

DOI: 10.5748/19CONTECSI/PSE/PRM/6980

**CATALOG OF SOFTWARE PROJECT MANAGEMENT APPROACHES USED
IN THE CONTEXT OF AGILE DEVELOPMENT**

**CATÁLOGO DE ABORDAGENS DE GERENCIAMENTO DE PROJETOS DE
SOFTWARE USADAS NO CONTEXTO ÁGIL DE DESENVOLVIMENTO**

Elielton da Costa Carvalho ; <https://orcid.org/0000-0002-9819-535>
UFPA

Sandro Ronaldo Bezerra Oliveira ; <https://orcid.org/0000-0002-8929-5145>
UFPA - Universidade Federal do Pará



CATALOG OF SOFTWARE PROJECT MANAGEMENT APPROACHES USED IN THE CONTEXT OF AGILE DEVELOPMENT

ABSTRACT: Project management approaches have become essential and indispensable in the daily lives of project managers. These solutions have evolved over time and have come to play a prominent role in the agile context of software development. However, these changes over time have brought challenges for organizations in choosing the most appropriate approach to project objectives and needs. Therefore, this work presents a catalog containing the software project management approaches used in the agile context and that were identified from a Systematic Literature Review. Each approach is classified in one of the eight categories listed in this work. With this, it is expected to contribute to the state of the art, project managers and enthusiasts when selecting an approach that best suits their daily needs.

Keywords: Software Project Management; Project Management Approaches; Catalog of Approaches; Agile Development

CATÁLOGO DE ABORDAGENS DE GERENCIAMENTO DE PROJETOS DE SOFTWARE USADAS NO CONTEXTO ÁGIL DE DESENVOLVIMENTO

RESUMO: As abordagens de gerenciamento de projetos se tornaram essenciais e indispensáveis no dia a dia dos gerentes de projetos. Essas soluções passaram por uma evolução ao longo do tempo e passaram a desempenhar papel de destaque no contexto ágil de desenvolvimento de software. No entanto, essas mudanças com o decorrer do tempo trouxe desafios para as organizações ao escolher a abordagem mais adequada para os objetivos e necessidades do projeto. Diante disso, este trabalho apresenta um catálogo contendo as abordagens de gerenciamento de projetos de software usadas no contexto ágil e que foram identificadas a partir de uma Revisão Sistemática da Literatura. Cada abordagem está classificada em uma das oito categorias elencadas neste trabalho. Com isso, espera-se contribuir com o estado da arte, gerentes de projetos e entusiastas na hora de selecionar uma abordagem que melhor se adequem às suas necessidades diárias.

Palavras-chave: Gerência de Projetos de Software; Abordagens de Gerenciamento de Projetos; Catálogo de Abordagens; Desenvolvimento Ágil

Agradecimentos: À CAPES pelo incentivo financeiro pela concessão de bolsa de pesquisa de mestrado. Este trabalho pertence ao projeto SPIDER/UFPA (<http://www.spider.ufpa.br>).

1. INTRODUÇÃO

Os projetos se tornaram ativos primordiais para o sucesso das organizações, independentemente da área em que ela está inserida. Os projetos que são desenvolvidos com sucesso melhoram os lucros das organizações envolvidas e satisfazem os interesses dos clientes. Sobre os projetos de desenvolvimento de software, estes se tornaram a base para um produto de software bem desenvolvido. Nas últimas décadas essa área do conhecimento evoluiu rapidamente se tornou mais estruturada e robusta, essencial para atender as crescentes demandas e melhorias tecnológicas e também um pré-requisito para qualquer empresa que almeja entregar qualidade por meio dos seus produtos (SALEEM et al., 2021).

Hoje os projetos exigem requisitos bem elaborados, isto é, requisitos precisos, um custo bem estimado e um cronograma bem ajustado para garantir o sucesso do desenvolvimento. No entanto, controlar as incertezas de um projeto, principalmente projetos de software, é uma tarefa que exige dos gerentes um aparato de abordagens que o auxiliem nas atividades, seja para comunicação com a equipe, para controlar o cronograma e até mesmo para mitigar os riscos do projeto (HALEEM et al., 2022).

Neste contexto, as abordagens de gerenciamento de projetos ganham papel de destaque, uma vez que se tornaram indispensáveis no dia a dia dos gerentes de projetos. Essas abordagens evoluíram do tradicional para o ágil e global, mudando não só o foco de atuação, mas a forma como elas entregam resultados à equipe de desenvolvimento. Essas mudanças sofridas pelas abordagens se tornaram um desafio para as organizações ao escolher a abordagem mais adequada para os objetivos e necessidades do projeto.

Assim, é possível notar, a literatura ainda tem carência de estudos com foco na compreensão das características das abordagens de gerenciamento de projetos de software nos mais diferentes níveis operacionais (SALEEM et al., 2021). Diante do exposto, este artigo tem como objetivo apresentar uma série de abordagens usadas no gerenciamento de projetos de software, no contexto ágil. Para tal, foi desenvolvido um catálogo com as abordagens extraídas de CARVALHO e OLIVEIRA (2022).

Além desta seção de introdução, este trabalho está estruturado da seguinte forma: a seção 2 apresenta a metodologia do trabalho e a classificação desta pesquisa; a seção 3 apresenta o catálogo das abordagens e, por fim, a seção 4 traz as conclusões deste trabalho.

2. METODOLOGIA

O campo da pesquisa científica é vasto e cada uma possui peculiaridades que a torna única. Então, é natural que as pesquisas necessitem de classificações, pois isso permite uma maior organização e, assim, proporciona um melhor entendimento a seu respeito (GIL, 2017). Mediante ao exposto, esta pesquisa é classificada:

Quanto à natureza: em uma pesquisa aplicada, uma vez que o intuito é gerar conhecimento sobre abordagens de gerenciamento de projetos para serem aplicados na prática;

Quanto à abordagem: em uma pesquisa qualitativa, uma vez que este trabalho apresenta apenas dados qualitativos, ou seja, a descrição das abordagens.

Quanto aos objetivos: em uma pesquisa descritiva, visto que descreve um conjunto de características de abordagens que compõem um campo de estudo científico;

Quanto aos procedimentos técnicos: em uma pesquisa bibliográfica, pois foi elaborada a partir de um trabalho publicado por CARVALHO e OLIVEIRA (2022).

