XXIV SEMEAD
Seminários em Administração
novembro de 2021
ISSN 2177-3866

INTEGRAÇÃO COLABORATIVA E AGILIDADE EM PROJETOS DE TECNOLOGIA DE INFORMAÇÃO DO SENAC SP

MARCELO DINIZ AUGUSTO
UNIVERSIDADE PRESBITERIANA MACKENZIE (MACKENZIE)

ALBERTO DE MEDEIROS JR
UNIVERSIDADE PRESBITERIANA MACKENZIE (MACKENZIE)

INTEGRAÇÃO COLABORATIVA E AGILIDADE EM PROJETOS DE TECNOLOGIA DE INFORMAÇÃO DO SENAC SP

1. INTRODUÇÃO

As metodologias de desenvolvimento de projetos de TI e as melhores práticas para gerenciamento e integração dos times de infraestrutura e desenvolvimento, são considerados por Veras (2014) fatores determinantes na maneira como os projetos tecnológicos são entregues, sejam eles na análise quanto ao prazo acordado, sejam eles quanto os escopos definidos no início do projeto, que estão sempre em constante mudança.

Este artigo aplicado é um estudo sobre as metodologias de gestão de projetos de TI e integração das equipes no ambiente corporativo do Senac de São Paulo, analisando a aderência das melhores práticas de gerenciamento. O estudo amparou-se em diferentes metodologias de gestão de projetos, Gestão Ágil e *DevOps,* demonstrando as suas utilidades e aplicabilidade e as fases que são percorridas até o momento da entrega do resultado.

A adoção de um modelo gerencial tem por base um forte alinhamento aos objetivos estratégicos do negócio e às respectivas áreas em questão, tentando obter a melhor solução e não obstante as lacunas que possam ser identificadas durante a sua adoção.

Ao se analisar o cenário atual e a estrutura organizacional do Senac São Paulo foi possível observar um relativo aumento nos atrasos dos cronogramas de entregas de projetos de tecnologia e na colaboração entre especialistas de tecnologia.

Identificado como esse problema, para solucioná-lo, buscou-se entendê-lo, analisando suas ocorrências.

As estratégias para solução do problema; o plano de ação e seus desdobramentos; a avaliação dos resultados obtidos; juntamente com as propostas de solução.

2. CONTEXTO INVESTIGADO

O Senac – Serviço Nacional de Aprendizagem Comercial, é uma instituição brasileira de educação profissional criada em 10 de janeiro de 1946 por um decreto de lei. É uma entidade privada com fins públicos que recebe contribuição compulsória das empresas do comércio e é administrada nacionalmente pela Confederação Nacional do Comércio.

2.1. O serviço e a empresa

O Senac está presente há mais de 70 anos em todos os estados do Brasil, em mais de 1850 municípios e capacitou mais de 2 milhões de brasileiros por ano, de acordo com as informações do site a nível Brasil do Senac (2020). Trata-se de uma empresa de capital mista de ensino e está entre as vinte maiores instituições de ensino no Brasil e com aproximadamente 9000 funcionários no estado de São Paulo.

Devido à abrangência nacional, o departamento de tecnologia da informação, conhecido como GTI (Gerência de Tecnologia da Informação), possui diversos projetos com diferentes soluções a serem desenvolvidos e implantadas, tanto para os alunos, que são considerados os clientes finais, como para os departamentos internos do Senac, que possuem soluções estratégicas e administrativas que precisam ser apoiadas e desenvolvidas, que são os clientes internos. Atualmente o departamento de tecnologia, possui mais de 300 projetos para o ano de 2020, assim como diversas outras soluções e melhorias que entram em uma fila para provisionamento de recursos e priorização de demanda para iniciarem como projeto. Entende-se como clientes externos os alunos e usuários de

aplicações externos ao Senac, os clientes internos são os funcionários que trabalham na empresa, seja na sede de São Paulo ou unidades operacionais.

A gerência de tecnologia da informação é composta por uma equipe de aproximadamente 120 funcionários contratados em regime CLT e colaboradores terceiros, divididos em diferentes equipes e especialidades.

Em 2020 havia um *backlog* de 303 projetos a serem contemplados, assim como, 603 projetos de melhorias que são considerados ajustes em projetos já entregues. Os projetos e melhorias são dinâmicos, adicionados a todo instante, sendo importante a velocidade da entrega para não aumentar ainda mais a fila de projetos.

3. DIAGNÓSTICO DA SITUAÇÃO-PROBLEMA

Para Marcondes, Miguel, Franklin e Perez (2017), faz-se necessário aprofundar o entendimento do problema, no sentido de validá-la, modificá-la ou mesmo substituí-la e ela vai ser tratada no âmbito do problema.

Um dos instrumentos utilizados para o diagnóstico da situação-problema é o diagrama de Ishikawa, também conhecido como diagrama espinha de peixe, ou diagrama de causa e efeito.

De acordo com Ishikawa (1993) o agrupamento dos fatores deve ser corretamente controlado a fim de que os processos sejam transformados em causas que resulte em bons produtos e efeitos. Trata-se de uma ferramenta para analisar as causas de um determinado problema e analisá-las de maneira organizada, diminuindo as chances de algum problema ser esquecido. O diagrama auxilia na relação existente entre o resultado em não conformidade ou indesejado de um processo (efeito) e os diferentes fatores (causas) que podem contribuir para que determinado problema ou resultado tenha ocorrido. Esta ferramenta se caracteriza como um instrumento para se aplicar no controle da qualidade, aplicável em atividades diversas, de modo que contribui na identificação de desvios no fluxo logístico, observando uma possível existência e localização dos gargalos na organização em que se aplicar a ferramenta da análise da espinha de peixe (Ishikawa, 1993).

