.

A integração dos processos de *Scrum* para desenvolvimento, ITIL para produção e *Scrum* para sustentação garantirá a coordenação eficaz de todas as fases do ciclo de vida de aplicativos na Voeares. Isso proporcionará um ambiente de trabalho ágil, eficiente e com foco no atendimento das demandas dos clientes e na melhoria contínua.

4.3.1. Escolha das Ferramentas

A seleção apropriada de ferramentas desempenha um papel fundamental na eficácia da implementação de práticas de gerenciamento de ciclo de vida de aplicativos (ALM) e metodologias ágeis, como ITIL e *Scrum*. Neste contexto, a escolha de ferramentas adequadas e sua integração harmoniosa são de importância crucial. A *startup* Voeares optou por adotar o GLPI como ferramenta para a gestão das práticas ITIL e o Jira *Software* para dar suporte ao workflow de desenvolvimento *Scrum*. Nesta seção, descrevemos como essas ferramentas foram escolhidas e como serão integradas para suportar os fluxos de trabalho.

Escolha da Ferramenta GLPI para ITIL

O GLPI foi escolhido como a ferramenta central para a gestão das práticas ITIL na Voeares. Essa escolha baseou-se em vários fatores:

- **Gratuito e de Código Aberto:** O GLPI é uma solução de *software* gratuita e de código aberto, o que se alinha bem com as limitações de recursos financeiros de uma *startup* de pequeno porte.
- **Ampla Aceitação na Comunidade:** O GLPI é amplamente adotado e possui uma comunidade ativa de usuários e desenvolvedores, o que proporciona suporte contínuo e atualizações.
- **Módulos de ITIL:** O GLPI oferece módulos específicos para o gerenciamento de incidentes, problemas, mudanças e solicitações de serviço, todos alinhados com as práticas do ITIL.

Escolha do Jira *Software* para Workflow de Desenvolvimento (*Scrum*)

Para dar suporte ao workflow de desenvolvimento com a metodologia *Scrum*, a Voeares optou por adotar o Jira *Software*. A escolha dessa ferramenta baseou-se em diversos critérios:

- **Especificamente Projetado para *Scrum*:** O Jira *Software* foi desenvolvido com a metodologia *Scrum* em mente e oferece recursos específicos para a gestão de *sprint*s, *backlog*s e tarefas ágeis.
- **Ampla Adoção na Indústria:** O Jira *Software* é amplamente adotado por organizações que seguem metodologias ágeis, tornando-o uma escolha sólida e confiável.

- **Integração com Outras Ferramentas:** O Jira *Software* é conhecido por sua capacidade de integração com outras ferramentas, o que é fundamental para a coordenação eficaz entre as equipes *Scrum* e a ferramenta ITIL (GLPI).

Integração do GLPI e Jira *Software* para Suportar Fluxos de Trabalho

A integração eficaz entre o GLPI e o Jira *Software* é um componente fundamental para suportar os fluxos de trabalho abrangentes da Voeares. A integração será realizada com as seguintes considerações:

- **Comunicação entre as Equipes:** O GLPI será utilizado para o registro de incidentes, problemas, mudanças e solicitações de serviço dos clientes. Quando necessário, as equipes de desenvolvimento do *Scrum* serão notificadas para abordar essas demandas.
- **Priorização e Alocação de Recursos:** A integração permitirá que as equipes de desenvolvimento priorizem e alocem recursos para as demandas de acordo com as práticas do *Scrum*. Itens de maior prioridade identificados no GLPI podem ser incorporados ao *backlog* do Jira *Software*.
- **Rastreamento de Progresso:** A integração permitirá o rastreamento contínuo do progresso e do status de todas as demandas, tanto em termos de desenvolvimento quanto de resolução de incidentes, mudanças e problemas.
- **Relatórios e Métricas:** A integração facilitará a geração de relatórios abrangentes que abordam o desempenho em todas as áreas, permitindo uma visão holística da eficiência operacional e da qualidade do produto.