Como mencionado, este trabalho foi desenvolvido a partir de uma Revisão Sistemática da Literatura (RSL) desenvolvida por CARVALHO e OLIVEIRA (2022), onde os autores identificaram um conjunto de abordagens usadas no gerenciamento de projetos de software, no contexto ágil. No entanto, o trabalho desenvolvido pelos autores apresenta as abordagens de

forma mais geral, condensadas em oito categorias distintas: (i) Softwares, (ii) Metodologias, (iii) Métodos, (iv) Modelos, (v) Ferramentas, (vi) Técnicas, (vii) Frameworks e (viii) Práticas.

Este trabalho, por outro lado, traz o detalhamento de cada uma das abordagens identificadas nos artigos selecionados pela RSL. A partir disso, o catálogo foi estruturado em cinco tópicos, da seguinte forma:

Fornecedor / Desenvolvedor: esse tópico apresenta os responsáveis pelo desenvolvimento e/ou manutenção da abordagem, seja pessoa física ou alguma empresa/organização.

Site de acesso: esse tópico traz um link onde o leitor pode consultar mais informações sobre a abordagem, caso tenha interesse.

Descrição: esse tópico apresenta a ferramenta, isto é, na descrição a abordagem e definida de acordo com o que foi identificado nos artigos da RSL, bem como informações adicionais retiradas de outras fontes complementares.

Finalidade: esse tópico traz o leitor o objetivo principal para o qual a abordagem foi concebida.

Como funciona: esse tópico apresenta um pouco mais da descrição da abordagem e como ela pode ser usada pelos interessados.

3. CATÁLOGO

Esta seção apresenta o catálogo de abordagens, conforme foi descrito na seção de metodologia. Devido à limitação de páginas, este trabalho apresenta apenas três abordagens para cada categoria, com exceção da categoria “Práticas” que apresenta apenas duas, pois foi o número total de abordagens identificadas pela RSL. As demais abordagens que não foram descritas neste trabalho podem ser encontradas no seguinte endereço: <https://zenodo.org/record/7226511#.Y1BjcHbMLcd>. As abordagens descritas neste trabalho foram selecionadas com base no número de citações nos trabalhos da RSL.

Importante destacar que, algumas abordagens já são mais difundidas que outras e, portanto, possuem mais informações disponíveis para consulta. Enquanto isso, algumas foram desenvolvidas e apresentadas apenas nos artigos selecionados na RSL. Além disso, vale mencionar que, grande parte dos links disponibilizados para mais informações remetem para o artigo de onde a abordagem foi retirada. Isso acontece, pois a maioria das abordagens foram frutos de pesquisas acadêmicas e que ainda não foram implementadas massivamente nas indústrias de desenvolvimento de software.

3.1. Softwares

3.1.1. T3 Software

Fornecedor / Desenvolvedor: Valerie Trapa e Santhpur Rao

Site de acesso: <https://ieeexplore.ieee.org/document/1667585>

Descrição: A ferramenta T3 é um software leve baseado em JUnit que fornece visibilidade do andamento do projeto para dar suporte ao desenvolvimento rápido. Usando visualizações centradas na história, as equipes ágeis podem avaliar rapidamente o status, determinar dependências e identificar pontos problemáticos. O T3 para relata o status realista do projeto e melhora a comunicação entre os membros da equipe distribuídos geograficamente.

Finalidade: O T3 tem como finalidade abordar questões sistemáticas em uma ampla gama de projetos. Além disso, ele busca auxiliar na fácil identificação da complexidade funcional do projeto e a experiência dos desenvolvedores.

Como funciona: T3 é um software que calcula a porcentagem de métricas completas do projeto. Além disso, fornece uma visão geral, por meio de relatórios, sobre o tempo de implementação de cada uma das histórias de usuários e do tempo total de implementação de todo o projeto, com base nessas histórias de usuários.

3.1.2. MasePlanner

Fornecedor / Desenvolvedor: Roberto Morgan e Frank Maurer

Site de acesso: <https://ieeexplore.ieee.org/document/4031753>

Descrição: MasePlanner é um software de planejamento ágil que oferece suporte para o planejamento baseado em cartão, facilitando a comunicação não verbal. O MasePlanner foi desenvolvido para melhorar o processo de planejamento, fornecendo às equipes um ambiente digital que oferece suporte ao gerenciamento de informações.

Finalidade: O MasePlanner tem como finalidade oferecer suporte a reuniões de planejamento, fornecendo um espaço de trabalho comum semelhante a uma mesa para criar e organizar cartões de história.

Como funciona: O MasePlanner suporta a criação de cartões de história e iterações permitindo sua organização. Ele permite o planejamento baseado em cartão que fornece conhecimento espacial da localização dos cartões. O MasePlanner foi projetado como uma ferramenta de suporte e, como tal, requer que exista um canal de voz de alta qualidade para que ocorra uma verdadeira colaboração. O MasePlanner permite que os artefatos de planejamento sejam criados, editados e organizados de maneira semelhante às reuniões de planejamento em papel.

3.1.3. Sistema de Apoio à Decisão de Planejamento de Sprint (SPESS)

Fornecedor / Desenvolvedor: Alhejab Alhazmi e Shihong Huang

Site de acesso: <https://ieeexplore.ieee.org/document/8479063>

Descrição: O SPESS é um software auxiliar os gestores no planejamento de Sprint. O SPESS é baseado principalmente em três fatores: a competência do desenvolvedor, a antiguidade do desenvolvedor e a dependência da tarefa.

Finalidade: Este software visa atribuir as tarefas de cada Sprint aos desenvolvedores garantindo que cada membro da equipe contribua com o máximo de seu potencial, e o planejamento do projeto seja otimizado para o menor tempo possível.

