RICAKIANDA
REVISTA DE INOVAÇÃO E INVESTIGAÇÃO CIENTÍFICA DA UNIVERSIDADE DE LUANDA

IMPACTO DAS METODOLOGIAS SCRUM E RUP PARA A GESTÃO DE PROJECTOS DE DESENVOLVIMENTO NO CENTRO DE INVESTIGAÇÃO DO INSTIC

José Gomes Figueiredo

jogofi06@gmail.com

Noémia Madalena Azevedo Morais Livondeni

noemia_morais22@hotmail.com

Cristina Manuel Alexandre Bongo

cristinaalexandre1708@gmail.com

Resumo

As equipes de desenvolvimento buscam metodologias que se alinhem aos objectivos de determinado projecto. A metodologia ágil Scrum, é uma das metodologias ágeis mais utilizadas, que se baseia na flexibilidade, na entrega incremental de valor e na colaboração contínua entre os membros da equipe (Schwaber, 2020). Ela estrutura o trabalho em ciclos curtos chamados sprints, onde requisitos são ajustados de forma interativa. Essa abordagem melhora a comunicação e permite rápida adaptação às mudanças, algo essencial em ambientes de pesquisa onde os objectivos podem evoluir com o tempo. No contexto do centro de investigação (CEICD), o Scrum pode ser particularmente útil para projectos experimentais, onde as hipóteses e metodologias podem mudar rapidamente com base em novos achados.

Este artigo teve como objectivo analisar o impacto da aplicação de duas metodologias distintas na gestão eficaz de projectos de desenvolvimento de software no Centro de Investigação Científica e Desenvolvimento (CEICD) do INSTIC: a metodologia ágil Scrum e a metodologia tradicional RUP (Rational Unified Process). A investigação procurou compreender como cada uma destas abordagens contribuiu para o sucesso dos projectos desenvolvidos neste centro.

No contexto específico do CEICD, onde a natureza dos projectos pode frequentemente envolver a participação de múltiplas partes interessadas com diferentes necessidades e onde se torna crucial a obtenção de validação formal em diversas etapas do ciclo de vida do projecto, a utilização da metodologia

RUP demonstrou garantir um processo de desenvolvimento mais seguro e com maior previsibilidade. A sua estrutura interativa e a ênfase na documentação e controlo parecem ter sido particularmente adequadas para lidar com a complexidade e os requisitos de aprovação formal inerentes a alguns projectos do centro.

De forma notável, e em paralelo com os benefícios observados na aplicação do RUP em cenários específicos, a metodologia Scrum também apresentou uma taxa de sucesso de 85% nos projectos onde foi implementada no CEICD. Este resultado sugere que a flexibilidade, a adaptabilidade e a abordagem interativa e incremental do Scrum foram igualmente eficazes em outros contextos de projecto dentro da organização.

Palavras-chave: Projectos, Desenvolvimento, Scrum, RUP, CEICD.

Abstract

Development teams look for methodologies that align with the objectives of a given project. Scrum is one of the most widely used agile methodologies, based on flexibility, incremental delivery of value and continuous collaboration between team members. It structures work in short cycles called sprints, where requirements are adjusted iteratively. This approach improves communication and allows rapid adaptation to change, which is essential in research environments where objectives can evolve over time. In the context of the research center (CEICD), Scrum can be particularly useful for experimental projects, where hypotheses and methodologies can change rapidly based on new findings.

The aim of this article was to analyze the impact of applying two different methodologies on the effective management of software development projects at INSTIC's Scientific Research and Development Centre (CEICD): the agile Scrum methodology and the traditional RUP (Rational Unified Process) methodology. The research sought to understand how each of these approaches contributed to the success of the projects developed at this center.

In the specific context of the CEICD, where the nature of projects can often involve the participation of multiple stakeholders with different needs and where it is crucial to obtain formal validation at various stages of the project life cycle, the use of the RUP methodology has proved to guarantee a safer and more predictable development process. Its iterative structure and emphasis on documentation and control seem to have been particularly suitable for dealing with the complexity and formal approval requirements inherent in some of the center's projects. Remarkably, and in parallel with the benefits observed in the

application of RUP in specific scenarios, the Scrum methodology also showed a success rate of 85% in the projects where it was implemented at CEICD. This result suggests that Scrum's flexibility, adaptability and iterative and incremental approach were equally effective in other project contexts within the organization.

Keywords: project, development, Scrum, RUP, CEICD.

Introdução

As metodologias Scrum e Rup foram criadas para revolucionar o processo de desenvolvimento de softwares, pois elas possuem prácticas para auxiliar na organização do tempo e dos processos a fim de se desenvolver um software de alta qualidade.

