Automatizando Auditorias de Processo de Software com Atlassian Rovo (Jira AI): Um Estudo de Caso na Indústria

Gabriel Cezar
Instituto de Pesquisas Eldorado
Campinas, Brasil
gabriel.cezar@eldorado.org.br

Mateus Nunes
Instituto de Pesquisas Eldorado
Campinas, Brasil
mateus.nunes@eldorado.org.br

Nayane Alves
Instituto de Pesquisas Eldorado
Manaus, Brasil
nayane.alves@eldorado.org.br

Anderson Arruda
Instituto de Pesquisas Eldorado
Campinas, Brasil
anderson.arruda@eldorado.org.br

Renan Silva
Instituto de Pesquisas Eldorado
Porto Alegre, Brasil
renan.silva@eldorado.org.br

Gizelle Lemos
Instituto de Pesquisas Eldorado
Campinas, Brasil
gizelle.lemos@eldorado.org.br

Resumo

Auditorias de processos de software são essenciais para garantir governança, conformidade e rastreabilidade em projetos de engenharia de software. No entanto, essas atividades costumam ser altamente manuais e operacionalmente custosas, exigindo grande esforço para coleta de evidências, validação de checklists e consolidação de relatórios em múltiplas ferramentas. Este artigo apresenta um relato de experiência na indústria sobre o uso do Atlassian Rovo integrado ao Jira Automation para apoiar auditorias semiautomatizadas de um processo corporativo de desenvolvimento de software. A solução proposta utiliza agentes de IA especializados para interpretar regras de adaptação de processo, coletar evidências contextuais a partir de ferramentas corporativas e gerar automaticamente comentários estruturados de auditoria. A abordagem foi aplicada em três projetos, considerando as fases de Iniciação, Execução e Encerramento de auditorias. Os resultados demonstraram que auditorias anteriormente executadas manualmente em 4 a 8 horas foram reduzidas em aproximadamente 50% e a execução dos agentes ocorreu em poucos minutos, reduzindo significativamente o esforço operacional repetitivo, ao mesmo tempo em que melhoraram a padronização e a rastreabilidade das atividades de auditoria. A experiência evidencia a aplicabilidade prática de agentes de IA generativa em atividades de governança e conformidade em engenharia de software em ambientes reais.

Palavras-chave

IA para Engenharia de Software, Jira, Auditoria de Processo de Software, Aderência ao Processo, Gestão de Conhecimento

1 Introdução

Organizações que desenvolvem software em ambientes corporativos frequentemente adotam processos padronizados de desenvolvimento de software para garantir qualidade, rastreabilidade e conformidade entre diferentes projetos [11, 12]. Nesse contexto, auditorias de aderência ao processo tornam-se essenciais para verificar a consistência da adoção das práticas organizacionais estabelecidas.

Entretanto, auditorias normalmente representam atividades operacionalmente custosas para equipes de Engenharia de Software, exigindo intensa navegação entre múltiplas ferramentas, coleta manual de evidências, interpretação documental e consolidação de relatórios [7, 10]. Na organização estudada, auditorias de aderência ao processo de software definido frequentemente demandavam entre 4 e 8 horas de execução manual por fase, o que levantou a possibilidade do uso de agentes de IA para apoio. O avanço recente de Large Language Models (LLMs) e agentes de IA generativa vem ampliando as possibilidades de automação também em Engenharia de Software [6, 9]. Apesar disso, aplicações voltadas para governança, conformidade e auditoria de processos ainda possuem pouca exploração prática reportada na literatura e em relatos da indústria [8, 13].

Neste contexto, este trabalho apresenta um relato de experiência sobre a utilização do Atlassian Rovo [1] integrado ao Jira Automation [3] para apoiar auditorias semiautomatizadas do processo de software. A solução proposta utiliza agentes especializados que interpretam a adaptação do processo, consultam ferramentas de trabalho dos times de projeto, consolidam evidências que foram encontradas nas ferramentas para responder às questões do checklist de auditoria, calculam a aderência do projeto em relação ao requerido no processo e produzem justificativas para a aderência calculada. A abordagem foi aplicada em três projetos reais considerando auditorias das fases de Iniciação, Execução e Encerramento.