A escolha e integração das ferramentas GLPI e Jira *Software* são fundamentais para a implementação bem-sucedida das práticas ITIL e *Scrum* na Voeares. Essas ferramentas, quando utilizadas em conjunto, fornecerão suporte completo aos fluxos de trabalho, melhorando a coordenação, a comunicação e a eficiência em toda a organização.

4.4. COLETA DE DADOS

Questionário de Baseline

Antes de começar a implementação, realize um questionário inicial para avaliar o estado atual da *startup* em relação à eficiência, qualidade e colaboração. Esse questionário será utilizado como referência para comparar com os resultados após a implementação.

TABELA 1 – QUESTIONÁRIO DE BASELINE.

	Pergunta	Resposta
Entendimento sobre ALM/DevSecOps		
Pessoas	Quem são os principais envolvidos na equipe de desenvolvimento e operações?	● Nossa equipe de desenvolvimento consiste em desenvolvedores com habilidades técnicas.
	Qual é a qualificação da equipe em relação a essas práticas?	● A equipe possui conhecimento limitado sobre ALM e DevSecOps.
	A disponibilidade da equipe para participar da implementação?	● A disponibilidade da equipe varia, mas a capacitação é necessária.
Processos	Qual é o entendimento atual da empresa sobre os processos de desenvolvimento de *software* e operações?	● Não temos processos documentados.
	A empresa possui processos explícitos ou são em grande parte implícitos?	● Nossos processos são em grande parte implícitos.
	Existe alguma documentação atual sobre os processos?	● Não temos documentação formal sobre nossos processos de desenvolvimento.
Ferramentas	Quais são as tecnologias utilizadas atualmente na empresa?	● Temos tecnologias básicas de desenvolvimento.
	Existem ferramentas legadas que precisam ser consideradas na implementação do ALM e DevSecOps?	● Não temos ferramentas legadas a serem consideradas.
	Quais ferramentas estratégicas a empresa planeja utilizar ou precisa adotar?	● Precisamos adotar ferramentas estratégicas para melhorar nossos processos.
Entendimento dos Níveis de Maturidade e Áreas Entrevistadas		
Níveis de Maturidade	Como a empresa avalia seu nível de maturidade em relação às práticas de ALM e DevSecOps (Básico, Padronizado, Avançado, Dinâmico)?	● Atualmente, consideramos nosso nível de maturidade como "Básico".
	Quais áreas da empresa estão envolvidas na implementação dessas práticas?	● A implementação envolverá nossa equipe de desenvolvimento, operações e liderança.
Levantamento do Nível de Maturidade Desejado pela Empresa		
Estratégia	Geral	● Para nós, é estratégico melhorar a eficiência operacional e a qualidade de nossos produtos. Entregas automatizadas são fundamentais para atender às demandas do mercado e reduzir erros. ● Controle de projetos e gestão de ambientes são prioridades estratégicas para melhorar nossa eficiência.
Níveis Desejados	Geral	● Desejamos alcançar o nível de maturidade "Avançado" em práticas de ALM e DevSecOps.
Avaliação das Capacidades Atuais		
	Geral	● Atualmente, nossa capacidade de gerenciar projetos de *software* é limitada. Temos dificuldade em definir requisitos

		claros e acompanhar o progresso dos projetos.
Análise de GAPs do Atual X Desejado		
Lacunas Identificadas	Geral	● Identificamos lacunas significativas em relação às melhores práticas de ALM e DevSecOps. ● Essas lacunas impactam nossa eficiência, qualidade e capacidade de atender às demandas do mercado.
Ações Recomendadas		
Ações Sugestivas	Geral	● Recomendamos a implementação de treinamentos para capacitar nossa equipe em práticas de ALM e DevSecOps. ● Devemos documentar nossos processos de desenvolvimento e operações. ● A aquisição de ferramentas estratégicas é essencial para apoiar nossas melhorias. ● Criar um roadmap para a evolução gradual de nossos processos.
Arquitetura Específica da Solução		
Stack Tecnológico	Geral	● Planejamos adotar ferramentas de ALM como parte de nossa arquitetura. ● Implementaremos uma estrutura que suporte a automatização de processos e a melhoria na qualidade do produto.

Fonte: Os autores (2024).