A gestão eficiente de projectos de desenvolvimento no âmbito de um centro de investigação, como o Instituto de Tecnologias de Informação e Comunicação (INSTIC), exige o uso de metodologias estruturadas que se alinham a agilidade, controle e qualidade. Neste contexto, duas abordagens destacam-se: Scrum (uma metodologia ágil) e RUP (Rational Unified Process – um processo de engenharia de software iterativo). Ambas oferecem estratégias distintas para o planeamento, execução e entrega de projectos, mas com impactos diferenciados em ambientes de pesquisa e desenvolvimento tecnológico.

Para avaliar o impacto dessas metodologias na gestão de projectos, foi fundamental a aplicação de métodos **qualitativos** e **quantitativos**. Métodos qualitativos, como **entrevistas semiestruturadas e estudos de caso**, foram levados a cabo e estes forneceram-nos mais insights sobre a experiência dos gestores e pesquisadores na adopção dessas abordagens. Já métodos quantitativos, como **análises estatísticas e métricas de desempenho**, puderam medir a eficiência e a taxa de sucesso dos projectos conduzidos sob cada metodologia. Estudos demonstraram que a adopção destas metodologias levou a um aumento de **85% na satisfação da equipe**, devido à melhoria na comunicação, colaboração e clareza de processos. A triangulação dessas abordagens permitiu uma compreensão mais abrangente do impacto das metodologias ágil Scrum e da tradicional RUP na productividade, inovação e colaboração dentro do centro de investigação (CEICD).

Revisão da literatura

SCRUM consiste em uma metodologia ágil de desenvolvimento de software que visa a agilidade, a redução de custos e o aumento da qualidade do produto. Sendo assim, esta metodologia não tem como foco principal o controlo do desenvolvimento do que foi previamente estabelecido, mais sim o controlo do processo de desenvolvimento do software a fim de se desenvolver o melhor software possível que atenda as necessidades do cliente, adaptando-se as mudanças.

Esta metodologia foi projectada para ajudar as equipes a desenvolver produtos complexos e de alto valor de forma iterativa e incremental e permitir Feedback rápido, inovação, melhoria contínua, adaptação rápida à mudança, clientes satisfeitos, e tempo reduzido de entrega. O Scrum divide o desenvolvimento em iterações (sprints) de trinta dias para a entrega de valor incremental. Equipes pequenas, de até dez pessoas, são formadas por projectistas, programadores, engenheiros e gerentes de qualidade.

Ainda (Mango, 2017), menciona que a Scrum é caracterizada por colocar como foco as iterações e o cliente. No início de cada iteração, toda a equipe se reúne e analisa os requisitos, a tecnologia e as habilidades necessárias; posteriormente, conseguem escolher quais actividades deverão ser feitas naquele próximo ciclo para agregar valor ao produto.

SCRUM é uma metodologia aplicável a qualquer projecto que tenha como problema prazos e que possa atender a requisitos complexos, e tem como foco produzir um software de forma flexível e em um ambiente em constante mudança.

Os papéis do scrum

Product Owner: é responsável por priorizar os itens do Product Backlog, este representa e defende os interesses do cliente, motiva a equipe e gerência novos requisitos, logo, ele é o responsável por maximizar o valor do software em desenvolvimento. Este também, deve de forma clara e transparente expressar o que está a ser realizado durante o desenvolvimento do software.

Segundo (Silva, et al., 2015), o Product Owner é um profissional que desempenha o papel de cliente diante os membros da equipe de desenvolvimento do software e tem proximidade tanto da equipe de desenvolvimento de software (a fim de esclarecer dúvidas) bem como de seus clientes. Este interage com grande frequência e diariamente com todos os envolvidos no projecto e ninguém, além dele, tem autoridade para interferir e priorizar quais serão as tarefas que irão requerer ajustes, sendo assim, ele é o tomador de decisões e deve liderar os membros da equipe durante o desenvolvimento do software.

Scrum Master: é responsável por garantir que a equipe siga continuamente as boas práticas oferecidas pelo Scrum durante todo processo de desenvolvimento do software, por remover qualquer obstáculo identificado, garantir que a equipe não sofra interferências e ajudar o Product Owner na priorização dos requisitos. (Silva, et al., 2015) menciona que o Scrum Master garante que todos os eventos e reuniões sejam realizados quando necessários, assim como a utilização dos artefactos.

