josif 2025
17ª Jornada Científica e Tecnológica
14º Simpósio de Pós-Graduação
do IFSULDEMINAS
ISSN: 2319-0124
REALIZAÇÃO
INSTITUTO FEDERAL Sul de Minas Gerais
APOIO
CNPq
FAPEMIG

INTEGRAÇÃO CONTÍNUA NO DESENVOLVIMENTO DE SOFTWARE: Benefícios, desafios e melhores práticas

Luana S. N. de CASTRO[1]; Paulo C. dos SANTOS[2]

RESUMO

Este estudo caracteriza-se como uma Revisão Bibliográfica que explora a importância da Integração Contínua (IC) no desenvolvimento de software, especialmente em projetos colaborativos com múltiplos desenvolvedores. A IC tem se mostrado uma prática eficaz na manutenção da qualidade do código, na redução de bugs e no aprimoramento da consistência durante o ciclo de desenvolvimento. No entanto, empresas enfrentam desafios como falhas de integração, bugs não detectados e atrasos nos pipelines de entrega. Este estudo analisa como a implementação de práticas eficazes de IC pode mitigar esses problemas, promovendo uma maior agilidade no ciclo de entrega e uma melhoria significativa na qualidade do software produzido. A pesquisa discute as melhores práticas e estratégias para integrar a IC de forma eficiente, proporcionando um ambiente de desenvolvimento mais colaborativo e produtivo.

Palavras-chave:
DevOps; Ferramentas CI; Qualidade de Software.

1. INTRODUÇÃO

No cenário atual do desenvolvimento de software, a crescente complexidade dos sistemas e a demanda por entregas rápidas e de alta qualidade têm desafiado as equipes a adotarem soluções mais eficientes e colaborativas. Calvano e John (2004) destacam que a complexidade moderna exige que ela seja entendida, prevista e medida para que sistemas bem-sucedidos possam ser projetados com confiança.

A Integração Contínua (IC) surge como uma aliada estratégica para enfrentar desafios como a detecção tardia de bugs, atrasos no pipeline de entrega e inconsistências no código em equipes distribuídas. Shahin et al. (2017) afirmam que a IC permite ciclos de lançamento mais curtos e frequentes, melhora a qualidade do software e aumenta a produtividade das equipes. No entanto, a má implementação pode comprometer a eficácia da IC e a qualidade do produto final (SILVA, 2023).

Este trabalho tem como objetivo realizar uma revisão bibliográfica sobre como práticas eficazes de IC podem melhorar a qualidade, reduzir bugs e acelerar ciclos de entrega em projetos colaborativos. Para tanto, analisam-se aspectos teóricos e práticos, com foco em ferramentas, estratégias e desafios documentados na literatura científica.

2. FUNDAMENTAÇÃO TEÓRICA

A Integração Contínua é uma prática de desenvolvimento de software que consiste na integração frequente de alterações de código em um repositório central, seguida de *builds* e testes

[1]Bolsista PIBIC/CNPq, IFSULDEMINAS – *Campus* Muzambinho. E-mail: luana.castro@alunos.ifsuldeminas.edu.br.
[2]Orientador, IFSULDEMINAS – *Campus* Muzambinho. E-mail: paulo.santos@muz.ifsuldeminas.edu.br.

automatizados. O objetivo principal é a identificação precoce de erros e conflitos, garantindo um código-fonte mais estável. Como destaca Meyer (2014, p. 15, tradução nossa), “a IC é mais do que um conjunto de práticas; é uma mentalidade focada em aumentar o valor do cliente”.

Entre seus principais benefícios estão a melhoria da qualidade do software, com a automatização de testes aumentando a confiabilidade do código, e a promoção da padronização e colaboração em equipes distribuídas. Duvall, Matyas e Glover (2007) reforçam que a IC reduz riscos e processos manuais repetitivos, proporcionando maior visibilidade do projeto. A agilidade no ciclo de desenvolvimento também é um aspecto crucial, com a entrega contínua permitindo liberações rápidas de funcionalidades e correções, o que reduz atrasos e aumenta a competitividade das organizações.

Para uma implementação eficaz, é essencial adotar ferramentas de automação como Jenkins, GitLab CI e Travis CI, além de boas práticas como testes automatizados abrangentes e revisão de código sistemática. Oliveira Lima (2021, p. 25) observa que “ferramentas são essenciais para a execução dos processos que envolvem o DevOps e, consequentemente, a IC”. Em resumo, a IC, quando bem aplicada, melhora a qualidade do software, aumenta a produtividade e fortalece a colaboração das equipes.

3. MATERIAL E MÉTODOS

Este estudo constitui uma revisão bibliográfica da literatura. A pesquisa foi conduzida para identificar, analisar e sintetizar os achados científicos sobre os benefícios, desafios e melhores práticas da Integração Contínua. O processo metodológico seguiu as etapas de planejamento, busca, seleção e análise de artigos.

A busca foi realizada nas bases de dados acadêmicas IEEE Xplore, Proquest e Unichristus, além de repositórios institucionais de universidades brasileiras. A *string* de busca principal combinou os seguintes termos e seus sinônimos: ("Continuous Integration" OR "CI/CD") AND ("software quality" OR "challenges" OR "best practices" OR "agile development").

Como critérios de inclusão, foram definidos: (a) artigos publicados entre 2014 e 2024; (b) artigos revisados por pares, dissertações e teses; (c) estudos que abordassem diretamente a aplicação da IC no contexto de desenvolvimento de software. Foram excluídos trabalhos que não apresentavam rigor metodológico, artigos de opinião e literatura cinzenta. Após a aplicação dos filtros e a leitura dos resumos e textos completos, um total de 28 publicações foram selecionadas para a análise aprofundada.