2 Contexto e Processo de Auditoria

A experiência descrita neste trabalho ocorreu em uma organização brasileira de pesquisa, desenvolvimento e inovação (P&D&I) que tem um processo corporativo de desenvolvimento de software. Nesse processo há práticas, artefatos e critérios de conformidade utilizados para garantir rastreabilidade, governança e padronização entre diferentes projetos.

As auditorias de aderência ao processo são conduzidas pela área de Engenharia de Software durante as fases de Iniciação, Execução e Encerramento dos projetos. Cada auditoria utiliza checklists específicos contendo critérios de conformidade, evidências esperadas e regras de pontuação conforme a aderência obtida pelo projeto. Antes da auditoria, cada projeto realiza uma adaptação formal do processo ao escopo do projeto, definindo disciplinas aplicáveis, exceções permitidas e evidências esperadas para aquele contexto específico.

A Figura 1 apresenta um exemplo de questão avaliada durante as auditorias de aderência ao processo. Embora a estrutura da questão sugira uma resposta objetiva, a avaliação realizada pelo auditor não se resume a um simples "sim" ou "não". Para determinar a aderência

Disciplina	Questão	Pontuação (0/1/2)	Evidência (Link)	Justificativa/Ação	Ação corretiva/Melhoria
Requisitos	Foi definido quem realiza o **Perfil de Product Owner** no projeto e estabelecido e documentado o **Escopo de Alto Nível** para o projeto?	<A pontuação sobre a aderência é 0, 1 ou 2: 0 = não existe evidências; 1 = aderência parcial; 2 = aderência total.>	<Anexar evidência do projeto>	<Incluir em caso de aderência <2>	<Descrever a ação esperada>

Figura 1: Exemplo de item de checklist utilizado durante as auditorias de aderência ao processo

do item, é necessário localizar evidências em diferentes ferramentas, interpretar documentos, validar a consistência das informações encontradas e verificar se elas atendem às regras estabelecidas na adaptação do processo ao projeto. Dessa forma, a auditoria envolve uma análise contextual que combina múltiplas fontes de informação e o julgamento técnico do auditor.

As evidências necessárias para essa análise encontram-se distribuídas em múltiplas ferramentas corporativas do ecossistema Atlassian [4], incluindo Jira, Confluence, Bitbucket e EazyBI. Antes da solução proposta com IA, todas as etapas eram executadas manualmentes, exigindo intensa navegação entre ferramentas, interpretação documental e consolidação textual das evidências.

Neste trabalho, agentes especializados foram utilizados para interpretar regras do processo de software, coletar evidências automaticamente, analisar artefatos e gerar comentários estruturados de auditoria diretamente nas issues do Jira.

3 Integração do Rovo à Atividade de Auditoria

Para reduzir o esforço operacional associado às auditorias do processo de software, foi desenvolvida uma solução baseada na integração de ferramentas corporativas, com o objetivo de automatizar atividades relacionadas à coleta, interpretação e consolidação de evidências, mantendo a rastreabilidade e a centralização das informações, entre elas:

O ***Jira Automation*** é um recurso do Jira que permite automatizar tarefas, processos e fluxos de trabalho por meio de regras configuráveis compostas por gatilhos, condições e ações [3]; O ***Atlassian Rovo*** é a camada de inteligência artificial do ecossistema Atlassian, composta por recursos como busca contextual, chat e agentes configuráveis capazes de apoiar a execução de atividades e reduzir tarefas repetitivas [1]; Já o ***Bitbucket*** é uma ferramenta de hospedagem e colaboração em código baseada em Git, integrada ao Jira e utilizada para apoiar o desenvolvimento, revisão, integração e entrega de software [2]; Enquanto o ***EazyBI*** é uma aplicação de Business Intelligence e análise de dados integrada ao Jira, utilizada para criação de dashboards, relatórios e métricas analíticas [5].