Como funciona: O SPESS reduz o tempo da sprint em até um dia. Isso se dá pelo fato de o SPESS ser aplicado considerando que todos os desenvolvedores têm a competência necessária para trabalhar em qualquer tarefa. A razão da diferença neste caso particular é devido ao seu método que depende da igualdade do número de tarefas atribuídas a cada desenvolvedor. Além da antiguidade dos desenvolvedores, o SPESS também leva em consideração dois fatores adicionais de planejamento de software - os níveis de competência dos desenvolvedores e a dependência das tarefas. A competência dos desenvolvedores é um dos principais fatores para determinar a quais desenvolvedores adequados são atribuídos as tarefas que correspondem às suas habilidades. O fator de

dependência da tarefa é usado para determinar a ordem em que as tarefas são desenvolvidas. Entre cada iteração no Sprint, o SPESS verifica o status de trabalho dos desenvolvedores e prepara o número de desenvolvedores com os quais trabalhar para a próxima iteração. Como resultado, a ferramenta SPESS atinge seu objetivo principal de que os desenvolvedores trabalhem nas tarefas de acordo com seu nível de habilidade com o tempo de acabamento ideal.

3.2. Metodologias

3.2.1. BloomSoft

Fornecedor / Desenvolvedor: Edgar Castillo-Barrera, Monica Amador-García, Héctor Pérez-González, Francisco Martínez-Pérez e Francisco Torres-Reyes

Site de acesso: <https://ieeexplore.ieee.org/document/8645899>

Descrição: A metodologia consiste em um ajuste da Taxonomia de Bloom, ou seja, em uma mudança de hierarquia do nível de Aplicação, que na Taxonomia de Bloom está no nível 3, mas de acordo com o processo de Desenvolvimento de Software passa para o nível 5. A mudança baseia-se na seguinte análise: para construir, integrar e testar um software é necessário ter realizado previamente uma análise e também uma síntese da informação, que corresponde em Engenharia de Software às fases de Design e Programação, respectivamente.

Finalidade: Permite ter uma forma ágil de classificar a complexidade das histórias de usuário a partir dos verbos nelas identificadas e, ao mesmo tempo, determinar com essa classificação o estágio de Desenvolvimento de Software a que pertence. Desta forma, a equipe de desenvolvimento pode classificar as histórias em etapas e com isso fazer um melhor planejamento dessas histórias a serem realizadas em cada Sprint.

Como funciona: As etapas do processo de Desenvolvimento de Software ou ciclo de vida clássico são: (1) Pré-análise/Planejamento, (2) Análise/Requisitos, (3) Projeto, (4) Desenvolvimento, (5) Testes e (6) Implementação. Considerando e analisando as etapas mencionadas, é feito um ajuste da Taxonomia de Bloom, que consiste em uma mudança de hierarquia do nível de Aplicação, que na Taxonomia de Bloom está no nível 3, mas conforme o processo de Desenvolvimento de Software passa para o nível 5. A mudança baseia-se na seguinte análise: para construir, integrar e testar um programa de Software, é necessário ter feito previamente uma análise e uma síntese da informação, que corresponde em Engenharia de Software às fases de Design e Programação respectivamente. A proposta segue uma analogia com o ciclo de vida do Software, ou seja, o nível de Aplicação no ciclo de vida do Software é ajustado à fase de Teste, gerando um melhor entendimento e permitindo uma determinação da complexidade das histórias de usuários. Para complementar a identificação da complexidade, foram adicionados verbos em cada nível da taxonomia relacionada à área de Engenharia de Software.

3.2.2. DevOps

Fornecedor / Desenvolvedor: John Allspaw e Paul Hammond

Site de acesso: <https://www.redhat.com/pt-br/topics/devops>

Descrição: DevOps vem da junção das palavras Development e Operations que foi mencionado pela primeira vez em 2009 em uma palestra chamada de "10+ Deploys Per Day", por uma dupla que trabalhava no Flickr.

Finalidade: O DevOps permite que funções anteriormente isoladas – desenvolvimento, operações de TI, engenharia da qualidade e segurança – atuem de forma coordenada e colaborativa para gerar produtos melhores e mais confiáveis. Ao adotar uma cultura de DevOps em conjunto com as práticas e ferramentas de DevOps, as equipes ganham a capacidade de responder melhor às necessidades dos clientes, aumentar a confiança nos aplicativos que constroem e cumprir as metas empresariais mais rapidamente.

Como funciona: Com a implementação de um modelo de DevOps, as equipes de desenvolvimento e operações não ficam mais separadas. Às vezes, essas duas equipes são combinadas em uma só. Os engenheiros trabalham durante o ciclo de vida inteiro do aplicativo, da fase de desenvolvimento e testes à fase de implantação e operações, e desenvolvem várias qualificações não limitadas a uma única função. Em alguns modelos de DevOps, as equipes de controle de qualidade e segurança também podem aumentar sua integração com o desenvolvimento, as operações e todo o ciclo de vida dos aplicativos. Quando a segurança é a prioridade de todos em uma equipe de DevOps, isso é denominado às vezes DevSecOps. Essas equipes usam práticas para automatizar processos que historicamente sempre foram manuais e lentos. Eles usam uma pilha de tecnologia e ferramentas que os ajudam a operar e desenvolver aplicativos de modo rápido e confiável. Essas ferramentas também ajudam os engenheiros a realizar tarefas independentemente (por exemplo, implantação de código ou provisionamento de infraestrutura) que normalmente exigiriam a ajuda de outras equipes, e isso aumenta ainda mais a velocidade da equipe.

3.2.3. Scrum

Fornecedor / Desenvolvedor: Scrum.org / Jeff Sutherland

Site de acesso: <https://www.scrum.org/>

Descrição: O Scrum é uma estrutura que ajuda as equipes a trabalharem juntas. Semelhante a uma equipe de rugby (de onde vem o nome) treinando para o grande jogo, o Scrum estimula as equipes a aprenderem com as experiências, a se organizarem enquanto resolvem um problema e a refletirem sobre os êxitos e fracassos para melhorarem sempre.

Finalidade: O Scrum tem como finalidade ser uma estrutura leve que ajuda pessoas, equipes e organizações a gerar valor por meio de soluções adaptáveis para problemas complexos.

Como funciona: A metodologia Scrum propõe que um projeto seja dividido em diversos (pequenos) ciclos de atividades, chamados de sprint, com reuniões frequentes para que a equipe possa alinhar o que vem fazendo e pensar formas de melhorar o processo com agilidade. Essa metodologia propõe que o projeto seja acompanhado sempre bem de perto e passe por mudanças de planejamento o tempo todo, de forma livre e pouco engessada.