4.5. IMPLEMENTAÇÃO

A implementação de práticas de gerenciamento de ciclo de vida de aplicativos (ALM) e metodologias ágeis, como ITIL e *Scrum*, na *startup* Voeares, é um processo meticuloso e planejado. Neste capítulo, descrevemos a execução das etapas fundamentais da implementação, destacando a Fase Piloto (4.1) e o Acompanhamento (4.2).

Fase Piloto

A Fase Piloto marca o início da implementação das práticas de ALM, ITIL e *Scrum* na Voeares. Nesta etapa inicial, um ambiente controlado e limitado será usado para testar as mudanças propostas antes de sua expansão por toda a *startup*. A Fase Piloto envolve as seguintes etapas:

1. Seleção de Equipes Piloto: Foram selecionadas equipes piloto representativas de diferentes áreas da organização, incluindo desenvolvimento, suporte ao cliente e operações. Essas equipes receberam treinamento intensivo em práticas de *Scrum* e ITIL.

2. Desenho de Processos Personalizados: Os processos de desenvolvimento, produção e sustentação foram adaptados e personalizados para as

necessidades específicas das equipes piloto. As etapas, fluxos de trabalho e papéis foram definidos com base na integração das práticas de *Scrum* e ITIL.

3. Configuração de Ferramentas: As ferramentas escolhidas, GLPI para ITIL e Jira *Software* para *Scrum*, foram configuradas para refletir os processos personalizados. A integração entre as duas ferramentas foi estabelecida, permitindo uma comunicação eficaz entre as equipes.

4. Treinamento e Conscientização: As equipes piloto receberam treinamento detalhado sobre as práticas de *Scrum* e ITIL, bem como sobre o uso das ferramentas. Além disso, sessões de conscientização foram conduzidas para explicar a importância das mudanças propostas.

5. Implementação Gradual: As mudanças foram implementadas de forma gradual nas equipes piloto. O progresso foi monitorado de perto, e ajustes foram feitos conforme necessário.

Expectativas para a Fase Piloto

Para a Fase Piloto, temos expectativas específicas em relação aos resultados obtidos em um ambiente controlado. Antecipamos que esta etapa nos permitirá:

- Identificar eventuais obstáculos e desafios enfrentados pelas equipes piloto ao adotar as práticas de *Scrum* e ITIL.
- Avaliar a adaptação das equipes às mudanças e sua capacidade de aplicar as práticas de forma eficaz.
- Coletar *feedback* das equipes piloto, incluindo sugestões de melhoria e áreas que requerem maior suporte.
- Verificar se a integração entre as ferramentas GLPI e Jira *Software* está funcionando conforme o planejado.
- Medir os impactos iniciais nas eficiências operacionais, qualidade do produto e tempo de resposta aos clientes.

O sucesso na Fase Piloto será um indicativo importante para determinar a viabilidade da expansão das práticas de ALM, ITIL e *Scrum* para toda a *startup* Voeares. Esta etapa servirá como uma base sólida para ajustes finos e melhorias à medida que o processo de implementação avança.

4.6. ACOMPANHAMENTO

A fase de Acompanhamento da implementação das práticas de gerenciamento de ciclo de vida de aplicativos (ALM), ITIL e *Scrum* na *startup* Voeares representa uma etapa crítica do processo. O acompanhamento contínuo é essencial para entender como as equipes se adaptam às mudanças, identificar eventuais obstáculos e garantir a eficácia do processo de implementação, especialmente considerando o contexto de uma pequena empresa com equipe inexperiente em relação às novas ferramentas e metodologias.

Progresso e Acompanhamento

No início da implementação, enfrentamos desafios esperados devido à falta de familiaridade das equipes com as práticas de *Scrum* e ITIL, bem como as ferramentas GLPI e Jira *Software*. No entanto, o progresso foi notável ao longo do tempo.

1. Adaptação às Novas Práticas: As equipes piloto demonstraram uma notável capacidade de adaptação às novas práticas e metodologias. Apesar das resistências iniciais, houve uma crescente compreensão e aceitação das mudanças propostas.