Team Scrum: para (Silva, et al., 2015) é um grupo multidisciplinar que organiza e realiza todo o trabalho e resultado planeado em um Sprint. É o team scrum quem planeia, nas reuniões de início do Sprint, todo o trabalho necessário para a entrega dos objectivos estabelecidos pelo Product Owner. O Team tem autoridade e propriedade para determinar tecnicamente como o produto será concebido e planear e estimar o tempo necessário das tarefas a serem executadas durante o Sprint. (Schwaber, et al., 2017) dizem que o Team Scrum consiste de profissionais que realizam o trabalho de entregar um incremento potencialmente liberável do produto "Pronto" ao final de cada Sprint.

Ciclo de vida do scrum

As tarefas de desenvolvimento do software são divididas entre várias equipes, um dos membros de cada equipe também desempenha o papel de Scrum Master (responsável por organizar o grupo). As equipes implementam recursos do producto em uma serie de Sprint de quatro a seis semanas, cada um dos quais culmina em um protótipo funcional. No final de cada Sprint, a equipe de desenvolvimento e a gerência realizam uma reunião de planeamento de meio dia ou dia inteiro para decidir quais tarefas devem ser realizadas durante o próximo sprint (sprint backlog).

Todos os dias, durante um sprint, as equipes realizam uma reunião scrum de 15 minutos, em que cada membro da equipe responde a três perguntas:

O que tu fizeste desde a última reunião scrum?

Tu tens algum obstáculo?

O que tu farás antes do próximo scrum?

Para qualquer problema que não possa ser imediatamente resolvido ou se um membro da equipe parece estar com problemas de progredir, o Scrum Master marca reuniões separadas ou toma qualquer outra acção apropriada. As reuniões do scrum não devem durar mais de 15 minutos e, para ajudar a garantir isso, os membros da equipe normalmente ficam de pé durante a reunião (Hicks, et al., 2010).

A Figura 2.2 ilustra os processos básicos do Scrum para o desenvolvimento de software.

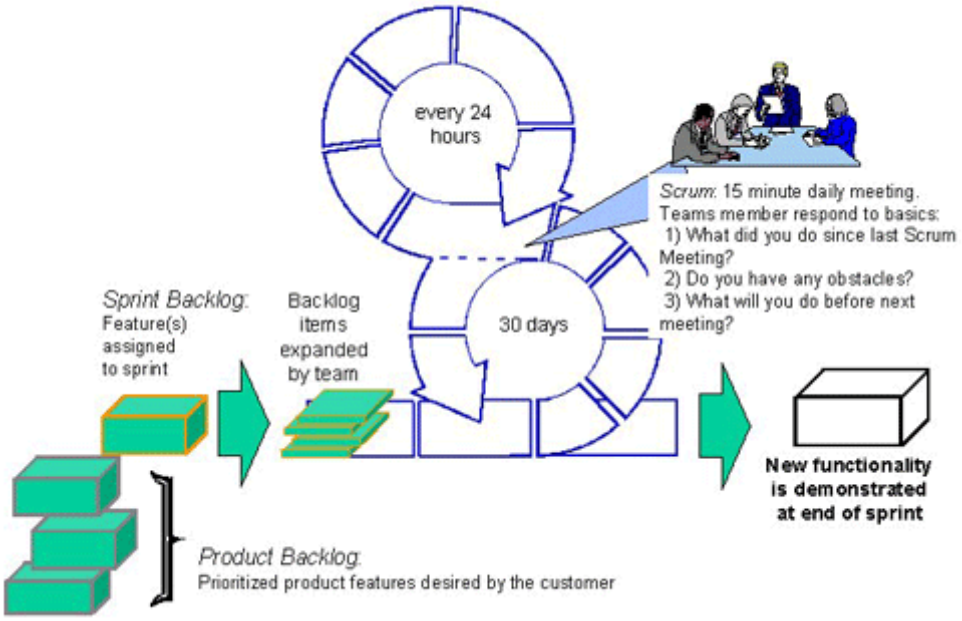

Figura 2.2 – Fonte: (Hicks, et al., 2010)

O Product Backlog é uma lista ordenada por prioridade (do mais ao menos relevante) gerenciada pelo Product Owner, onde consta tudo que será necessário para melhorar e concluir o software. Esta lista apresenta todas melhorias e correcções a serem feitas no software e as características, funções, requisitos que deve ter a nova versão do software.

O Sprint Backlog refere-se a uma lista de tarefas, seleccionadas do topo do Product Backlog, e suas respectivas estimativas de duração previstas a serem concluídas no Sprint. Esta lista é definida pelo Team Scrum.