3.3. Métodos

3.3.1. Planning Poker

Fornecedor / Desenvolvedor: Mike Cohn

Site de acesso: <https://www.planningpoker.com/>

Descrição: O Planning Poker garante que todos os desenvolvedores participem do processo de estimativa e que a opinião de todos seja ouvida, independentemente de estarem entre as pessoas mais influentes do grupo. Também pode reduzir o problema potencial da chamada ancoragem. Ancoragem é o impacto que alguém que apresenta uma estimativa ou expectativa de custo tem nas estimativas subsequentes, mesmo que a estimativa/expectativa seja irrealista.

Finalidade: A finalidade do Planning Poker é estimar de forma mais rápida e consensual um conjunto de tarefas que precisam ser desenvolvidas em uma sprint.

Como funciona: O Planning Poker consiste na obtenção da estimativa através de um jogo de cartas, o que faz com que todos os membros que fazem parte da equipe participem da estimativa, colocando a sua visão de complexidade, levando em consideração variáveis como tempo e o esforço para pontuar um cartão e após isso chegar a um denominador comum na equipe através de consenso. O Poker no nome vem por conta de as estimativas serem feitas com um baralho. Não um baralho comum, mas um que usa a Sequência de Fibonacci.

3.3.2. COSMIC

Fornecedor / Desenvolvedor: Jean-Marc Desharnais, Luigi Buglione e Buğra Kocatürk

Site de acesso: <https://dl.acm.org/doi/abs/10.1145/2181101.2181117>

Descrição: O COSMIC é um método padronizado de medição do tamanho funcional do software, seja qual for o seu domínio de aplicação. O método COSMIC também foi aceito pela ISO / IEC JTC1 / SC7 como um padrão internacional (ISO / IEC 19761).

Finalidade: O COSMIC tem como finalidade reduzir o risco de o projeto sair do escopo definido, controlar o custo e a qualidade do produto desenvolvido e ajudar na conclusão oportuna do projeto.

Como funciona: O COSMIC pode ser aplicado desde as fases iniciais, necessitando, pelo menos, de requisitos de alto nível a serem refinados durante o ciclo de vida do projeto. No processo de medição, os movimentos de dados são contados, os quais podem ser definidos como “componentes funcionais básicos que movem um único tipo de grupo de dados”. Existem 4 subtipos de movimentos de dados: Entry (E), eXit (X), Read (R) e Write (W). A soma das movimentações de dados dentro do escopo de medição estabelecido retorna o número de Pontos de Função COSMIC (CFP).

3.3.3. FASTDash

Fornecedor / Desenvolvedor: Jacob T. Biehl, Mary Czerwinski, Greg Smith e George G. Robertson

Site de acesso: <https://dl.acm.org/doi/10.1145/1240624.1240823>

Descrição: FASTDash é uma nova visualização interativa que busca melhorar a percepção das atividades da equipe usando uma representação espacial da base de código compartilhada que destaca as atividades atuais dos membros da equipe.

Finalidade: O FASTDash visa contribuir com um sistema de relatórios de conscientização subjacente que pode ser usado para suportar uma infinidade de ambientes de programação, sistemas de repositório e futuras ferramentas de visualização; um novo esquema de codificação de observação que pode ser usado de forma mais ampla para estudos de comportamento de grupo em espaços de trabalho compartilhados.

Como funciona: Um desenvolvedor pode determinar rapidamente quais membros da equipe têm arquivos de origem verificados, quais arquivos estão sendo visualizados e quais métodos e classes estão sendo alterados no momento. A visualização pode ser anotada, permitindo que os programadores complementem as informações da atividade com detalhes de status adicionais. Ele fornece conhecimento imediato de possíveis situações de conflito, como dois programadores editando o mesmo arquivo de origem. O FASTDash foi desenvolvido por meio de design centrado no usuário, incluindo pesquisas, entrevistas em equipe e observação *in situ*.

3.4. Modelos

3.4.1. Blueprint

Fornecedor / Desenvolvedor: Cristiano P. Godoy, Lanier M. Santos, Andre F. Cruz, Rafael S. Zerbini, Elisangela P. Silva e Cícero A. L. Pahins

Site de acesso: <https://dl.acm.org/doi/10.1109/ICGSE.2019.00014>

Descrição: O Blueprint é um modelo de desenvolvimento de software baseada nas metodologias ágeis Scrum e Kanban e adaptada à tendência atual do ambiente global de software. O modelo Blueprint é uma ferramenta inspirada em Scrum e Kanban que gerencia entidades de negócios do projeto antes de seu desenvolvimento. O Blueprint apresenta adaptações importantes para Scrum e Kanban para reduzir burocracias indesejadas e para facilitar o desenvolvimento de software global.

Finalidade: O modelo Blueprint propõe um gerenciamento de projeto leve que é combinado com uma organização de equipe para incentivar e facilitar a comunicação entre equipes em locais diferentes.

Como funciona: O Blueprint propõe uma hierarquia de equipe que pode se adaptar melhor às mudanças nos requisitos de software durante a execução do projeto. O Blueprint define sua hierarquia de equipe como subequipes. O número de membros por subequipe não é definido pelo Blueprint, mas como uma analogia ao Scrum, subequipes devem se concentrar em pequenos grupos. Além disso, o número de subequipes pode mudar durante a execução de um projeto. O Blueprint segue a noção de que o número de membros não representa qualidade ou produtividade. No modelo Blueprint, cada subequipe tem um líder definido como um aspecto da execução do trabalho com responsabilidade. Cada líder de subequipe é responsável pelas atividades e resultados de seu grupo. Ele também é responsável por integrar as tarefas do grupo ao projeto, coordenando esse processo com os demais líderes. Todos os líderes da subequipe respondem ao líder principal.

3.4.2. Rugby

Fornecedor / Desenvolvedor: Stephan Krusche, Lucas Alperowitz, Bernd Bruegge e Martin O. Wagner

Site de acesso: <https://dl.acm.org/doi/10.1145/2593812.2593818>

Descrição: O Rugby é um modelo de processo ágil que inclui fluxos de trabalho para a entrega contínua de software. Ele permite que os funcionários de meio período trabalhem em uma organização baseada em projetos com vários projetos para a entrega rápida de protótipos e produtos. O Rugby foi projetado para ser usado em organizações baseadas em projetos com vários projetos.