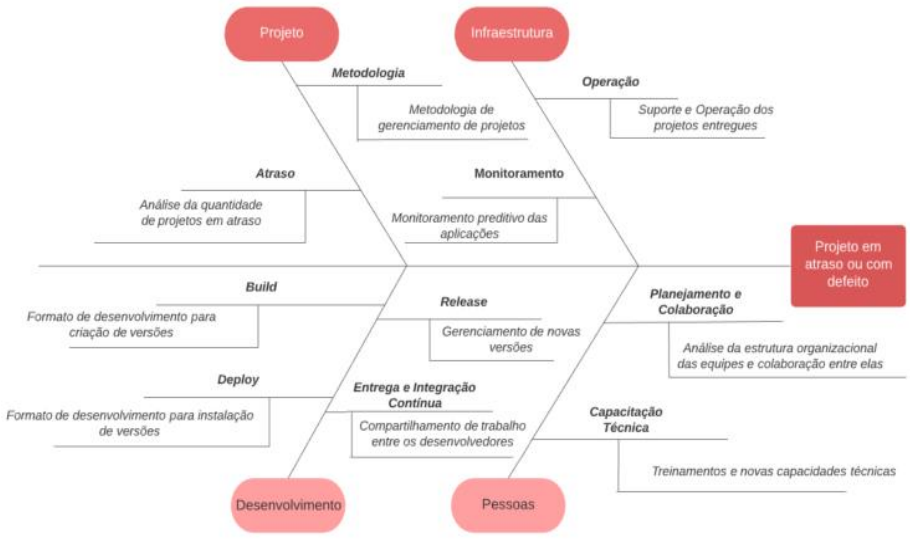

Figura 1: Diagrama de Ishikawa
Fonte: Desenvolvido pelos autores (2020)

Com base em observações e análises de cada entrega e por dificuldades notórias em participações nos projetos, assim como a vivência no departamento de tecnologia, foi possível analisar e destacar as percepções e elaborar conforme a figura 1, os processos e as causas que resultam em um projeto com atraso ou defeito:

- **Projeto** – a metodologia utilizada, assim como a quantificação de projetos em atraso, a organização colaborativa para execução do projeto e o formato que o planejamento é executado.
- **Metodologia** – Metodologia para gestão e condução de projetos.
- **Atraso** – Análise dos dados por ferramentas de BI para visualização dos projetos em atraso e dentro do cronograma
- **Infraestrutura** – Entende-se como infraestrutura os compostos tecnológicos, tais como: servidores, armazenamento de dados. A forma como o monitoramento dos serviços são realizados e são operados, assim como o nível de automação analisados.
- **Monitoramento** – Metodologia e ferramentas para monitoramento das aplicações e disponibilidade dos ambientes.
- **Operação** – Suporte das aplicações e funcionalidades, operação com foco na análise preditiva e melhoria da disponibilidade.
- **Desenvolvimento** – O método de desenvolvimento utilizado pelos desenvolvedores e a maneira como são lançadas novas versões e o versionamento dela, assim como a entrega desses serviços e a integração entre elas.
- ***Build*** – Maneira como são geradas as versões dos projetos e a ferramenta utilizada para gerenciamento.
- ***Deploy*** – Formato como são realizados os desenvolvimentos dos projetos para instalação das versões em ambientes de homologação, desenvolvimento e produção.
- ***Release*** – Ferramenta utilizada e metodologia para releases de versões das aplicações.
- **Entrega e Integração Contínua** – metodologia e ferramentas para compartilhamento de trabalhos e rotinas entre os desenvolvedores.
- **Pessoas** – os papéis e responsabilidades dos colaboradores, analistas e desenvolvedores e a capacidade técnica as mudanças e novas tecnologias abordadas.
- **Capacitação Técnica** – Análise das equipes quanto as melhorias propostas e conhecimentos tecnológicos e a eventual necessidade de treinamentos.
- **Planejamento e Colaboração** – Análise da estrutura organizacional para verificação da necessidade de ajustes na estrutura organizacional assim como acontecem a colaboração das equipes entre os projetos.

Para o diagnóstico do problema, também foram coletados dados dos projetos atribuídos para o ano de 2020 e foram realizadas análises nas equipes de tecnologia e as suas respectivas divisões, onde, atualmente é utilizado o *framework* do ITIL como metodologia de projetos, governança de TI e governança corporativa.

De acordo com o PMBOK (2011) “um projeto é um esforço temporário empreendido para criar um produto, serviço ou resultado exclusivo. Os projetos e operações diferem, principalmente, no fato que os projetos são temporários e exclusivos, enquanto as operações são contínuas e repetitivas”.

O departamento de tecnologia do Senac São Paulo, possui em seu *backlog* (listagem de projetos aguardando para execução) um total de 724 projetos, que são categorizados como:

- **Melhoria -** Projetos já implantados que precisam de ajustes, novas funcionalidades ou correções.
- **Novos Projetos -** Projetos que precisam ser desenvolvidos e são considerados como novas soluções ou entregas.

Pode-se, ainda, identificar também, quantidade de projetos e melhorias que precisam ser entregues, com o envolvimento de diversas equipes, sejam elas técnicas, divididas por equipes e especialização, ou sejam elas de negócio, divididas por departamento. As informações foram retiradas da ferramenta Project da Microsoft, que é a aplicação institucional para controle de projetos.

Para realização dos projetos, é seguido um modelo de distribuição departamental, dentro da Gerência de Tecnologia, utilizando o conceito de distribuição por especialidades e aplicações, ou seja, são separados por "silos". Os departamentos dentro das suas especialidades são integrados e compartilham os mesmos projetos, ora como responsável e em outro momento como participantes ou colaboradores, levando em conta no momento do planejamento, o cronograma da equipe responsável pelo projeto no momento de sua definição.

Trata-se de um modelo tradicional, denominado cascata (w*aterfall*), deixa evidente as etapas que os projetos precisam percorrer, pois precisam ser executadas de maneira sequencial o que acaba e deixa fácil identificar as etapas nas quais o projeto se encontra.