O fluxo é iniciado a partir de uma requisição de auditoria criada no Jira. A execução automatizada ocorre quando a requisição é iniciada através de uma palavra chave no Jira indicando a fase auditada (*Iniciação*, *Execução* ou *Encerramento*). A descrição da requisição contém o link da Adaptação do processo para o projeto, utilizada como principal fonte contextual da auditoria.

A **Figura 2** apresenta a visão geral da arquitetura proposta. O fluxo integra Jira Automation, Rovo Agents, Bitbucket e EazyBI para coleta contextual de evidências, preparação das informações de auditoria e geração automatizada dos comentários estruturados no Jira.

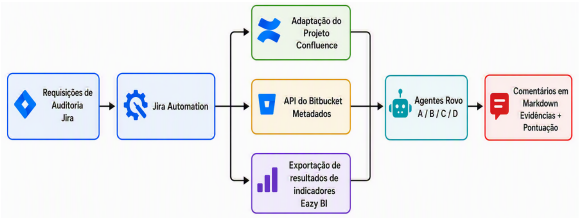

Figura 2: Arquitetura da solução baseada em Jira Automation e Rovo Agents para auditorias do processo de software

A solução foi estruturada em agentes especializados. O Agente A realiza a interpretação da Adaptação do processo de software para identificar regras aplicáveis, exceções permitidas e critérios esperados para a auditoria. O Agente B executa consultas ao Bitbucket para coletar informações relacionadas ao repositório e estrutura do projeto. O Agente C consolida as evidências encontradas em linguagem natural, produzindo contexto resumido para as verificações automatizadas. Já o Agente D coleta métricas complementares a partir do Jira, Confluence e EazyBI.

O desenvolvimento e a configuração da automação e dos agentes especializados demandaram aproximadamente 120 horas de esforço, executadas por um estagiário da área de Engenharia de Software. As atividades incluíram definição das instruções dos agentes, ajustes de integração com as ferramentas corporativas, refinamento das verificações automatizadas e validação dos resultados obtidos. Esse resultado sugere que a construção da solução apresenta uma barreira de entrada relativamente baixa, mesmo quando executada por um profissional em início de carreira com apoio da equipe especializada.

Com o contexto preparado, agentes especializados executam verificações específicas conforme a fase auditada. Na fase de Iniciação, os agentes validam itens relacionados a planejamento, configuração e documentação do projeto. Na fase de Execução, os agentes analisam práticas ágeis, rastreabilidade, indicadores e evidências de desenvolvimento. Já na fase de Encerramento, são verificadas pendências, documentação final e registros de lições aprendidas.

Os resultados são publicados automaticamente em formato *Markdown* diretamente na requisição de auditoria do Jira, incluindo pontuação da aderência calculada pelo(s) agente(s), justificativas, links para evidências e ações corretivas quando aplicável. Em seguida, o resultado gerado pelos agentes é analisado pelo auditor, que confirma ou refuta a aderência indicada com base nas evidências apresentadas e no contexto do projeto. Dessa forma, toda a rastreabilidade da auditoria permanece centralizada no ambiente corporativo utilizado pelas equipes, enquanto a palavra final e a responsabilidade da auditoria continuam sendo do auditor. A solução foi desenvolvida respeitando as permissões existentes, garantindo que os agentes visualizem apenas conteúdos autorizados ao usuário da automação.