Finalidade: O Rugby apresenta uma metodologia leve para desenvolver e lançar protótipos rápidos e aprender com comentários de feedback em ciclos paralelos rápidos.

Como funciona: O Rugby é composto por até oito desenvolvedores, um líder de equipe e um líder de projeto. A equipe do projeto é auto-organizada, multifuncional e, portanto, responsável por todos os aspectos de desenvolvimento e entrega de software. O líder do projeto e o líder da equipe cumprem um papel semelhante ao de um Scrum master enquanto estão em um relacionamento de mestre-aprendiz. Enquanto o gerente de projeto já é experiente, o líder da equipe é um desenvolvedor experiente. Assim, ele está familiarizado com a infraestrutura e os aspectos organizacionais do Rugby. Uma tarefa do líder da equipe é organizar a primeira reunião da equipe e garantir que a equipe organize todas as reuniões seguintes de forma estruturada.

3.4.3. TaskAllocator

Fornecedor / Desenvolvedor: Saad Shafiq, Atif Mashkoor, Christoph Mayr-Dorn e Alexander Egyed

Site de acesso: <https://ieeexplore.ieee.org/document/9461028>

Descrição: O TaskAllocator é um modelo que aproveita a arquitetura LSTM (long short-term memory), uma variante da rede neural recorrente artificial (RNN) e é frequentemente usada para problemas de classificação de texto.

Finalidade: O TaskAllocator tem como finalidade ajudar os gerentes de projeto a alocar a tarefa recém-adicionada ou modificada para a função mais adequada na equipe.

Como funciona: O TaskAllocator aprende com os recursos textuais das alocações de tarefas anteriores e prevê a função aproximada das tarefas recebidas. Uma das características distintivas do TaskAllocator é sua arquitetura flexível. Atualmente, o TaskAllocator está explorando a memória de curto prazo (LSTM) para previsões.

3.5. Ferramentas

3.5.1. aWall

Fornecedor / Desenvolvedor: Magdalena Mateescu, Martin Kropp, Roger Burkhard, Carmen Zahn e Dario Vischi

Site de acesso: <https://dl.acm.org/doi/10.1145/2817721.2835072>

Descrição: Uma ferramenta ágil de colaboração em equipe para grandes sistemas de parede multitoque. O aWall foi projetado com base em pesquisas empíricas de usuários usando novos conceitos de interação e visualização para apoiar e promover o estilo de trabalho ágil altamente colaborativo e comunicativo. O aplicativo é baseado em tecnologia web e pode ser usado tanto em ambiente colocalizado quanto distribuído. Os protótipos implementados foram validados com usuários finais em uma oficina de usuários. Na demonstração, os usuários podem experimentar os conceitos de interação e visualização na prática.

Finalidade: O objetivo é propiciar um processo ágil de intensa interação, colaboração e comunicação aberta constante entre todos os membros da equipe.

Como funciona: O aWall oferece suporte à colaboração ágil da equipe e promove o estilo ágil de trabalho por meio de um sistema de parede multitoque extragrande. O aWall tira proveito da resolução 4K e da tecnologia web e com isso supera as possibilidades das ferramentas de desktop existentes. A ferramenta diferencia três aspectos importantes para ferramentas de colaboração ágil: a necessidade de ferramentas para sustentar a atividade principal da equipe ágil (visão de ação), as necessidades de comunicação da equipe ágil e, em terceiro lugar, suas necessidades de atualizar informações que podem ser distribuídas entre os membros da equipe e artefatos (visão de informações). As visualizações de ações foram cuidadosamente projetadas para sustentar a maioria das interações que precisam ser realizadas durante uma reunião específica. A visualização de informações fornece informações extras necessárias para uma reunião e, com isso, contribui indiretamente para o alcance da meta. As visualizações de informações representam a memória transacional da equipe. Qualquer sistema de memória precisa permitir mudanças. Portanto, as informações representadas nessas visualizações podem ser manipuladas a qualquer momento.

3.5.2. Quadro de Equipes Virtuais

Fornecedor / Desenvolvedor: Olga Liskin e Kurt Schneider

Site de acesso: <https://ieeexplore.ieee.org/document/6337316>

Descrição: Um quadro de equipes virtuais é uma ferramenta para visualizar a estrutura entre histórias de usuários. Abandonar estruturas fixas para representar histórias do usuário dá aos usuários liberdade para especificar os atributos do cartão da história por si próprios. A posição de um cartão no quadro branco ou em relação a outros cartões, bem como muitos outros atributos, pode ter um significado e, portanto, fornecer informações adicionais que estendem o conteúdo puro.

Finalidade: O quadro de equipes virtuais tem como finalidade proporcionar maior visibilidade do status do projeto e das tarefas para equipes que trabalham geograficamente distribuídas.

Como funciona: Inicialmente, pode ser implementado um catálogo de atributos visuais e uma ferramenta de quadro de equipe que ofereça suporte a esses atributos. Após isso, uma avaliação pode ser conduzida para verificar o benefício e o esforço de usar a abordagem do quadro de equipe para diferentes tarefas e a comparamos com ferramentas baseadas em tabelas.

3.5.3. ASAP (*Agile SoftwAre Planning Tool*)

Fornecedor / Desenvolvedor: Uffe Kock Wiil e Rasmus Rosenqvist Petersen

Site de acesso: <https://dl.acm.org/doi/abs/10.1145/1379092.1379101>

Descrição: Um Espaço no ASAP é uma grande superfície 2D usada para organizar os elementos de informação de acordo com seus relacionamentos. O Cartão de Tarefas é o elemento básico de informação usado no ASAP. Ele representa o equivalente em papel com uma grade na qual cada célula recebe um significado específico. O valor das células pode ser alterado pelo usuário. Os elementos de informação são conhecidos em todos os domínios de estruturação de hipertexto. Os Cartões de Tarefas são criados com um gesto de arrastar dentro do Espaço. Um cartão de tarefas é excluído ao pressionar a “cruz vermelha” no canto superior direito.

Finalidade: A primeira versão do ASAP mostrou que uma seleção cuidadosamente escolhida (combinação) de conceitos existentes de vários domínios de estruturação de hipertexto pode fornecer suporte para a gestão do conhecimento complexo tarefa que está planejando.