Para (Silva, et al., 2015) todo o desenvolvimento tem como base iterações com duração entre 2 a 6 semanas, chamadas Sprints. A primeira etapa no Sprint é a reunião de planeamento (Sprint Planning), onde a equipe (Scrum Team), junto ao cliente (Product Owner) define o que será implementado na interacção, sendo responsabilidade do cliente realizar a priorização do trabalho a ser feito. Em seguida é realizada, apenas pela equipe do projecto, uma reunião (Sprint Retrospective) onde o Sprint é avaliado sob a perspectiva do processo, equipe ou producto, quais foram os acertos e os erros com o propósito de melhorar o processo de trabalho.

Pilares do scrum

O scrum tem três pilares que são: transparência, inspecção e adaptação.

Estes são descritos por (Schwaber, et al., 2017) e (STOPA, et al., 20219) da seguinte forma:

Transparência - as actividades significativas do projecto devem estar sempre visíveis os responsáveis pelo projecto, além de requerer que estes aspectos tenham uma definição padrão simples para que os observadores compartilhem de forma ágil um mesmo entendimento daquilo que está a ser desenvolvido.

Inspecção - Inspecções regulares, feitas pelos usuários Scrum, do trabalho em andamento são essenciais para se detectar variações indesejadas e se alcançar o resultado desejado. A inspecção deve ser frequente, mas não deve ser tão frequente a fim de não atrapalhar o objectivo dos trabalhos. As inspecções são mais benéficas quando realizadas de forma diligente por inspectores especializados no trabalho a se verificar.

Adaptação - Envolve fazer ajustes necessários no processo ou producto sempre que surgirem desvios. Se um inspector determina que um ou mais aspectos de um processo desviou-se ou está fora dos limites aceitáveis, e que o resultado do produto será inaceitável, o processo ou o material que está a ser produzido deve ser ajustado. As equipes Scrum têm a flexibilidade de adaptar o backlog do producto, o producto e seus planos futuros a cada sprint visando garantir que quaisquer mudanças necessárias possam ser implementadas tão rapidamente.

RUP - rational unified process

O RUP é um producto do processo de engenharia de software, criado pela Rational Software Corporation e posteriormente comprada pela IBM, que fornece uma *abordagem disciplinada para atribuir tarefas e responsabilidades dentro de uma organização de desenvolvimento de software*. Esta também fornece técnicas a serem seguidas por *desenvolvedores de software* a fim de melhorar sua productividade e desenvolver um software de qualidade que atenda as necessidades do cliente.

Para (Emanoele, 2023), o Rup é uma *metodologia utilizada em projectos complexos a fim de promover uma solução na organização das tarefas e das responsabilidades das equipes*. Sua estrutura de trabalho é baseada em um método *iterativo incremental* aplicada a projectos de desenvolvimento de softwares.

Acrescenta ainda (Fonsec, 2012), o Rup define claramente quem é responsável pelo que no projecto, assim como o que deve ser feito e quando fazê-las, e é guiada por casos de uso e centralizada na

arquitectura. (Fonsec, 2012) descreve o rup como uma metodologia flexível, que pode ser utilizada tanto por grandes equipes como por pequenas equipes, e até mesmo empregada para pequenos, médios ou grandes projectos de software.

A metodologia RUP, acrescenta (SOMMERVILLE, 2011) utiliza a UML para apoiar as prácticas de engenharia de software orientada a objectos, especificos, modela e documenta artefactos do projecto, por meio de modelos sequenciais, modelos de objectos etc.

O RUP não é uma metodologia ágil, ela apenas adopta algumas das características ágeis, como a adaptação \a mudanças.

(SOMMERVILLE, 2011) descreve o RUP como um modelo constituído de fases que identifica quatro fases distintas no processo de software.

Fases do RUP

As fases do RUP são estreitamente relacionadas ao negócio, e não a assuntos técnicos.

Concepção - O objectivo da fase de concepção é estabelecer um business case para o sistema. Deve-se identificar todas as entidades externas (pessoas e sistemas) que vão interagir com o sistema e definir as interacções. Em seguida, deve-se usar essas informações para avaliar a contribuição do sistema para o negócio. Se essa contribuição for pequena, então o projecto poderá ser cancelado depois dessa fase.

Elaboração - As metas da fase de elaboração, servem para desenvolver uma compreensão do problema dominante, estabelecer um framework da arquitectura para o sistema, desenvolver o plano do projecto e identificar os maiores riscos do projecto. No fim dessa fase, deve-se ter um modelo de requisitos para o sistema, que pode ser um conjunto de casos de uso da UML, uma descrição da arquitectura ou um plano de desenvolvimento do software.