3.1 Proposta de Solução

A solução oferecida é a adoção de uma metodologia amparada com um estudo de caso junto aos gestores das equipes de tecnologia, que consiste em aperfeiçoar e melhorar o processo de gestão de projetos e as suas dependências tecnológicas, para possibilitarem um trabalho em conjunto visando a melhoria de tempo de execução dos projetos, assim evitando ou diminuindo o atraso nas suas respectivas entregas.

Para evitar o atraso nos projetos o departamento de tecnologia deve-se formatar uma nova proposta para controle de cronograma de projetos, uma participação mais próxima com os departamentos envolvidos, novos formatos tecnológicos e assim permitindo a mudança no formato de gestão e condução dos projetos. Penrose (1959) considera que a singularidade de cada empresa está na distinção entre recursos e os possíveis serviços que podem ser obtidos com o emprego de cada um deles. Assim, a GTI do Senac atua na solução dos problemas, levando em conta as limitações institucionais e arranjos departamentais, preparando o negócio para o novo modelo que agilize o desenvolvimento de projetos e que possibilite a gestão colaborativa entre os desenvolvedores e os operadores da tecnologia propiciando uma entrega contínua das soluções.

A seguir, serão utilizadas técnicas consagradas para metodologias Ágil e *DevOps*, listando as que se aplicam na solução do problema e na proposta do trabalho.

3.2 Gestão Ágil de Projetos

O termo "ágil" está cada vez mais presente nas empresas e acaba se confundindo com agilidade ou projetos mais rápidos. De acordo com Highsmith (2012), que diz que a agilidade é a habilidade para criar e responder às mudanças para conseguir obter lucro nos negócios. Reforçando o conceito, Kruchten (2008, p. 3), reconhecida referência em processos de desenvolvimento de software, define "ágil" como sendo "a habilidade que uma organização tem de reagir e se adaptar a mudanças no seu ambiente de forma mais rápida do que a taxa dessas mudanças". De acordo com Camargo (2019, p. 100) "ágil" é

“a habilidade de entregar valor aos clientes, em um ambiente dinâmico, com incerteza, volatilidade, mudanças e adaptação constante”.

Adotar o modelo ágil, portanto, não significa fazer de forma rápida, mas ter um *mindset* ágil, isto é, deve-se ser efetivamente ágil. Quando se fala em *mindset*, quer-se referenciar a um modelo mental pelo qual se compreende como as pessoas trabalham, como se relacionam e se comunicam entre si e, como geram valor aos clientes. Essa dificuldade, pode ser um fator determinante para a solução dos problemas, reforçado novamente por Camargo (2019, p. 102), que afirma que “[...] não existe uma lista pronta de metodologias e ferramentas predefinidas para serem utilizadas em todas as situações”. Existem, porém, algumas práticas que podem ser adotadas para começar a gerenciar os projetos de forma ágil, como veremos mais adiante.

Essa metodologia vai de encontro ao atual modelo ora adotado baseado no controle e definido por etapas, ou seja, o modelo cascata utilizado pela GTI, onde o departamento ou a empresa deixa de ser responsável por processos e ferramentas de maneira sequencial para com a agilidade, no contexto não ser puramente rapidez, auxiliar a chegar no lugar certo, com o escopo correto no menor tempo possível.

3.2.1. Manifesto Ágil

No ano de 2001, um grupo formado por 17 especialistas em desenvolvimento de software se reuniu em Utah, nos Estados Unidos, para discutir uma nova forma de gerar melhores resultados em seus projetos. Eles buscavam uma alternativa ao modelo sequencial de desenvolvimento de software, como o ora adotado na GTI, em formato de cascata e sequencial. Embora o manifesto seja direcionado para a tecnologia, ele pode ser aplicado na empresa em diferentes departamentos e setores.

De acordo com a organização criadora da metodologia ágil (agilemanifesto.org), há quatro principais pilares:

1. **Indivíduos e interação entre eles mais do que processos e ferramentas:** fundamentada nos pilares na comunicação e interação entre os participantes do projeto e não focando somente nos processos e ferramentas a serem utilizadas. Um ambiente criativo para solucionar problemas por meio de tarefas organizadas e comunicação frequente.
2. **Software em funcionamento mais do que documentação abrangente**: foco na entrega e no funcionamento do software. Dedicando uma parte maior para entregas e não somente em uma documentação abrangente, dividindo em espaços curtos de tempo ao em vez de um longo planejamento.
3. **Colaboração do cliente mais do que negociação de contratos:** participação colaborativa desde o momento da criação do projeto, onde a participação será mais importante que as tratativas contratuais e acordos entre áreas. Os contratos continuarão existindo, mas esse manifesto diz respeito sobre a importância da participação dos clientes e participantes do projeto.
4. **Responder a mudanças mais do que seguir um plano:** o escopo sofre mudanças a todo instante e um dos fatores determinantes será como se adaptará a essas mudanças e a velocidade que elas serão realizadas e ajustadas.

3.2.2 Princípios do Manifesto Ágil

Podem-se citar também, os princípios do manifesto ágil de acordo com a organização (agilemanifesto.org):

- Nossa maior prioridade é satisfazer o cliente pela entrega contínua e adiantada de software com valor agregado.
- Mudanças nos requisitos são bem-vindas, mesmo tardiamente no desenvolvimento. Processos ágeis tiram vantagem das mudanças visando vantagem competitiva para o cliente.
- Entregar frequentemente software funcionando, de poucas semanas a poucos meses, com preferência à menor escala de tempo.
- Pessoas de negócio e desenvolvedores devem trabalhar diariamente em conjunto por todo o projeto.
- Construa projetos em torno de indivíduos motivados. Dê a eles o ambiente e o suporte necessário e confie neles para fazer o trabalho.
- O método mais eficiente e eficaz de transmitir informações para e entre uma equipe de desenvolvimento é pela conversa face a face.
- Software funcionando é a medida primária de progresso.
- Os processos ágeis promovem desenvolvimento sustentável. Os patrocinadores, desenvolvedores e usuários devem ser capazes de manter um ritmo constante indefinidamente.
- Contínua atenção à excelência técnica e bom design aumenta a agilidade.
- Simplicidade—a arte de maximizar a quantidade de trabalho não realizado—é essencial.
- As melhores arquiteturas, requisitos e designs emergem de equipes auto-organizáveis.
- Em intervalos regulares, a equipe reflete sobre como se tornar mais eficaz e então refina e ajusta seu comportamento de acordo.