O novo conceito de Separador foi introduzido como um novo mecanismo de estruturação. O Separador é principalmente interessante em relação ao domínio de estruturação espacial ao trabalhar com certos tipos de tarefas de gestão do conhecimento (como planejamento de projetos).

Como funciona: O ASAP substitui os cartões de tarefas em papel por cartões eletrônicos e fornece uma grande superfície de tela para se parecer com a superfície da mesa. A configuração típica de implantação para ASAP é um SMART Board. A atividade de planejamento Blitz é composta de dez etapas gerais.

Etapa 1: Reúna os participantes. Os planejadores podem digitar seus nomes o mais rápido possível. Os nomes são usados ao distribuir tarefas entre os membros da equipe e são visíveis na Visualização de Tarefas Distribuídas.

Etapa 2: faça um *brainstorming* das tarefas. Os planejadores podem criar novos cartões de tarefas com um simples gesto. Os planejadores podem digitar os nomes das tarefas recém-criadas na célula TASK.

Etapa 3: Organize as tarefas. Os planejadores podem organizar os cartões de tarefas na superfície 2D do ASAP, posicionando-os ao lado dos cartões de tarefas relacionados.

Etapa 4: Revise as tarefas. Os planejadores têm uma visão geral fácil de compreender dos Cartões de Tarefas dispostos como resultado da Etapa 3. Os planejadores podem reorganizar os Cartões de Tarefas nesta etapa da mesma maneira que na Etapa 3, se assim o desejarem.

Etapa 5: Estime e marque as tarefas. Os planejadores podem inserir estimativas para cada tarefa (digitando a estimativa na célula TEMPO). Os planejadores podem atribuir tarefas aos membros da equipe (digitando o nome da pessoa na célula PESSOA).

Etapa 6: Classifique as tarefas. Os planejadores podem classificar as tarefas em iterações sequenciais usando os separadores, colocando os cartões de tarefas em relação aos separadores. Os planejadores podem ter uma visão geral das iterações usando o plano de projeto gerado automaticamente.

Etapa 7: marque o esqueleto ambulante, a versão mais antiga e a receita mais antiga. Os planejadores podem usar Separadores para indicar marcos importantes do projeto, bem como esqueleto ambulante, versão inicial e receita inicial.

Etapa 8: Identifique outras versões. Os planejadores podem adicionar separadores adicionais para dar suporte a esta etapa.

Etapa 9: otimize o plano para se adequar às prioridades do projeto. Os planejadores podem mover os cartões de tarefas para melhor corresponder às prioridades do projeto.

Etapa 10: Capture a saída. Atualmente, existem duas opções para capturar a saída. Os planejadores podem armazenar o plano do projeto, permitindo que ele seja recuperado e revisado posteriormente. Os planejadores podem exportar o projeto em formato XML para ser usado por ferramentas de gerenciamento de projetos de terceiros (se desejado).

3.6. Técnicas

3.6.1. FLOW Mapping

Fornecedor / Desenvolvedor: Kai Stapel, Eric Knauss, Kurt Schneider e Nico Zazworka

Site de acesso: <https://ieeexplore.ieee.org/abstract/document/6063167>

Descrição: Um FLOW Mapping é uma técnica FLOW especial, ou seja, visualização dos participantes do projeto, documentos e fluxos de informações, estendido por recursos para melhorar a conscientização em equipes distribuídas.

Finalidade: FLOW Mapping é uma técnica usada para planejar e orientar a comunicação em projetos de desenvolvimento distribuído de software.

Como funciona: O FLOW Mapping funciona com metáfora do estado da informação para distinguir entre informação sólida, ou seja, acessível a longo prazo, acessível repetível e compreensível por terceiros, ou fluida, ou seja, informação que viola um dos critérios sólidos. Representantes típicos de informações sólidas são documentos formais ou e-mails formais. A informação fluida, por outro lado, é o conhecimento na mente das pessoas, ou mídia que depende fortemente do conhecimento das pessoas para uma interpretação correta, como notas informais, e-mails informais ou o conteúdo de conversas, por exemplo, por telefone.

3.6.2. OPENTCQ

Fornecedor / Desenvolvedor: Jamshaid Iqbal Janjua, Aatka Ali e Muhammad Umar Chaudhry

Site de acesso: <https://dl.acm.org/doi/10.1145/2967878.2967920>

Descrição: A maioria dos modelos e técnicas para o gerenciamento de mudanças compartilham uma estratégia comum, e o processo ou análise de cada iteração é ignorado, por isso encontrou a falha para incorporar a mudança. O formulário de solicitação de mudança (CRF) é documentado para cada vez que o requisito de mudança surge. Uma estratégia sólida de estimativa de requisitos está faltando para lidar com a mudança. O OPENTCQ, processa cada iteração antes de sua integração ao produto como um todo. Essa técnica é, na verdade, customizada a partir do Unified Process (UP) e do Agile Model.

Finalidade: A OPENTCQ se propõe a dimensionar o modelo tradicional para atingir o nível de agilidade.

Como funciona: A OPENTCQ permite ter uma forma ágil de classificar a complexidade das histórias de usuário a partir dos verbos nelas identificadas e, ao mesmo tempo, determinar com essa classificação o estágio de Desenvolvimento de Software a que pertence. Desta forma, a equipe de desenvolvimento pode classificar as histórias em etapas e com isso fazer um melhor planejamento dessas histórias a serem realizadas para cada Sprint.

3.6.3. Arquiteturas de Software Distribuídas Mistas

Fornecedor / Desenvolvedor: Jorge Valenzuela Posadas

Site de acesso: <https://ieeexplore.ieee.org/document/8079698>

Descrição: A arquiteturas mista são usados para atender as reais exigências do governo, porém quando há picos de demanda em prazos específicos, a abordagem de resposta de microsserviços melhora a economia de custos de infraestrutura e responde de forma mais ágil por sua escalabilidade granular.

Finalidade: A arquitetura mista permite trazer e compartilhar informações em escala massiva de lugares remotos onde realizam projetos sócio-produtivos em diversos setores onde existe pobreza e pobreza extrema.