Construção- Esta fase envolve projecto, programação e testes do sistema. Durante essa fase, as partes do sistema são desenvolvidas em paralelo e integradas. Na conclusão dessa fase, deve-se ter um sistema de software já funcionando, bem como a documentação associada pronta para ser entregue aos usuários.

Transição. Esta é a fase final do RUP que implica transferência do sistema da comunidade de desenvolvimento para a comunidade de usuários, e em seu funcionamento um ambiente real. Na

conclusão dessa fase, deve-se ter um sistema de software documentado e funcionando correctamente em seu ambiente operacional.

Boas práticas do RUP

(SOMMERVILLE, 2011) apresenta as boas práticas do RUP da seguinte forma:

Desenvolver o software iterativamente –O software é desenvolvido em partes, com cada iteração entregando uma versão incrementada do sistema com base nas prioridades dos clientes, permitindo ajustes e melhorias contínuas;

Gerenciar requisitos – Os requisitos do cliente devem ser explicitamente documentados e deve-se acompanhar as mudanças para que o software atenda às necessidades do cliente;

Usar arquitecturas baseadas em componentes – Deve se estruturar a arquitectura do sistema em componentes do Rup promove, o que pode facilitar o desenvolvimento e reduzir o tempo de desenvolvimento.

Modelar software visualmente – Usar modelos gráficos da UML para apresentar uma visão do software, o que facilita a comunicação entre os membros da equipe e a compreensão do projecto.

Verificar a qualidade do software – Assegurar que o software atenda aos padrões de qualidade organizacional por meio de testes e inspecções feitos ao longo do processo de desenvolvimento.

Controlar as mudanças do software – Permite controlar as alterações no software, garantindo a estabilidade e a integridade do sistema.

Afinidades e diferenças entre o scrum e o RUP

Scrum e RUP apresentam algumas semelhanças. (Emanoele, 2020) destaca algumas delas tais como apresentadas na tabela 2.7.1:

Tabela 2.7.1 – Semelhanças e diferenças entre Scrum e Rup(Emanoele, 2020)

Aspectos	Scrum	RUP
Estruturas de trabalho	Possui um ciclo de vida utilizando artefactos e eventos, que estão em constante melhorias e iterações, chamadas de *sprints*.	Possui um processo dividido em fases e iterações, em que cada etapa possui um objectivo já estabelecido.

	Os objectivos de cada sprint são traçados no início do mesmo.	
Tempo do processo de iterações	O ciclo acontece em período de tempo menor.	O ciclo acontece em um período de tempo maior.
Aplicação	Tem foco na gestão ágil de projectos, e é utilizada juntamente com outras metodologias e ferramentas, em diversas áreas.	Apresenta processos mais específicos para a área de desenvolvimento de software que utiliza os conceitos de orientação a objecto e da UML para criar as notações gráficas através de diagramas.

Metodologia

Esta secção descreverá a abordagem metodológica adoptada para a colecta de dados. Antes da iniciar a colecta, foram estabelecidos os objectivos específicos da mesma, como, a recolha de dados **qualitativos** e **quantitativos**. Métodos qualitativos, como **entrevistas semiestruturadas e estudos de caso**, foram levados a cabo e estes forneceram-nos mais insights sobre a experiência dos gestores e desenvolvedores na adopção destas abordagens. Já métodos quantitativos, como **análises estatísticas e métricas de desempenho**, puderam medir a eficiência e a taxa de sucesso dos projectos conduzidos sob cada metodologia. Serão apresentados detalhes sobre a colecta de dados, processamento, escolha das metodologias mais usadas, classificação das mesmas e a devida avaliação do uso destas metodologias.

Colecta dos dados e apresentação dos resultados

Os dados apresentados nesta secção resultaram da aplicação de **métodos qualitativos e quantitativos** durante a recolha de informações realizada com a equipa de desenvolvimento do Centro, junto aos gestores de projecto no local. O processo foi orientado por **questões de investigação**, que direccionaram a busca de dados relevantes, focando na **comparação entre metodologias RUP e SCRUM**.

Foram considerados os seguintes critérios(métricas) de análise:

Desempenho da equipa (productividade, colaboração);

Documentação do projecto (rigor, completude);

Eficiência metodológica (cumprimento de prazos, alocação de recursos);

Vantagens comparativas na aplicação a projectos de desenvolvimento.

Estes elementos permitiram uma **avaliação estruturada** das metodologias, destacando o seu impacto práctico e adaptabilidade aos contextos em estudo.

"Os gestores de projecto entrevistados destacaram a **flexibilidade das metodologias SCRUM e OpenUp** *em contraste com a* **previsibilidade da metodologia RUP***, sobretudo em projectos com requisitos dinâmicos".*

A tabela 3.1, mostra os resultados obtidos a partir do CEICD, por meio de entrevistas e questionários feitos durante o desenvolver deste artigo.