3.3 Metodologia Ágil: *Scrum*

O *Scrum* é um exemplo de Metodologia Ágil a ser adotada no desenvolvimento de softwares que se apoia nos princípios do Manifesto Ágil para elaboração e dinâmica do formato de trabalho e de acordo com Heller (2018) é uma das metodologias mais utilizadas pelos times de desenvolvimento de software e a mais usada para exemplificar os conceitos do ágil.

De acordo com Cohn (2011, p.30) em um projeto *Scrum*, testadores e programadores têm que desaprender alguns comportamentos. Os testadores precisam aprender que o teste deve abordar a conformidade e necessidades do usuário, assim como, os programadores precisam entender que não é necessário um projeto em sua totalidade para iniciar a codificação.

A metodologia abordada, o *Scrum*, parte de um *Product Backlog*, que são as listas de coisas a se fazer, retirando dessa lista os itens que precisam ser desenvolvidos e tudo isso acontecendo um ciclo pré-definido de tempo. Os ciclos definidos são chamados de sprints, que são o resultado de um produto ou funcionalidade no período definido e normalmente duram de 2 a 4 semanas, também com reuniões diárias entre os times para verificação de problemas, ou necessidade de ajuda e ajustar qualquer problema que possa estar acontecendo no período.

No final dos *sprints*, após validados, o time entrega os itens e pode pegar novos produtos no *backlog*.

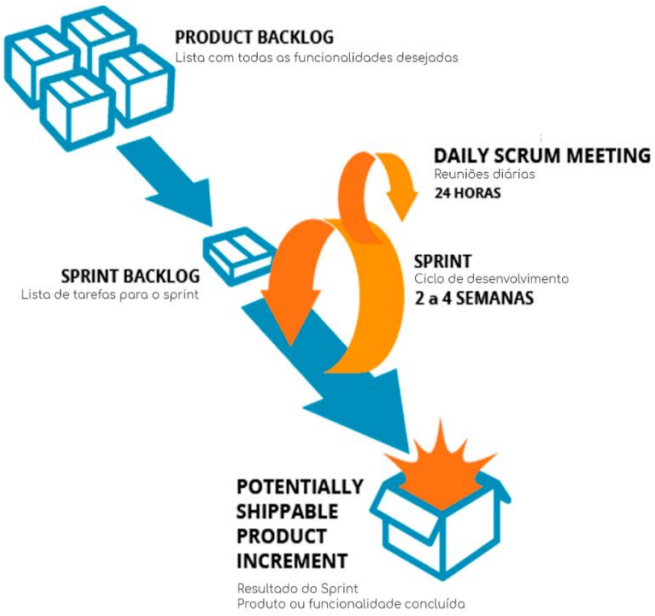

Figura 2: Modelo ágil de desenvolvimento de software (*Scrum*)
Fonte: https:// www.hellerhaus.com.br/metodologia-agil//

Na figura 2, é possível verificar o fluxo de entregas desde o *backlog* e o resultado das entregas, com as reuniões e sprints, até o momento da entrega de um produto ou funcionalidade.

De acordo Vettorazzo (2018, p.17) *Scrum* é um *framework* do qual as pessoas podem tratar e resolver problemas complexos e adaptativos, enquanto produtiva e criativamente entregam produtos com mais alto valor possível. É leve, simples de entender e extremamente difícil de dominar. No *Scrum*, os projetos estão divididos em ciclos e o *sprint* representa um *time box* dentro do qual um conjunto de atividades devem ser executadas, sendo elas: requisitos, Análise, Projeto, Evolução e Entrega.

Para Vettorazzo (2018, p.19) quadro *Scrum* é o lugar onde você pode organizar seu *backlog*, assim como tarefas que estão sendo trabalhadas no sprint atual e seu andamento. Existe uma lista de *backlogs* criados, que estão em formatos de *cards* que são movimentados e divididos em: item de *backlog*, a fazer, fazendo e feito. Esses *cards* são movimentados conforme o cronograma ou a tarefa muda de etapa.

Existem diferentes ferramentas *Scrum* para auxiliar os times na criação dos *backlogs*, criações de *cards* (figuras interativas de trabalho) que podem ser acessadas pela internet e facilitam a troca de informações e ajudam as equipes a estruturar e melhorar seus fluxos de trabalho. Usualmente utiliza-se um recurso visual para a organização dos *cards* e *sprints* denominado Trello. Nessa interface, os cartões são movimentados e podem ser designados para recursos específicos, com organização das datas até o momento da entrega que acontecem de maneira faseada, caracterizado pelos *sprints*.

3.4 *DevOps*

A palavra *DevOps* é a contração em inglês das equipes envolvidas na construção e implantação de software. O "*Dev*", é referente à equipe de desenvolvimento responsável pelos testes, codificação e análise do projeto (*Development*), já o "*Op*", trata-se da equipe responsável pela solução de incidentes, problemas, criação e implantação do ambiente de produção e monitoramento (*Operations*).

De acordo com Muniz, Santos, Irigoyen e Moutinho (2019) o *DevOps* é uma cultura fortemente colaborativa entre as equipes de Desenvolvimento e Operações para

entregar software funcionando em produção de forma ágil, segura e estável. E reforça também, que mais do que um conceito, é uma jornada de aproximação entre as pessoas com ações práticas de automação para acelerar as implantações com qualidade e empatia entre todos os envolvidos.