4. RESULTADOS E DISCUSSÃO

A análise dos 28 artigos selecionados revelou padrões consistentes sobre os impactos, ferramentas e desafios da Integração Contínua. A Tabela 1 sintetiza a distribuição temática das publicações analisadas, oferecendo uma visão quantitativa da ênfase dada a cada subtema pela

comunidade científica na última década.

Tabela 1 – Resumo das informações sobre publicações relacionadas à Integração Contínua (N=28)

Temática	Base de dados científica	Período	Percentual
Práticas e ferramentas de Integração Contínua (CI/CD)	IEEE Xplore, Proquest, Repositórios institucionais (UFC, UFOP, Unichristus)	2015 a 2023	42%
Impactos da IC na qualidade e produtividade	Repositório UFC, IEEE Xplore	2017 a 2023	29%
Desafios na implementação da IC	Repositórios UFC, UFOP	2017 a 2023	21%

Fonte: Elaborado pelos autores (2025).

4.1 Impacto da IC na Qualidade do Software

A literatura é quase unânime em apontar a automação de testes como o principal vetor de qualidade proporcionado pela IC (SHAHIN et al., 2017). A detecção precoce de erros, como apontado por Silva (2023), não apenas reduz os custos com manutenção, mas também aumenta a confiança da equipe no código-base. Cerca de 29% das publicações analisadas focaram diretamente nesses impactos, correlacionando a frequência de integração com a estabilidade do software.

4.2 Práticas e Ferramentas de IC

Sendo o tema mais prevalente (42%), a discussão sobre ferramentas como Jenkins, GitLab CI e CircleCI demonstra a preocupação da área com a operacionalização da IC. Os estudos indicam que não existe uma "ferramenta perfeita", mas sim uma escolha adequada ao contexto do projeto (tamanho da equipe, tecnologia utilizada, infraestrutura existente). Oliveira Lima (2021) argumenta que a eficácia não está na ferramenta em si, mas na sua correta configuração e na cultura DevOps que a suporta.

4.3 Desafios na Implementação

Apesar dos benefícios, a implementação da IC não é trivial. Os desafios, abordados em 21% dos trabalhos, incluem a complexidade na configuração inicial dos *pipelines*, a criação de suítes de testes que sejam rápidas e abrangentes, e a resistência cultural à disciplina de integrar o código com frequência. Silva (2023) investiga os efeitos de "más práticas" de IC, como ignorar *builds* quebrados, mostrando que a simples adoção de ferramentas sem adesão às práticas pode ser contraproducente.

5. CONCLUSÃO

Esta revisão bibliográfica confirma que a Integração Contínua é uma estratégia fundamental para o desenvolvimento de software moderno, impactando positivamente a qualidade, a

produtividade e a colaboração. A análise da literatura demonstrou que o sucesso da IC depende de um tripé: ferramentas adequadas, práticas rigorosas e uma cultura de colaboração e disciplina.

Os principais desafios, como a complexidade da configuração e a manutenção de testes eficientes, podem ser mitigados com planejamento e adesão a boas práticas, como a revisão de código e o monitoramento contínuo. Como implicação prática, gestores de projetos devem priorizar não apenas o investimento em ferramentas, mas também o treinamento das equipes e a promoção de uma mentalidade ágil.

Para pesquisas futuras, sugere-se a investigação do impacto da Inteligência Artificial na otimização de *pipelines* de CI/CD, bem como estudos longitudinais que meçam os ganhos de produtividade em equipes antes e depois da adoção madura da Integração Contínua.

REFERÊNCIAS

CALVANO, C.; JOHN, P. **Systems engineering in an age of complexity.** IEEE Engineering Management Review, v. 32, p. 29-38, 2004. Disponível em: https://incose.onlinelibrary.wiley.com/doi/10.1002/sys.10054. Acesso em: 31 jan. 2025.

DUVALL, Paul M.; MATYAS, Steve; GLOVER, Andrew. **Integração Contínua: Melhorando a Qualidade do Software e Reduzindo o Risco**. Boston: Addison-Wesley, 2007.

MEYER, M. **Continuous Integration and Its Tools.** IEEE Software, v. 31, n. 3, p. 14-16, 2014. Disponível em: https://ieeexplore.ieee.org/document/6802994. Acesso em: 04 fev. 2025.

LIMA, Pedro Henrique Oliveira. **Ferramentas essenciais para a execução dos processos que envolvem o DevOps e consequentemente a CI**. 2021. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) – Centro Universitário Christus, Fortaleza, 2021. Disponível em: https://repositorio.unichristus.edu.br/jspui/handle/123456789/1259. Acesso em: 04 fev. 2025.

SHAHIN, M.; BABAR, M.; ZHU, L. **Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices.** IEEE Access, v. 5, p. 3909-3943, 2017. Disponível em: https://ieeexplore.ieee.org/document/7884954. Acesso em: 31 jan. 2025.

SILVA, Ruben Blenicio Tavares da. **Investigando os efeitos das más práticas de integração contínua em projetos de software industriais**. 2023. 131 f. Dissertação (Mestrado em Computação) – Universidade Federal do Ceará, Quixadá, 2023. Disponível em: http://repositorio.ufc.br/handle/riufc/76950. Acesso em: 31 jan. 2025.