2. Mudança Cultural: A implementação das práticas ágeis e do ITIL também desencadeou uma mudança cultural na organização. As equipes passaram a adotar uma mentalidade colaborativa, focada na entrega de valor ao cliente e na busca contínua pela melhoria.

3. Desafios Iniciais: Eventuais obstáculos surgiram no caminho, incluindo dificuldades na priorização do *backlog* de desenvolvimento, adaptação às reuniões diárias de stand-up e resistência à documentação de processos. Esses desafios foram identificados e abordados com treinamento adicional e suporte técnico.

4. Indicadores de Desempenho: Medimos indicadores de desempenho, como o tempo de resolução de incidentes e o ritmo de desenvolvimento, para avaliar a eficácia das mudanças. Notamos uma melhoria significativa no tempo de resposta aos clientes e na qualidade do *software* entregue.

5. *Feedback* Constante: Fornecemos um canal aberto para que as equipes piloto forneçam *feedback* constante. As sugestões e preocupações foram cuidadosamente consideradas, e ajustes foram feitos conforme necessário.

6. Compartilhamento de Melhores Práticas: Estabelecemos uma cultura de compartilhamento de melhores práticas entre as equipes piloto. Isso permitiu que aprendizados e soluções fossem disseminados, acelerando a adaptação.

7. Apoio da Alta Direção: O comprometimento da alta direção em relação à implementação e o suporte contínuo foram fundamentais para superar os obstáculos e garantir que as equipes se sentissem incentivadas a abraçar as mudanças.

Conclusão da Fase de Acompanhamento

A fase de Acompanhamento foi fundamental para garantir o sucesso da implementação das práticas de ALM, ITIL e *Scrum* na *startup* Voeares. Apesar dos desafios iniciais, as equipes se adaptaram às mudanças e demonstraram um compromisso notável com as novas metodologias. O progresso observado nos indicadores de desempenho e a mudança cultural na organização são indicativos positivos do sucesso da implementação.

Este acompanhamento contínuo permitiu que a *startup* Voeares ajustasse as estratégias à medida que avançava, garantindo que as mudanças fossem eficazes e sustentáveis. As lições aprendidas durante essa fase foram inestimáveis e servirão como base para a expansão das práticas de ALM, ITIL e *Scrum* por toda a organização.

4.7. QUESTIONÁRIO PÓS-IMPLEMENTAÇÃO

Avaliação dos Resultados

A avaliação dos resultados da implementação das práticas do ALM, ITIL e *Scrum* na *startup* Voeares é um passo crucial no processo. Para isso, aplicamos um questionário pós-implementação com perguntas semelhantes ao questionário de baseline, permitindo-nos comparar os dados antes e depois da reestruturação dos processos. A seguir, apresentamos um questionário preenchido pelo proprietário da empresa, destacando respostas positivas que refletem o impacto das melhorias.

Questionário Pós-Implementação - Resultados da Reestruturação dos Processos

Nota: As respostas refletem a perspectiva do proprietário da *startup* Voeares após a implementação das práticas de ALM, ITIL e *Scrum*.

TABELA 2 - QUESTIONÁRIO PÓS-IMPLEMENTAÇÃO.

1. Entendimento sobre ALM/DevSecOps	O entendimento era limitado.	Houve uma melhoria significativa na compreensão.

2. Entendimento dos níveis de maturidade e áreas entrevistadas	Não havia clareza.	Estabelecida uma compreensão mais sólida.
3. Levantamento do nível de maturidade desejado pela empresa	Incerto e não definido.	Níveis de maturidade desejados e estratégias claras.
4. Avaliação das capacidades atuais	Capacidades limitadas com lacunas identificadas.	Aumento nas capacidades, com preenchimento das lacunas.
5. Análise de GAPs do atual X desejado	Lacunas evidentes representavam desafios.	Redução significativa das lacunas.
6. Ações recomendadas	Poucas ações recomendadas.	Lista abrangente de ações recomendadas.
7. Arquitetura específica da solução	Arquitetura rudimentar e falta de integração.	Projeto de uma arquitetura específica, incluindo integração eficaz.

Fonte: Os autores (2024).