Nas equipes de TI, cita-se o exemplo desse trabalho no Senac, as equipes de tecnologia estão separadas em diferentes estruturas e departamentos e são zonas de conflitos sobre de quem é a responsabilidade em um problema, onde desenvolvedores alegam problema de infraestrutura e a infraestrutura aponta problemas no desenvolvimento. Com o *DevOps*, munido das melhores práticas e ferramentas, os conflitos tendem a diminuir, visando a empatia pautada nos processos e soluções tecnológicas para melhoria contínua e processos enxutos.

Muniz *et al*. (2019) apontam as ações concretas para disseminar a cultura *DevOps* e promover os quatro pilares para a sua adoção em efetivo:

1. **Colaboração** – Construção de resultado com interações de pessoas com experiências e um propósito em comum.
2. **Afinidade** – Relações interdependentes entre os times com objetivos organizacionais complementares para reforçar o sentimento de empatia entre as equipes.
3. **Ferramentas** – aceleradores e impulsionadores de mudança para mudança da cultura.
4. **Escala** – dimensionamento que leva em conta como os outros três pilares podem ser aplicados.

De acordo com a AWS (2020) alguns conceitos são considerados como melhores práticas para adoção do *DevOps*:

- **Integração Contínua** – Desenvolvedores com frequência juntam as suas alterações de código em um repositório central.
- **Entrega Contínua** – A entrega contínua é uma prática de desenvolvimento de software em que alterações são criadas, testadas e preparadas automaticamente para liberação em produção.
- **Microsserviços** – É uma abordagem de projeto para a criação de um aplicativo único como um conjunto de pequenos serviços. Cada serviço é executado de maneira independente, para funcionalidades especificas e se comunicam geralmente por uma interface de programação de aplicativo (API).
- **Infraestrutura como código** - O modelo controlado por API da nuvem permite que desenvolvedores e administradores de sistema interajam com a infraestrutura de modo programático e em escala, em vez de precisarem instalar e configurar manualmente os recursos. Como são definidos por código, infraestrutura e servidores podem ser implantados rapidamente usando padrões normativos, atualizados com os patches e as versões mais recentes ou duplicados várias vezes.
- **Monitoramento e registro em *log*** - As empresas monitoram métricas e logs para ver como a performance do aplicativo e da infraestrutura afeta a experiência do usuário final do seu produto. Ao capturar, categorizar e analisar dados e logs gerados pelos aplicativos e pela infraestrutura, as empresas compreendem como as alterações ou atualizações afetam os usuários, o que proporciona um esclarecimento sobre as causas raiz dos problemas ou das alterações inesperadas.
- **Comunicação e colaboração** - O aumento da comunicação e da colaboração em uma empresa é um dos principais aspectos culturais do *DevOps*. O uso das ferramentas de *DevOps* e da automação do processo de entrega de software

estabelece a colaboração ao unir fisicamente os fluxos de trabalho e as responsabilidades de desenvolvimento e operações.

As seguintes etapas são passos essenciais e fazem parte do ciclo de vida de criação e desenvolvimento de uma aplicação:

- **Codificação (*Code*): p**ara armazenamento e compartilhamento entre equipes de projetos e códigos, que são armazenados e compartilhados de maneira que possa existir uma base de conhecimento mínima e reaproveitamento de soluções.
- **Compilação (*Build*):** é o processo responsável pela compilação dos códigos da linguagem utilizada para desenvolvimento da aplicação.
- **Teste:** é o teste de carga onde se simula o acesso às funcionalidades das aplicações e serviços.
- ***Release*:** solução para controle de versões de códigos compilados nas aplicações, sendo possível controlar as mudanças e ter o controle das diferentes funcionalidades e versões disponibilizadas.
- ***Deploy***: é o processo de realizar a mudança do código desenvolvido em um repositório que dispara automaticamente o processo de construção (build) para a aplicação.
- **Monitoramento:** consiste em garantir a evolução escalável da rotina das aplicações, seja ela de maneira preditiva ou reativa. Esses monitoramentos ajudam no cumprimento de padrões de tempo de atendimento de incidentes.
- **Operação**: é a correção de comportamentos inadequados que causem problemas de uso ou funcionamento do sistema e quaisquer desvios em relação aos requisitos necessários a execução do sistema. Consiste na maneira de operar a solução no que tange a disponibilidade, provisão de recursos tecnológicos para infraestrutura e desenvolvimento.

4. INTERVENÇÃO PROPOSTA

4.1. Plano de Ação

O plano de ação estabelece a estratégia adotada para diminuição dos projetos utilizando a gestão ágil de projetos e *DevOps* para integração das equipes e projetos, assim como as necessidades tecnológicas para implantação de acordo com os aspectos analisados.

Inicialmente foi necessária a aprovação da gerência do departamento de tecnologia que foi feita mediante uma apresentação para exibir as necessidades e as lacunas existentes entre os times de infraestrutura e desenvolvimento, e foi solicitada a criação de uma equipe que representasse as mudanças estruturais em tecnologia, bem como a definição dos papéis e responsabilidades que se pode cumprir utilizando colaboradores cedidos por diferentes coordenações para criação da equipe. Na demonstração para as equipes de tecnologia, foram demonstrados, também, os conceitos de *DevOps* e ágil assim como a quantidade de projetos que se encontravam em atraso.

Com a aprovação da diretoria, foi necessária uma apresentação para todo o departamento de tecnologia do Senac São Paulo com um plano contendo um *Roadmap* com as principais entregas e planejamento com os macros dos principais pontos a serem cumpridos de maneira institucional.

Um dos principais riscos referentes às mudanças propostas foram a falta de conhecimento das metodologias e tecnologias apresentadas e a possível resistência a mudanças dos colaboradores no processo.

Para financiamento das atividades foram utilizadas somente soluções que já se encontravam disponíveis de maneira gratuita nas comunidades ou parcerias com empresas de software que já possuíam relacionamento com o Senac pela relação contratual ou educacional.