Tabela 3.1 - Resultado comparativo entre as metodologias RUP e SCRUM (fonte: Autores)

Critério de Análise	Metodologia Tradicional (RUP)	Metodologia Ágil (Scrum)	Observações
Desempenho da Equipa	- Comunicação formalizada - Ritmo previsível	- Colaboração intensiva - Adaptação contínua	**Scrum** favorece equipas multidisciplinares
Documentação	- Detalhada e obrigatória - Fase inicial crítica.	- Mínima e funcional - Foco em "working software"	**RUP** exige mais recursos para documentação
Eficiência (Prazos)	- Cumprimento rígido de fases - Risco de atrasos	- Entregas iterativa - Flexibilidade a mudanças	**Scrum** reduz atrasos em projetos dinâmicos
Gestão de Recursos	- Alocação fixa por fase - Custo inicial elevado	- Recursos ajustáveis por sprint - Custos iterativos	**RUP** pode ser menos eficiente em projectos incertos.
Vantagens Principais	- Controlo rigoroso - Ideal para requisitos estáveis	- Rapidez de resposta - Satisfação do cliente	Escolha depende do tipo de projecto e stakeholders

O mapa mental apresentado na figura 3.1, resulta da análise dos dados da Tabela 3.1, sintetizando-os de forma clara e objectiva.

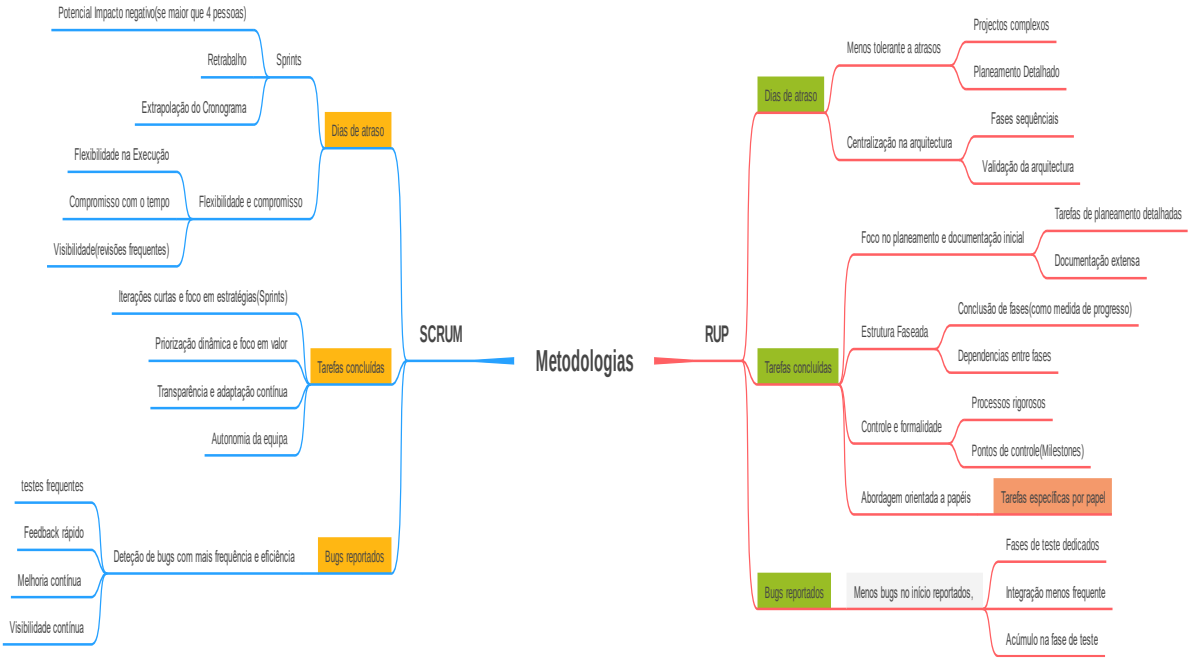

Figura 3.1 - fonte: (autores, 2025).

DESEMPENHO DAS METODOLOGIAS APLICADAS AOS PROJECTOS

Para sustentar as métricas de desempenho foram analisados os ***bugs reportados*** durante o desenvolvimento do projecto as ***tarefas concluídas*** e os ***dias de atraso*** e, com base na fórmula abaixo nos foi possível achar o percentual necessário para validar o que noutrora foi afirmado.