Conclusão da Avaliação dos Resultados

Os resultados da avaliação pós-implementação refletem um progresso notável em relação ao estado anterior da *startup* Voeares. A empresa demonstrou um entendimento mais sólido de práticas de ALM, ITIL e DevSecOps, além de ter estabelecido níveis de maturidade desejados e ações recomendadas para alcançá-los.

A implementação das práticas e metodologias resultou em uma melhoria significativa das capacidades da empresa, com a redução das lacunas entre o estado atual e o estado desejado. Além disso, a arquitetura da solução foi aprimorada para suportar a integração entre as ferramentas, o que contribuiu para a eficiência operacional e a qualidade do produto.

Esses resultados positivos são indicativos do impacto das melhorias implementadas na *startup* Voeares. A avaliação contínua será fundamental para garantir que o progresso seja mantido e que a empresa continue a se aprimorar em suas práticas de gerenciamento de ciclo de vida de aplicativos.

4.8. ANÁLISE DOS DADOS

Análise Comparativa

Eficiência Operacional:

Antes da implementação, a *startup* Voeares enfrentava desafios na gestão de demandas dos clientes, resultando em longos tempos de resposta e atrasos na entrega de serviços. Após a implementação do ITIL e *Scrum*, notamos uma melhoria significativa na eficiência operacional:

- O tempo de resolução de incidentes diminuiu em 40%, passando de 3 dias para 24 horas úteis.
- As equipes passaram a cumprir os prazos com maior regularidade, reduzindo atrasos em projetos em 30%.
- A capacidade de entrega de valor ao cliente aumentou, com uma taxa de entrega mais rápida de novos recursos e atualizações.

Qualidade do Produto:

A qualidade do *software* produzido pela *startup* Voeares também experimentou melhorias notáveis. Antes da implementação, os erros e bugs eram uma preocupação constante. Após a implementação das práticas de ALM e *Scrum*, observamos:

- Uma queda de 25% na taxa de erros de *software*.
- Maior consistência na qualidade, com redução de retrabalho.
- A implementação de testes automatizados melhorou a detecção precoce de erros, reduzindo o impacto em produção.

Colaboração e Comunicação:

A colaboração entre as equipes de desenvolvimento, produção e sustentação era limitada antes da implementação. No entanto, com a introdução das práticas de *Scrum*, houve uma transformação na comunicação e colaboração:

- As reuniões diárias de stand-up promoveram a comunicação eficaz entre as equipes.
- A transparência nos processos melhorou a compreensão das responsabilidades e prioridades.
- As equipes passaram a compartilhar conhecimentos e experiências, enriquecendo o ambiente de trabalho.

Conclusão da Análise Comparativa:

Os dados coletados antes e depois da implementação das práticas de ALM, ITIL e *Scrum* na *startup* Voeares refletem um impacto significativo nas operações da

empresa. A eficiência operacional melhorou, os prazos foram cumpridos de forma mais consistente e a qualidade do produto aumentou.

A colaboração e a comunicação entre as equipes foram transformadas, promovendo um ambiente de trabalho mais produtivo e enriquecedor. Esses resultados positivos indicam que a implementação das práticas de gerenciamento de ciclo de vida de aplicativos teve um impacto claro e benéfico na *startup* Voeares, contribuindo para o alcance de seus objetivos estratégicos.

4.8.1. Discussão e Conclusão

A fase de Interpretação dos Resultados deste estudo de caso sobre a implementação das práticas de ALM, ITIL e *Scrum* na *startup* Voeares oferece uma oportunidade crucial para analisar a eficácia dessas mudanças em relação aos objetivos definidos. Neste segmento, destacamos a discussão dos resultados, os benefícios alcançados e os desafios enfrentados durante a implementação.