Como funciona: Para implementar ambas as arquiteturas, usamos o . Net framework, Selecionamos ASP.Net Web API para o desenvolvimento de microsserviços, pois oferece a opção de deploy em On-Premise Server e depois, a opção de deploy em Cloud migrando para ASP.Net Core ou utilizando framework de microsserviço puro Azure Service Fabric, sendo a última opção a melhor maneira de implantar em Cloud. Para aplicação monolítica, selecionamos ASP.Net MVC porque permitem criar uma base de código tradicional e usar serviços WCF (baseado em XML) por meio de contratos de serviço de proxy. Ambos os aplicativos foram implantados em servidores IIS e a arquitetura de microsserviços foi implantada nos Serviços de Nuvem do Azure para este artigo. MySQL e Oracle foram usados como banco de dados relacional na arquitetura de microsserviços e apenas Oracle foi usado na arquitetura monolítica.

3.7. Frameworks

3.7.1. Eagle

Fornecedor / Desenvolvedor: Alejandro Guerreiro, Rafael Fresno, An Ju, Armando Fox, Pablo Fernández, Carlos Müller e A. Ruiz-Cortés

Site de acesso: <https://dl.acm.org/doi/abs/10.1145/3338906.3341181>

Descrição: A abordagem permite um esclarecimento ao tornar o status, os desafios e o progresso do projeto transparentes para todos os participantes a qualquer momento. Recursos podem ser atribuídos aos problemas que causam estagnação na conformidade métrica. Além disso, os pesquisadores se concentraram em desafios relevantes, resolvendo o caso de negócios. O foco nos valores de negócios por meio de métricas reduziu essencialmente o risco de falha.

Finalidade: O Eagle tem como finalidade dar maior transparência para as atividades do projeto.

Como funciona: A fim de rastrear a aderência de uma equipe às diferentes práticas, desenvolvemos um painel interativo que é gerado automaticamente a partir do TPA4 que tanto os membros da equipe quanto os gerentes de projeto podem usar. O painel permite selecionar um período de tempo para analisar, por exemplo, uma iteração que foi insatisfatória para detectar as práticas que foram violadas. Além disso, uma anotação sobre o gráfico é possível para destacar comentários persistentes ao longo da linha do tempo para análise posterior. Além disso, cada gráfico está vinculado a uma tabela complementar acima dele com o grau de cumprimento dos TPs para cada dia; caso o objetivo não seja cumprido é destacado com fundo vermelho e com fundo verde caso contrário. Caso sejam necessárias mais informações, os usuários podem obter uma lista de evidências que suportam esse grau de cumprimento.

3.7.2. Framework de Gerenciamento de Projetos para o Desenvolvimento Global de Software

Fornecedor / Desenvolvedor: Ritu Jain, Ugrasen Suman

Site de acesso: <https://dl.acm.org/doi/10.1145/3178315.3178329>

Descrição: Esse framework assimila as áreas de conhecimento do PMBOK com as áreas de conhecimento necessárias para uma gestão eficaz do GSD. Ele orientaria o gerente de projetos da GSD sobre os aspectos a serem considerados na execução de projetos distribuídos. Essa estrutura também atuaria como uma linha de base para pesquisadores para investigações adicionais no domínio de gerenciamento de projetos GSD.

Finalidade: Esse framework tem como finalidade proporcionar um ambiente de desenvolvimento mais coerente com a realidade da equipe e mais robusto para desenvolver projetos atuais.

Como funciona: A estrutura proposta abrange gerenciamento de viabilidade e risco, gerenciamento de equipe virtual, gerenciamento de conhecimento, gerenciamento de escopo e recursos, gerenciamento de desempenho e gerenciamento de integração GSD. O framework garante uma operação bem-sucedida de projetos GSD, pois auxilia no gerenciamento de equipe virtual robusto, garantindo coesão de equipe, configuração de equipe, motivação, alocação de tarefas e gerenciamento de conflitos.

3.7.3. Framework de Medição de Agilidade para Gerenciamento de Projetos de Software

Fornecedor / Desenvolvedor: Mourad Mounir, Akram Salah, Amr Kamel e Hanan Moussa

Site de acesso: <https://dl.acm.org/doi/abs/10.1145/3436829.3436853>

Descrição: A principal noção dessa abordagem é que para que um processo seja verdadeiramente ágil, os valores ágeis devem ser alcançados independentemente de quais práticas ágeis são usadas e sem ditado para as escolhas das práticas. Concentrar-se apenas nos valores ágeis também torna muito simples e fácil conduzir a avaliação sem seguir uma abordagem complexa. Isso apoiará a ideia de fazer a avaliação de agilidade mais de uma vez em diferentes períodos apenas para garantir que haja uma melhoria na agilidade ao longo do cronograma do projeto.

Finalidade: Esse framework se concentra em medir a eficácia do Agile e a medição da agilidade é incluída para garantir que a causa e o efeito do Agile sejam medidos corretamente no desenvolvimento de sistemas críticos.

Como funciona: Esse framework de medição não combina todas as abordagens já usadas para medir a agilidade do processo ou a eficácia do Agile em sistemas críticos. Essa abordagem de medição de agilidade vai depender da medição da conquista dos valores culturais Agile pelo processo atual adotado. Os princípios e práticas ágeis adotados têm seu efeito no alcance dos valores ágeis, porém recomenda-se a liberdade de escolha das práticas a serem adotadas, pois cada projeto ou organização possui suas próprias restrições, ambiente e ferramentas específicas. A principal noção dessa abordagem é que para que um processo seja verdadeiramente ágil, os valores ágeis devem ser alcançados independentemente de quais práticas ágeis são usadas e sem ditado para as escolhas das práticas. Concentrar-se apenas nos valores ágeis também torna muito simples e fácil conduzir a avaliação sem seguir uma abordagem complexa. Isso apoiará a ideia de fazer a avaliação de agilidade mais de uma vez em diferentes períodos apenas para garantir que haja uma melhoria na agilidade ao longo do cronograma do projeto.