Segue-se então a fórmula de cálculo percentual, usada para achar as métricas das metodologias propostas neste artigo:

Percentagem = (**Valor da Métrica Base** / **Valor Total ou Base de Comparação**) * **100%**

Métricas base, estabelecidas:

Percentagem de **Tarefas Concluídas;**

Percentagem de **Bugs Reportados;**

Percentagem de **Dias Em Atraso.**

Com base nas respostas obtidas a partir dos questionários, foi possível criar a tabela de indicadores com o foco nas métricas previamente estabelecidas(ver tabela 3):

Tabela 3 – Análise de indicadores

Indicador	Sprint/Fase 1	Sprint/Fase 2	Sprint/Fase 3	Média
Dias de atraso	2	0	1	1.0
Tarefas concluídas	85%	92%	78%	85%
Bugs reportados	5	3	4	4.0

Assim sendo temos que:

Percentagem de Tarefas Concluídas

Percentagem = (Número de tarefas concluídas / Número total de tarefas planeadas) * 100%

SPRINT 1: Percentagem = (5/8)*100%= 62.5%

SPRINT 2: Percentagem = (7/8)*100%= 87.5%

SPRINT 3: Percentagem = (6/8)*100%= 75%

De suida são apresentados resultados baseados nas respostas teóricas respondidas pela equipe de gestores do projecto de desenvolvimento de software no CEICD, que revelou opiniões distintas:

Gestor 1: "*prefiro o Scrum, pois é um framework ágil que foca diretamente no desenvolvimento iterativo e na colaboração com o cliente. Ele permite entregas frequentes, adaptação rápida às mudanças e mantém o cliente envolvido durante todo o processo. Essa abordagem reduz riscos e aumenta a qualidade do produto final, já que os feedbacks são constantes e incorporados ao longo do desenvolvimento*".

Gestor 2: "*As duas são boas metodologias dependendo do projecto e contexto de aplicação. Para projectos menores com menos documentação o SCRUM é uma boa opção. Para projectos maiores, como o que estamos a desenvolver, o RUP se adapta melhor devido a sua confiabilidade, embora que o SCRUM também em muitos casos esteja a ser utilizado, como por exemplo na elaboração de sprints*".

Gestor 3: "*Apostar mais na metodologia OpenUp que é praticamente uma analogia do RUP nas metodologias ágeis, pois ambas possuem os mesmos objectivos de desenvolvimento com qualidade e documentação forte*".

Discussão

O Scrum segundo (SHWABER, 2002), tem sido sugerido como uma forma de responder rapidamente as mudanças, diminuindo o tempo de desenvolvimento e melhorando a comunicação e colaboração, especialmente em situações onde o tempo é uma vantagem competitiva crítica para a organização (Runeson, 2003).

De acordo com os resultados obtidos foi possível notar que:

Metodologias ágeis são uma mais valia para se alcançar resultados estratégicos;

O uso das metodologias híbridas, dependendo do contexto ou o tipo de projecto abre caminhos excepcionais e alberga valor nos projectos;

Metodologias ágeis têm recebido significante atenção e dentre elas, o Scrum tem se tornado uma das mais populares e bem-sucedida metodologia de gerenciamento de projetos de software aplicados atualmente na indústria (WANGENHEIM, 2003);

O Scrum é um framework fundamentalmente empírico e a sua teoria é baseada em experiências prácticas de controle de processos. Isso mostra a sua eficácia com relação as prácticas de gerenciamento e desenvolvimento de productos, de modo que você possa melhorá-las ((SCHWABER & SUTHERLAND, 2013)).

Com base nas limitações identificadas e nos resultados deste estudo, diversos factores para pesquisas futuras apresentaram-se promissores. Primeiramente, seria valioso expandir a dimensão da amostra de participantes para incluir um leque mais vasto de gestores e, potencialmente, membros das equipas do projecto e outros intervenientes. Além disso, a adoção de uma abordagem metodológica híbrida, combinando entrevistas com a análise documental do projeto, poderia fornecer uma compreensão mais rica e triangulada dos desafios e sucessos na gestão de projectos no CEICD.

Conclusão

Os estudos indicam que, na Scrum nos dias de atraso são superados com sprints também é mas flexível com compromissos, ao passo que na Rup e centralizada na arquitectura.

A importância de se basear em metodologias de gestão de desenvolvimento como estas, estão nos padrões universais que regem as normas de desenvolvimento de software, acabamos por estudamos metodologias que se enquadram no nosso ambiente de trabalho e realidade, justificando com argumentações de enquadrar os diagramas de UML, enquadrar os Sprints, complemento de tarefas a tempo.