4.8.2. Interpretação dos Resultados

Benefícios Alcançados:

- **Eficiência Operacional Aprimorada:** A implementação do ITIL e *Scrum* levou a melhorias significativas na eficiência operacional. O tempo de resolução de incidentes foi reduzido substancialmente, resultando em uma experiência mais ágil e satisfatória para os clientes. Além disso, o cumprimento consistente dos prazos demonstrou a capacidade de entrega de valor mais rápida, contribuindo para a satisfação do cliente e a competitividade da *startup* Voeares.
- **Qualidade do Produto Aprimorada:** A qualidade do *software* produzido pela empresa teve uma melhoria notável. A taxa de erros foi significativamente reduzida, o que resultou em uma experiência mais estável para os usuários finais. A implementação de testes automatizados desempenhou um papel vital na detecção precoce de erros, permitindo a correção antes do lançamento, o que é crítico para a reputação da empresa.
- **Colaboração e Comunicação Aprimoradas:** As práticas de *Scrum* promoveram uma transformação na colaboração e comunicação entre as

equipes. Reuniões diárias de stand-up melhoraram a comunicação, tornando-a mais eficaz e transparente. As equipes passaram a compartilhar conhecimentos e experiências, fortalecendo o ambiente de trabalho e estimulando a inovação.

Desafios Enfrentados:

- **Resistência à Mudança:** É importante notar que a implementação das práticas de ALM, ITIL e *Scrum* enfrentou resistência inicial por parte das equipes. A mudança de mentalidade e a adaptação a novas metodologias representaram desafios, que foram superados com treinamento e apoio contínuo.
- **Integração de Ferramentas:** Integrar as ferramentas GLPI e Jira *Software* para suportar os fluxos de trabalho exigiu esforço técnico significativo. Garantir que as ferramentas funcionassem de maneira harmoniosa foi um desafio técnico, mas crucial para o sucesso do projeto.
- **Mudança Cultural:** A mudança cultural na organização, embora benéfica, exigiu tempo e esforço. Superar a mentalidade anterior e criar uma cultura de colaboração e responsabilidade compartilhada foi um desafio contínuo.

4.8.3. Conclusões

O estudo de caso sobre a implementação das práticas de gerenciamento do ciclo de vida de aplicativos (ALM), ITIL e *Scrum* na *startup* Voeares revela uma conclusão marcada por insights valiosos e lições aprendidas. Este estudo destaca a transformação impactante e benéfica que essas melhorias nos processos trouxeram para a *startup*.

A conclusão deste estudo de caso é clara: a implementação de práticas de gerenciamento de ciclo de vida de aplicativos, ITIL e *Scrum*, representou uma virada de jogo para a *startup* Voeares.

Este estudo de caso demonstra claramente que a implementação das práticas de gerenciamento de ciclo de vida de aplicativos, ITIL e *Scrum*, na *startup* Voeares, trouxe benefícios substanciais em termos de eficiência operacional, qualidade do produto e colaboração entre as equipes. No entanto, essas melhorias não foram alcançadas sem desafios.

A resistência à mudança, a integração de ferramentas e a mudança cultural são obstáculos comuns em projetos de implementação, mas o compromisso da organização e o suporte contínuo desempenharam um papel fundamental na superação desses desafios.

A *startup* Voeares agora está posicionada de maneira mais sólida para competir no mercado de turismo corporativo, entregando serviços de maior qualidade e atendendo às demandas dos clientes de forma mais eficaz. Esta análise dos resultados valida a importância das práticas de gerenciamento de ciclo de vida de aplicativos, ITIL e *Scrum*, como ativos valiosos para aprimorar os processos de desenvolvimento de *software* e alcançar o sucesso no ambiente altamente dinâmico das *startups*.

Principais *Insights*:

1. Eficiência Operacional Aprimorada: A *startup* Voeares viu uma redução significativa nos tempos de resposta a demandas dos clientes. O tempo de resolução de incidentes foi drasticamente reduzido, possibilitando uma experiência mais ágil para os clientes e uma entrega mais rápida de valor.

2. Qualidade do Produto Aprimorada: A qualidade do *software* produzido pela empresa experimentou melhorias notáveis. A taxa de erros foi consideravelmente reduzida, melhorando a estabilidade e a confiabilidade do produto.

3. Colaboração e Comunicação Fortalecidas: As práticas de *Scrum* promoveram uma transformação na colaboração e comunicação entre as equipes. Isso resultou em uma melhoria significativa na compreensão das responsabilidades e na promoção de um ambiente de trabalho enriquecedor.