Algumas atividades foram consideradas determinantes para o sucesso do processo, sendo elas:

- **Bloqueio de servidores/aplicações –** Retirada de acesso mediante preenchimento de formulário para otimização de recursos e tempo dos desenvolvedores realizarem as tarefas pertinentes as suas funções
- **Versionamento –** Utilização da solução GIT para versionamento de liberação em produção e criação das camadas de homologação e desenvolvimento.
- **Migração de versões –** Aplicações com soluções desatualizadas e fora da conformidade dos fabricantes serão atualizadas.
- **Saneamento e correções –** Utilização da solução Dynatrace (solução de experiência do usuário e ajustes tecnológicos) para resolução de problemas repetitivos ou estruturais.
- **Administração dos ambientes –** As aplicações passaram a ser de responsabilidade da equipe de Middleware.
- **Esteira de Desenvolvimento –** A esteira de desenvolvimento, que são as etapas de validações e processos de desenvolvimento, foi feita com as ferramentas Jenkins (automação de execução de scripts), GIT (versionamento), SONAR (qualidade de software) e Dynatrace (desempenho, monitoramento e experiência do usuário), visando a alta disponibilidade e liberações constantes em produção com monitoramento proativo das aplicações.
- ***Openshift*/Conteinerização/Microsserviços** – Para melhor disponibilidade dos serviços e nível de automação com infraestrutura baseada em código, diversas aplicações foram migradas para o conceito de *container*/microsserviços que propiciam grande nível de automação e não necessitam de iteração humana e são pontos chave para disponibilidade e um novo formato de esteira de desenvolvimento. Os microsserviços são uma abordagem diferenciada de desenvolvimento de aplicação. Nela o aplicativo é dividido em funcionalidades/serviços tornando a aplicação mais versátil, escalável e funcional. Normalmente, os microsserviços utilizam o conceito de *container*s, que empacota os códigos para facilitar a execução de uma aplicação, ou seja, os *container*s oferecem a possibilidade de tornar os serviços em funcionalidades independentes.

4.2. Projeto Metodologia Ágil e *DevOps*

Na intervenção, inicialmente foi criada por um dos autores deste artigo aplicado, como líder do projeto, uma equipe denominada *Middleware* focada na infraestrutura, com o objetivo de ser uma facilitadora na comunicação com as equipes de desenvolvimento.

O motivo de não se criar uma equipe de infraestrutura com nomes utilizados atualmente como *DevOps*, fez-se necessário para evitar resistência quanto a eventuais mudanças que poderiam acontecer evitando-se ansiedade nas equipes de desenvolvimento, possibilitando que as necessidades e as facilidades que a equipe de infraestrutura propiciariam fossem tratadas de maneira conjunta, com a participação de todos no processo e

que fossem pontos chave para a implantação da cultura Ágil e metodologia *DevOps* (figura 3).

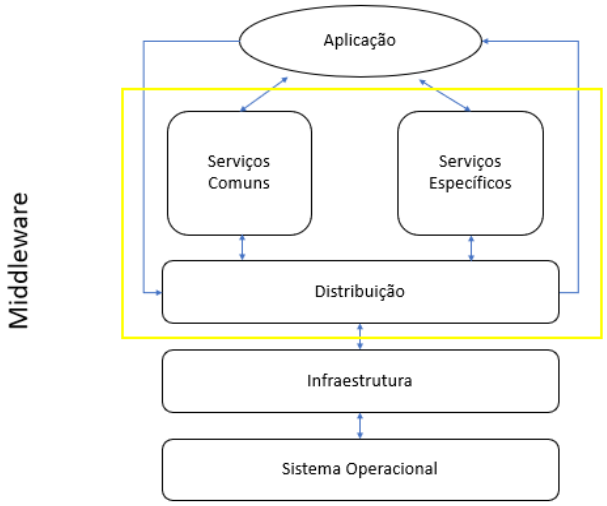

Figura 3: Infraestrutura como Facilitadora entre as Equipes de Desenvolvimento
Fonte: Elaborado pelos autores

Um dos cuidados durante todo o processo de mudanças estruturais e metodológicas, foi com a cultura da empresa, muitos colaboradores com muitos anos de empresa que podiam se sentir afetados pelo processo e por suas possíveis mudanças e consequências.

Todas as etapas a serem cumpridas e um *roadmap* (figura 4) com os objetivos a serem alcançados foram apresentados em uma reunião para toda a Gerência de Tecnologia. Após o término da exposição, muitos colaboradores procuraram um dos autores deste artigo, pois identificavam-se com o processo e estavam ansiosos com tais mudanças, pois precisavam também de ajuda nas tratativas com a equipe de infraestrutura e de algumas lacunas que existiam para o desenvolvimento das soluções. Todas as etapas foram cumpridas e estão disponíveis sejam em melhorias processuais ou tecnológicas.

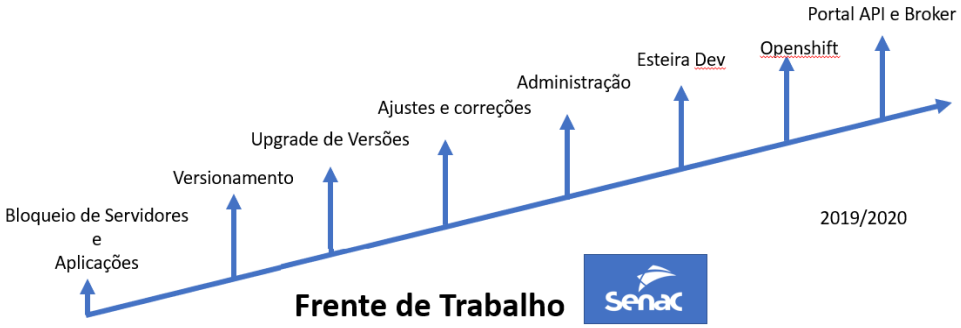

Figura 4: *Roadmap* da equipe *Middleware*
Fonte: Elaborado pelos autores

O passo inicial para implantação da metodologia *DevOps*, foi a retirada de acesso de todos os desenvolvedores aos servidores de produção dos aplicativos, ou seja, não mais poderiam alterá-los quando já em uso ou reiniciar serviços sem o consentimento de todos, assim como realização de testes e homologação nos ambientes produtivos.