3.8. Práticas

3.8.1. MEDIATION - Colaboração de Pesquisa Orientada por Métricas

Fornecedor / Desenvolvedor: Marc Schreiber, Bodo Kraft e Albert Zündorf

Site de acesso: <https://ieeexplore.ieee.org/document/7964364>

Descrição: A abordagem é chamada Metrics Driven Research Collaboration (MEDIATION). De acordo com o nome, a definição de métricas é a tarefa mais importante em uma colaboração de pesquisa; a definição de métricas traz definições de objetivos comuns em foco. Cada métrica está vinculada aos requisitos globais do projeto. A colaboração de pesquisa deve elaborar um consenso sobre um pequeno número de métricas de alto nível que colocam os requisitos mais importantes em foco. A princípio, todos os participantes devem escolher um requisito e definir uma métrica correspondente por completo. Esta métrica deve ter as seguintes propriedades. A métrica deve: representar uma meta comum de negócios de alto nível, ser avaliado automaticamente, corresponder ao status do projeto, ter um limite inicial e, ter um rico conjunto de dados de entrada e saída de exemplo.

Finalidade: A abordagem tem como finalidade o esclarecimento da equipe ao tornar o status, os desafios e o progresso do projeto transparentes para todos os participantes a qualquer momento.

Como funciona: A abordagem garante continuamente que todos os participantes do projeto tenham um objetivo comum: o sucesso do projeto. Depois de aplicar o MEDIATION o desenvolvimento e o progresso tornaram-se mais eficazes. Recursos podem ser atribuídos aos problemas que causam estagnação na conformidade métrica. Todos os participantes foram convencidos de que o produto de software desenvolvido aumenta o valor do negócio. Além disso, os pesquisadores se concentraram em desafios relevantes, resolvendo o caso de negócios. O foco nos valores de negócios por meio de métricas reduziu essencialmente o risco de falha. Por meio de nossa abordagem MEDIATION, a equipe do projeto concentra-se nos requisitos mais importantes e

verifica continuamente se o produto de software está em conformidade com as métricas correspondentes.

3.8.2. Fireteams

Fornecedor / Desenvolvedor: He Zhang, Huang Huang, Dong Shao e Xin Huang

Site de acesso: <https://dl.acm.org/doi/10.1145/3368089.3417068>

Descrição: A fim de reduzir as despesas gerais de gerenciamento de projetos e melhorar a produtividade, uma empresa global de tecnologia da informação e comunicação institucionaliza uma prática de equipe pequena em toda a organização, chamados Fireteams, para resolver os problemas decorrentes de aspectos humanos e sociais, como amizade, talento, habilidade e comunicações.

Finalidade: Os Fireteams tem como objetivo gerar menos sobrecarga de comunicação e gerenciamento, maior agilidade e simultaneidade e melhor habilidade pessoal, com equipes reduzidas.

Como funciona: Os Fireteams funcionam a partir da divisão de equipes em equipes menores (geralmente 4 a 5 pessoas), o que garante maior controle para os gerentes, menos despesas gerais de comunicação, maior agilidade e simultaneidade e capacidade pessoal aprimorada e, conseqüentemente, aumentam a produtividade dos esquadrões.

4. CONCLUSÃO

O gerenciamento de projetos de software desempenha um papel importante dentro da área de TI, pois possibilita o desenvolvimento de produtos de software com alto grau de qualidade e dentro do prazo e orçamento planejados. Porém, para que isso ocorra efetivamente, os gerentes devem ter ao seu lado abordagens que garantam o sucesso do projeto. Portanto, identificar as abordagens que são utilizadas pelos gerentes de projetos de software e entender como e onde elas estão sendo aplicadas torna-se uma pesquisa relevante, uma vez que dá possibilidade dos demais profissionais conhecerem o que está sendo empregado na área.

A principal contribuição deste trabalho foi detalhar as características das abordagens de gerenciamento utilizados por gerentes de projetos de software a partir a RSL. Verificou-se, com isso, que a maioria das abordagens estão direcionadas para o gerenciamento da comunicação, do cronograma e da qualidade dos projetos. Além disso, as abordagens que possuem o maior número de soluções são os softwares, metodologias e métodos, respectivamente. Com isso, nota-se que as abordagens de gerenciamento seguem a tendência da informatização, onde hoje os serviços mais essenciais são realizados com auxílio de algum tipo de software.

Os resultados deste trabalho podem ser úteis para que organizações e profissionais que atuam no mercado de desenvolvimento de software conheçam as principais abordagens de gerenciamento de projetos de software, bem como a forma como eles podem ser utilizados. Também podem ajudar os desenvolvedores e pesquisadores na definição de requisitos que possam atualizar, melhorar e até criar novas soluções com base no que foi apresentado neste trabalho.

REFERÊNCIAS BIBLIOGRÁFICAS

CARVALHO, Elielton; OLIVEIRA, Sandro. The Diversity of Approaches to Support Software Project Management in the Agile Context: Trends, Comparisons and Gaps. In: 17TH INTERNATIONAL CONFERENCE ON EVALUATION OF NOVEL APPROACHES TO SOFTWARE ENGINEERING, 2022, Online Streaming, --- Select a Country ---. 17th

International Conference on Evaluation of Novel Approaches to Software Engineering. [S. l.]: SCITEPRESS - Science and Technology Publications, 2022. Disponível em: <https://doi.org/10.5220/0011063500003176>. Acesso em: 19 out. 2022.

GIL, Antônio Carlos.. Como elaborar projetos de pesquisa. 6. ed. São Paulo: Atlas, 2017.

HALEEM, Mohd; FAROOQUI, Md Faizan; FAISAL, Md. Tackling Requirements Uncertainty in Software Projects: A Cognitive Approach. International Journal of Cognitive Computing in Engineering, v. 2, p. 180-190, jun. 2021. Disponível em: <https://doi.org/10.1016/j.ijcce.2021.10.003>. Acesso em: 19 out. 2022.

SALEEM, Nazish; MATHRANI, Sanjay; SAJJAD, Aymen. Exploring the Evolutionary Characteristics of Project Management Approaches at Different Levels of Operations. In: 2021 IEEE ASIA-PACIFIC CONFERENCE ON COMPUTER SCIENCE AND DATA ENGINEERING (CSDE), 2021, Brisbane, Australia. 2021 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE). [S. l.]: IEEE, 2021. Disponível em: <https://doi.org/10.1109/csde53843.2021.9718471>. Acesso em: 19 out. 2022.

SHENHAR, Aaron J.; DVIR, Dov. Reinventing Project Management: The Diamond Approach to Successful Growth & Innovation. [S. l.]: Harvard Business School Press, 2007. 276 p. ISBN 9781591398004.