4 Método de Avaliação

A solução proposta foi avaliada em três projetos reais da organização estudada considerando auditorias das fases de Iniciação, Execução e Encerramento do processo de software. A avaliação

comparou o esforço operacional necessário para execução manual das auditorias com o tempo necessário para execução automatizada utilizando Jira Automation e Rovo Agents. Os tempos manuais são baseados em registros históricos da equipe de Engenharia de Software durante auditorias convencionais, incluindo coleta de evidências, interpretação documental, validação dos checklists e consolidação dos comentários de auditoria, e não em experimento controlado. Já o tempo de execução automatizada corresponde ao período necessário para conclusão da execução dos agentes após o disparo da automação no Jira, incluindo leitura contextual da Adaptação do processo de software, coleta de informações em ferramentas corporativas, execução das verificações automatizadas e publicação dos comentários estruturados.

Os dados de execução automatizada foram obtidos diretamente a partir dos logs do Jira Automation. Além do tempo de execução, também foram analisados: a quantidade de itens presentes nos checklists; e a quantidade de ações executadas automaticamente pelos agentes. Essas informações foram utilizadas para avaliar o nível de cobertura da automação em relação às verificações previstas nas auditorias.

A proposta não possui como objetivo substituir a atuação humana no processo de auditoria, mas reduzir o esforço operacional associado à coleta, triagem e consolidação das evidências.

5 Resultados e Discussão

A solução proposta foi aplicada em três projetos reais durante auditorias das fases de Iniciação, Execução e Encerramento do processo de software. Os resultados obtidos permitiram avaliar tanto o tempo necessário para execução automatizada das auditorias quanto o nível de cobertura das verificações realizadas pelos agentes especializados.

A **Tabela 1** apresenta os tempos de execução automatizada registrados pelos agentes, os tempos estimados para execução manual das auditorias e a quantidade de itens avaliados em cada fase.

Os resultados evidenciam uma diferença significativa entre o esforço necessário para execução manual das auditorias e o tempo necessário para conclusão da execução automatizada pelos agentes. Enquanto o processo manual demandava entre 4 e 8 horas por fase auditada, as execuções automatizadas foram concluídas em poucos minutos em todos os cenários avaliados.

A fase de Execução apresentou o cenário mais relevante do ponto de vista corporativo. Além de possuir o maior número de itens auditáveis, essa fase historicamente representa o maior esforço operacional da equipe de Engenharia de Software. Ainda assim, os agentes concluíram suas execuções em menos de 6 minutos nos três projetos avaliados.

A **Figura 3** apresenta a comparação entre o tempo médio necessário para execução exclusivamente manual das auditorias e o tempo médio obtido com a abordagem combinada entre atividades manuais e automatizadas por agentes. Os resultados demonstram redução consistente em todas as etapas avaliadas, com diminuição de 5,0 para 2,5 horas na Iniciação, de 7,0 para 3,5 horas na Execução e de 5,0 para 2,5 horas no Encerramento. Dessa forma, a utilização dos agentes contribuiu para uma redução média de aproximadamente 50% no tempo total das atividades analisadas.

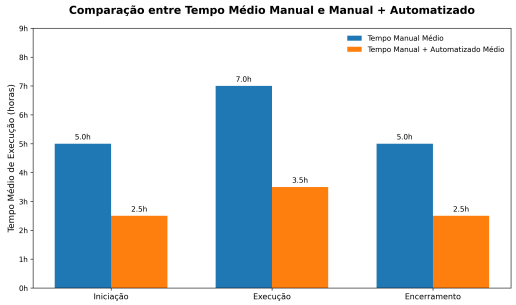

Figura 3: Comparação entre execução manual e execução automatizada das auditorias

A **Figura 4** apresenta a relação entre a quantidade de itens existentes nos checklists e a quantidade de ações executadas automaticamente pelos agentes. Além da redução de tempo, os resultados também demonstraram boa cobertura da automação em relação aos checklists utilizados nas auditorias. Nas auditorias de Iniciação e Encerramento, os agentes atingiram níveis de cobertura entre 88% e 100% dos itens previstos nos checklists.