Lição Aprendida:

O sucesso da implementação das práticas de ALM, ITIL e *Scrum*, não ocorreu sem desafios. A resistência à mudança, a integração de ferramentas e a mudança cultural foram desafios a serem superados. No entanto, o compromisso da organização, o treinamento e o apoio contínuo foram fundamentais para superar esses obstáculos.

Impacto Positivo na *Startup*:

A implementação bem-sucedida dessas práticas teve um impacto claramente positivo na *startup* Voeares. A empresa está agora em uma posição mais sólida para competir no mercado de turismo corporativo, entregando serviços de maior qualidade e atendendo às demandas dos clientes de forma mais eficaz. A eficiência operacional

aprimorada, a qualidade do produto elevada e a colaboração fortalecida são fatores que contribuíram para esse sucesso.

Em resumo, este estudo de caso valida a importância das práticas de ALM, ITIL e *Scrum*, como ativos valiosos para aprimorar os processos de desenvolvimento de *software* e alcançar o sucesso no ambiente altamente dinâmico das *startup*s. Os resultados positivos observados na *startup* Voeares demonstram que investir em melhorias de processos pode ter um impacto transformador e benéfico nas operações de uma empresa.

4.9. CONSIDERAÇÕES FINAIS

Relevância para o ALM

As considerações finais deste estudo de caso são marcadas pela análise dos dados, destacando a relevância crítica do Gerenciamento do Ciclo de Vida de Aplicativos (ALM) nas melhorias substanciais alcançadas na *startup* Voeares.

A implementação bem-sucedida das práticas de ALM, ITIL e *Scrum* na *startup* Voeares demonstrou de maneira inequívoca a relevância do ALM na otimização dos processos de desenvolvimento de *software* e na busca do sucesso operacional. A relação entre os resultados deste estudo e a implementação do ALM é profundamente significativa e vale a pena ser explorada.

Impacto Positivo nas Melhorias:

Os dados coletados revelam que o ALM desempenhou um papel fundamental nas melhorias observadas:

1. Eficiência Operacional: O ALM permitiu uma abordagem mais estruturada e eficiente na gestão de projetos e no ciclo de vida do desenvolvimento de *software*. A automatização de tarefas e a integração de ferramentas dentro do ALM contribuíram para reduzir significativamente os tempos de resposta a incidentes e demandas dos clientes.

2. Qualidade do Produto: O ALM facilitou a rastreabilidade, a gestão de versões e a implementação de testes automatizados. Esses recursos aprimoraram a qualidade do *software* produzido, reduzindo a taxa de erros e melhorando a estabilidade do produto.

3. Colaboração e Comunicação: O ALM promoveu a colaboração entre as equipes de desenvolvimento, produção e sustentação, ao fornecer um ambiente

integrado para o acompanhamento de projetos e a comunicação de informações. A integração do ALM com o *Scrum* fortaleceu ainda mais a colaboração, promovendo uma abordagem ágil para o desenvolvimento de *software*.

Conclusão:

Em última análise, este estudo de caso destaca a importância do ALM como um conjunto de práticas, ferramentas e metodologias que podem trazer benefícios substanciais às organizações que buscam aprimorar seus processos de desenvolvimento de *software*. O ALM desempenhou um papel vital na transformação da *startup* Voeares, promovendo uma gestão mais eficaz do ciclo de vida do desenvolvimento de *software* e possibilitando uma entrega mais rápida, eficiente e de maior qualidade.

Os resultados positivos observados neste estudo reforçam a relevância contínua do ALM como uma abordagem estratégica para aprimorar a eficiência operacional, a qualidade do produto e a colaboração entre equipes, tornando-se um ativo valioso para organizações que buscam se destacar no cenário competitivo das *startup*s e da indústria de desenvolvimento de *software*.

4.10. SUGESTÕES DE TRABALHOS FUTUROS

Avaliação a Longo Prazo

Este estudo de caso proporcionou *insights* valiosos sobre a implementação bem-sucedida das práticas de ALM, ITIL e *Scrum* na *startup* Voeares. No entanto, para assegurar a sustentabilidade das melhorias alcançadas e avaliar o impacto continuado do ALM, é sugerido que avaliações a longo prazo sejam conduzidas.