Para executar essa tarefa, foram enviados formulários para preenchimento dos desenvolvedores para iniciar as etapas apresentadas no *roadmap* da equipe. Esse passo inicial foi considerado de extrema importância, pois erros repetitivos seriam descobertos e com isso iniciava-se a desoneração de tempo de desenvolvedores em funções que não são de sua responsabilidade eliminando-se, assim, as múltiplas funções.

Na figura 5 verifica-se como *DevOps* do Senac foi configurado, aplicando as ferramentas de tecnologia para elaboração das etapas da metodologia. Um dos autores deste artigo aplicado elaborou e implantou junto a equipe liderada por ele, todos as etapas consideradas necessárias tecnologicamente para o *DevOps*, sendo elas:

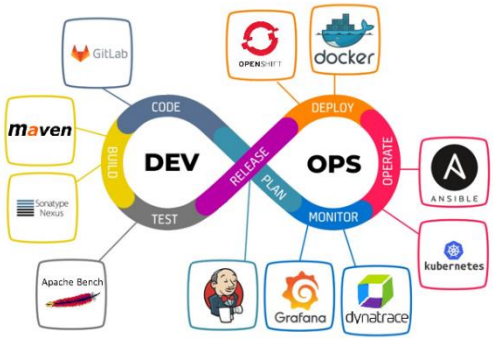

Figura 5: Ferramentas de *DevOps* implantadas na GTI do SENAC
Fonte: Adaptado pelos autores

- **Codificação (*Code*) –** Foi utilizada a solução GitLab que é um gerenciador de repositório de software baseado em git que é um gerenciador de versões de software.
- **Compilação (*Build*) –**Implantou-se a solução Maven que é uma ferramenta de automação de compilação e o Nexus para gerenciar artefatos de software.
- **Teste** – Inicialmente foi utilizado o Apache Bench para testes manuais das aplicações e serviços.
- ***Release* –** Para provisionamento automático de versões (*releases*), foi disponibilizada a ferramenta Jenkins para administração. Por ela, são disparadas rotinas de atualização e versão das aplicações.
- ***Deploy*** – O Openshit e conceito de *container* para aplicações legadas e passíveis de migração. O ganho quanto as tarefas realizadas anteriormente pelos desenvolvedores passam a ser geridas automaticamente, assim como os recursos e escalabilidade das aplicações, que podem ser feitas com as aplicações online e de maneira que conforme a utilização da aplicação, escale mais recursos tecnológicos como memória e processamento, impactando positivamente na otimização, economia e distribuição de recursos na aquisição de servidores. O projeto de criação do Openshift, liderado por um dos autores deste artigo aplicado, atualmente conta com mais de 150 serviços disponibilizados e com disponibilidade de quase 100%.
- **Monitoramento –** A solução Dynatrace foi implantada pela equipe de Middleware e é utilizada para identificação de problemas e análise da experiência do usuário. Com ela é possível analisar as aplicações com uma interface amigável e, também, com um monitoramento proativo verificar possíveis problemas em códigos, segurança e problemas estruturais. Uma das funcionalidades disponibilizadas, possibilita assistir à navegação dos usuários nas aplicações e com ela entender o comportamento dos clientes/usuários para identificar possíveis melhorias ou ajustes tecnológicos. A ferramenta em questão, foi considerada

ponto chave em toda a estrutura de desenvolvimento e metodologia *DevOps*, pois uniu as áreas de negócio e tecnologia com visões estratégicas, além de ser um pacificador nas "guerras" comuns que acontecem nos momentos de problemas entre as equipes de infraestrutura e desenvolvimento. A Solução Grafana é utilizada somente para ajustes e análises tecnológicas da solução Openshift, pois já é contemplada dentro da solução.

- **Operação –** Ansible é uma ferramenta de automação de rotinas de infraestrutura, para gerenciar, automatizar, configurar servidores e implantar aplicativos, por linha de código, solução que embarcada dentro da solução do Openshift/Kubernetes é utilizada para orquestração de *containers*.

Todas as soluções e metodologias aplicadas estavam disponíveis para os colaboradores pela plataforma online para treinamento e certificações.

5. RESULTADOS OBTIDOS

Todas as etapas percorridas, auxiliaram na economia de licenciamento de software, em torno de R$ 2.500,00 por licença, resultando em cerca de R$700.000,00 de economia com a criação de mais de 300 serviços que agora executam em formato de *container* e não fazem uso de um sistema operacional licenciado e na redução do número de incidentes com a retirada de acessos e utilização da ferramenta Dynatrace, pois todos os problemas foram tratados pontualmente.

Outro fator importante é quanto a disponibilidade dos serviços, no modelo de *container* as mudanças são realizadas de forma que não tragam mais indisponibilidade e podem ser feitas de maneira contínua e integrada, agilizando o processo de Gestão de Mudança, não sendo mais necessário intervenção humana e somente focar em novas funcionalidades, assim como uma base de conhecimento criada para compartilhamento de informações e códigos para reaproveitamento de implantações e documentação. Todos os monitoramentos já são nativos no momento da criação dos *containers*, ou seja, não é necessário instalação, configuração de ferramentas de monitoramento manualmente.

Os serviços por estarem sendo executados em formato de *container*, desoneraram a equipe de monitoramento 24x7 e os desenvolvedores com horas de trabalho, pois em caso de falhas ou alto consumo, o ambiente ajusta-se automaticamente, realizando uma conta de homem x hora, é possível constatar que somente com a exoneração de horas de um analista júnior (R$ 9.000,00) e um programador pleno (R$ 7.000,00), foi possível economizar algo em torno de R$ 280.000,00 e mais horas da equipe de monitoramento que até a elaboração deste trabalho ainda não foi calculada, pois propicia-se a possibilidade de redução de colaboradores de missão crítica que atuam durante todos os períodos no monitoramento e procedimentos para aplicações.