Segundo estudos efectuados notou-se que o uso de metodologias em um determinado problema a ser resolvido deve se ter em conta as causas na qual serão solucionadas, assim sendo para a implementação de uma metodologia deve-se estudar o problema, no caso do problema estudado no INSTIC, notou-se o uso de metodologias híbridas, Scrum e Rup, ambas serviram para uso por duas razões, Scrum por causa dos Sprint a serem entregues e Rup por causa das dos diagramas de UML. Os estudos indicam que Scrum dedica-se na entrega de valor ao cliente de forma eficiente, rápida e flexível, especialmente em projectos complexos que exigem adaptação a mudanças.

Assim, a análise comparativa das duas metodologias revelou que ambas metodologias, tradicional RUP e a ágil Scrum, demonstraram ser abordagens válidas e eficazes para a gestão de projectos de desenvolvimento de software no CEICD. A escolha entre uma e outra metodologia pareceu estar intrinsecamente ligada ao contexto específico do projecto, às suas características particulares, aos requisitos das partes interessadas e à necessidade de um maior grau de formalidade e previsibilidade versus flexibilidade e adaptação rápida a mudanças. O uso das metodologias híbridas, são fundamentais para implementação definitiva do software, auxiliando um ou outro aspecto que esteja em falta.

Dada a importância da gestão do tempo identificada nas entrevistas, pesquisas futuras poderiam investigar em profundidade os factores que contribuem para os atrasos(bugs reportados) e explorar estratégias eficazes para mitigar estes problemas.

Referências bibliográficas

Emanoele, Alícia. 2020. https://voitto.com.br/blog/artigo/o-que-e-rup. *https://voitto.com.br.* [Online] 27 de 08 de 2020.

—. 2023. https://voitto.com.br/blog/artigo/rup-e-scrum. *https://voitto.com.br.* [Online] Voitto, 2023.

Fonsec, Bradley Felipe Zanoni. 2012. Proposta de unificação das metodologias RUP e Scrum através de sua releitura. *Proposta de unificação das metodologias RUP e Scrum através de sua releitura.* 2012.

Hicks, Michael e Foster, Jeffrey S. 2010. Adapting Scrum to Managing a Research Group . *Adapting Scrum to Managing a Research Group .* 2010.

Mango, Renan. 2017. https://www.linkedin.com/pulse/comparativo-entre-metodologias-%C3%A1geis-scrum-e-xp-para-produ%C3%A7%C3%A3o-mango/?originalSubdomain=pt. *https://www.linkedin.com.* [Online] Linkedin., 08 de 12 de 2017.

Schwaber, Ken e Sutherland, Jeff. 2017. Guia do Scrum. *Um guia definitivo para o Scrum: As regras do Jogo.* 2017.

Silva, Edson Coutinho da e Lovato, Leandro Alvarez. 2015. FRAMEWORKSCRUM: EFICIÊNCIA EM PROJETOS DE SOFTWARE. *FRAMEWORKSCRUM: EFICIÊNCIA EM PROJETOS DE SOFTWARE.* 2015.

Soares, Michel dos Santos. Metodologias Ágeis Extreme Programming e scrum para desenvolvimento de software . *Metodologias Ágeis Extreme Programming e scrum para desenvolvimento de software .*

SOMMERVILLE, Ian. 2011. *Engenharia de Software.* São Paulo : Pearson Education do Brasil, 2011.

Srinivasan, Ram e Maloney, Bernie. https://www.scrumalliance.org/about-scrum. *https://www.scrumalliance.org.* [Online]

STOPA, Gabriel Rocha e RACHID, Christien Lana. 20219. SCRUM: METODOLOGIA ÁGIL COMO FERRAMENTA DE GERENCIAMENTO DE PROJETOS. *SCRUM: METODOLOGIA ÁGIL COMO FERRAMENTA DE GERENCIAMENTO DE PROJETOS.* 20219.

Schwaber, K., & Sutherland, J. (2020). **The Scrum Guide**.

Rubin, K. S. (2012). Essential Scrum: **A Practical Guide to the Most Popular Agile Process.** Addison-Wesley.

ANDERSON, DJ. **Agile management for software engineering, applying the theory and constraints for business results**. Prentice Hall, Upper Saddle River, NJ, 2003.

KARLSTRÖM, D.; RUNESON, P. **Integrating agile software development into stage-gate managed product development. Empirical Software Engineering**, 11(2):203–225, 2006.

SCHWABER, Ken e SUTHERLAND Jeff. **The Scrum Guide**. Scrum.org, 2013.

VON WANGENHEIM, C. G.; SAVI, R.; BORGATTO, A. F. SCRUMIA-**An educational game for teaching SCRUM in computing courses**. Journal of Systems and So