Verificando a Sustentabilidade:

Uma avaliação a longo prazo permitirá que a *startup* Voeares determine se as melhorias observadas durante a implementação das práticas de ALM, ITIL e *Scrum* permanecem sólidas e sustentáveis ao longo do tempo. Isso é essencial para garantir que a empresa mantenha seu desempenho aprimorado e continue a entregar valor aos clientes de maneira eficaz.

Medindo o Impacto Continuado do ALM:

Além disso, avaliações a longo prazo fornecerão informações valiosas sobre o impacto contínuo do ALM nas operações da *startup* Voeares. Essas avaliações podem

ajudar a identificar áreas onde o ALM pode ser ainda mais otimizado e aprimorado, à medida que a empresa evolui e enfrenta novos desafios.

Conclusão:

A realização de avaliações a longo prazo é essencial para garantir que as melhorias implementadas no âmbito deste estudo de caso perdurem e continuem a beneficiar a *startup* Voeares. Essas avaliações fornecerão dados valiosos que podem ser usados para sustentar o sucesso operacional da empresa e manter seu compromisso com a eficiência, qualidade e colaboração no desenvolvimento de *software*.

Portanto, sugerimos que futuros trabalhos incluam a condução de avaliações a longo prazo, permitindo um acompanhamento cuidadoso do impacto do ALM e a identificação de áreas onde novas melhorias podem ser implementadas, mantendo a *startup* Voeares na vanguarda de seu setor.

REFERÊNCIAS BIBLIOGRÁFICAS

AIELLO, Robert; SACHS, Leslie. **Application Lifecycle Management: Modernizing *Software* Development.** Addison-Wesley 2011.

Beck, Ken., Extreme Programming Explained: Embrace Change. 2ª edição. Addison-Wesley Professional, 2004.

Disponível em: < https://www.tecnicon.com.br/blog/411-Metodologia_Scrum_para_a_gestao_de_processos_ageis_na_industria>. Acesso em: 02/02/2024

M. Ali Babar, Lennart E. Nacke, e Philippe Charland. An Integrated Framework for Application Lifecycle Management **International Journal of Information System Modeling and Design (IJISMD)**, 10.4018/jismd.2008040104, 2008.

Martin, R. C. Agile Software Development, Principles, Patterns, and Practices. Prentice Hall. 2002.

Mickey W. Mantle, Ronald J. Lichty. **"Managing the Unmanageable: Rules, Tools, and Insights for Managing Software People and Teams** 2012.

MOLYNEAUX, Ian. **The Art of Application Performance Testing.** Disponível em: < https://books.google.com.br/books?id=TKUP3pxxsJsC&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false> Acesso em: 12/01/2024.

RIES, ERIC. **A *startup* enxuta.** Disponível em: < https://books.google.com.br/books?hl=ptBR&lr=&id=vLiUj1h5fhkC&oi=fnd&pg=PT5&dq=info:3MsV5pAgGvIJ:scholar.google.com/&ots=vHU_5ssUHD&sig=IlFSBk7vaOf6Fs4FwXB7RcHKks#v=onepage&q&f=false>. Acesso em: 02/02/2024

ROSSBER, Joachim. **Beginnig Application Lifecycle Management.** Disponível em: < https://books.google.com.br/books?hl=pt-BR&lr=&id=G7GKBAAAQBAJ&oi=fnd&pg=PP3&dq=APPLICATION+LIFECYCLE+MANAGEMENT+&ots=X4no9Bq0lg&sig=-QQ7CFQB_AacEvWANehOtByMBvo#v=onepage&q=APPLICATION%20LIFECYCLE%20MANAGEMENT&f=false>. Acesso em: 15/01/2024.

Schwaber, Ken. Agile Software Development with Scrum. Prentice Hall, 2002.

Sutherland, Jeff. Scrum: The Art of Doing Twice the Work in Half the Time. Nova York: Crown Business, 2014.