No referido trabalho, foram migradas mais de 300 aplicações para esse modelo apresentado, contemplando a estruturação do desenvolvimento e modelo de container e já existe a previsão da migração e criação de mais 300 serviços.

Com as metodologias adotadas para gestão de projetos, aliada ao DevOps para integração das equipes e soluções tecnológicas, foi possível entregar projetos de maneira faseadas para homologação dos demandantes, possibilitando um escopo flexível nos projetos, sendo possível ajustes pontuais e momentâneos nas soluções e com isso auxiliando nos prazos dos projetos e no cumprimento das datas, assim como na aceleração do desenvolvimento dos projetos em atraso.

6. CONTRIBUIÇÃO TECNOLÓGICA-SOCIAL

Neste artigo aplicado foi possível aprofundar-se nas principais metodologias e soluções tecnológicos tanto para com o uso de aplicativos para promover as mudanças, como no que tange a gestão de projetos e integração entre as equipes. As mudanças continuam acontecendo no GTI e Senac São Paulo e a proposta apresentada está sendo utilizada em outros projetos, mostrando resultados surpreendentes, com entregas dentro do prazo combinado e alguns até mesmo antes do período acordado.

A inovação proposta com a mudança no formato de desenvolvimento e na gestão de projetos é um passo muito grande dentro da instituição Senac e propicia inúmeras possibilidades tanto para o fornecimento de soluções para os clientes internos tanto para os externos, com possibilidades de aumento de receita pelo fornecimento de soluções com um grau de maturidade elevado para integrar equipes e soluções tecnológicas.

Durante essa jornada, muitos desafios foram enfrentados e um dos principais foi a rotatividade de funcionários e, apesar do esforço de retenção do Senac SP, houve perda de alguns colaboradores da equipe, devido ao cenário da pandemia e a quantidade de soluções e ofertas pela internet e, também, devido ao fato dos colaboradores obterem um perfil multidisciplinar de soluções e gestão de projetos que são de grande relevância para o mercado de trabalho, sendo compreensível tendo-se em vista a evolução profissional desses colaboradores, uma vez que o Senac SP é uma entidade de capacitação.

Quando se faz a abordagem do ágil e *DevOps*, muitos pensam que elas se conflitam de alguma maneira, mas é possível observar que elas se complementam em pontos cruciais das entregas faseadas, em questões colaborativas entre as equipes e o *DevOps* surge como um facilitador para que essas entregas aconteçam no âmbito tecnológico, apoiada em ferramentas estruturais e definidas para cada etapa de um processo de entrega de software.

Uma sugestão é que as mudanças podem ocorrer, mesmo quando se trata de cenários corporativos mais conservadores. Um dos pontos principais é reforçar a necessidade do engajamento de todos os envolvidos, assim como da equipe que organizará todos essas mudanças, pois até o total entendimento e capacitação das equipes, muitos conflitos e preocupações podem acontecer, mas na verdade até o total entendimento do processo e benefícios propostos podem gerar desconfiança.

Um outro ponto importante que deve ser reforçado reforça é com relação a *containerização,* que se mostrou um grande aliado na automação de rotinas manuais, aumento de disponibilidade das aplicações e controle das mudanças. Quando se tem uma solução dessas para utilização, o caminho para embarcar as soluções que facilitam o dos desenvolvedores é mais simples e desonera o tempo que muitos gastam com a administração dessas aplicações. Um outro ponto positivo é que devido ao fato de as aplicações estarem centralizadas em uma única solução, facilita a gestão e padronização do trabalho de todos os colaboradores.

Quando se trata de projetos ágeis em que entregas faseadas são entregues de acordo com o modelo adotado, não é possível pensar nesse modelo sem pensar em uma mudança estrutural e tecnológica nos times de desenvolvimento de sistemas e nesse trabalho é possível identificar que ambas se complementam e não conflitam nas necessidades, recomendações e solução dos problemas.

REFERÊNCIAS

Agile Manifesto. *Princípios por trás do Manifesto Ágil.* Recuperado em 10 de outubro de 2020 de https://agilemanifesto.org/iso/ptbr/principles.html

AWS (2020). *Amazon - Whats is DevOps* Recuperado em 14 de setembro de 2020 de https://aws.amazon.com/pt/devops/what-is-devops/

Camargo, R. (2019) PM VISUAL: *Gestão de projetos simples e eficaz.* São Paulo: Saraiva

Cohn, M., (2011) *Desenvolvimento de software com Scrum*. Porto Alegre: SAGAH

Fagundes, E.M. Um kit de ferramentas para excelência na Gestão de TI. *Information System Audit & Control.* Recuperado em 04 de abril de 2020 de https://efagundes.com/artigos/cobit/

Highsmith (2012) *What is agility?* Recuperado em 05 de outubro de 2020 *http://jimhighsmith.com/what-is-agility*

Ishikawa, K., (1993) *Controle de qualidade total: à maneira japonesa.* Rio de Janeiro: Campos, 1993.

Kruchten, P. (2008) *Agility situated: contexto does matter, a lot.* The University of British Columbia, XP 2008. Limerick, Irelandjune.

Marcondes, R. C., Miguel, L. A. P.; Franklin, M.A., & Perez, G. (2017). Metodologia para elaboração de trabalhos práticos e aplicados: administração e contabilidade. Recuperado de http://up.mackenzie.br/stricto-sensu/administração-do-desenvolvimento-de-negocios-profissional/.

Muniz, A., Santos, R., Irigoyen, A., & Moutinho, R. (2019) *Jornada DevOps: Unindo Cultura ágil, Lean e tecnologia para entrega de software com qualidade*. Rio de Janeiro: Brasport

Penrose, E.G. (1959) *The theory of the growth of the firm.* New York: Wiley

Senac (2020) Recuperado em 05 de março de 2020 de https://www.senac.br