Já na fase de Execução, os níveis de cobertura variaram entre 43% e 52%, refletindo a maior complexidade contextual e operacional dessa etapa. Grande parte deste resultado está relacionado à ausência de integração dos agentes com alguns ambientes e ferramentas necessários para obtenção de determinadas evidências. Em diversos itens do checklist, a avaliação depende da consulta de informações distribuídas em sistemas ainda não acessíveis pela solução implementada, reduzindo a capacidade de execução automatizada dessas verificações. Dessa forma, a menor cobertura observada não representa uma limitação do processo de auditoria em si, mas uma oportunidade de evolução das integrações da solução proposta.

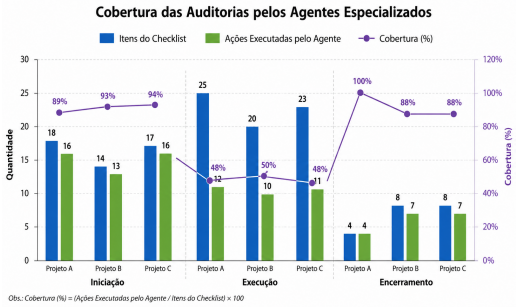

Figura 4: Cobertura das auditorias pelos agentes especializados

Além dos resultados quantitativos, também foram observados benefícios qualitativos relevantes, incluindo maior padronização dos comentários de auditoria, melhoria da rastreabilidade das evidências e redução do esforço operacional associado à navegação entre ferramentas e consolidação manual das informações.

Também foram observadas divergências pontuais entre as avaliações realizadas pelos agentes e as conclusões dos auditores. Essas situações ocorreram principalmente em verificações que exigiam

Tabela 1: Resultados da execução automatizada das auditorias

Fase	Projeto	Tempo Automatizado	Tempo Manual	Tempo Manual + Automatizado	Itens Checklist	Ações do Agente
Iniciação	Projeto A	6:06	4 a 6h	2 a 3h	18	16
	Projeto B	4:10	4 a 6h	2 a 3h	14	13
	Projeto C	3:43	4 a 6h	2 a 3h	17	16
Execução	Projeto A	4:37	6 a 8h	3 a 4h	25	12
	Projeto B	5:14	6 a 8h	3 a 4h	20	10
	Projeto C	4:49	6 a 8h	3 a 4h	23	11
Encerramento	Projeto A	2:24	4 a 6h	2 a 3h	4	4
	Projeto B	5:49	4 a 6h	2 a 3h	8	7
	Projeto C	3:09	4 a 6h	2 a 3h	8	7

maior interpretação contextual, resultando ocasionalmente em falsos positivos ou falsos negativos. Embora pouco frequentes, tais ocorrências reforçam a importância da validação final realizada pelos auditores.

Os resultados também demonstraram que agentes especializados apresentaram melhor desempenho em atividades relacionadas à verificação da conformidade dos artefatos e consolidação de evidências distribuídas em múltiplas ferramentas corporativas. Por outro lado, verificações dependentes de interpretação contextual e avaliação subjetiva ainda exigiram atuação humana significativa.

Apesar dos resultados positivos, a adoção da solução também evidenciou desafios importantes. A qualidade das respostas produzidas pelos agentes mostrou forte dependência da padronização dos artefatos e da qualidade das evidências registradas nas ferramentas corporativas. Projetos com documentação inconsistente, nomenclaturas não padronizadas ou ausência de rastreabilidade dificultaram a interpretação contextual realizada pelos agentes.

Observou-se também que a evolução dos modelos de linguagem utilizados pela plataforma contribuiu para respostas progressivamente mais coerentes e alinhadas ao contexto das auditorias.

Dessa forma, a supervisão humana continuou sendo essencial em análises relacionadas à tomada de decisão organizacional e à validação final das auditorias.

De forma geral, a experiência demonstrou que a integração entre Jira Automation e Rovo Agents possui forte potencial para apoiar auditorias de processo em ambientes reais, especialmente em organizações que já utilizam o ecossistema Atlassian.

6 Conclusão

Este trabalho apresentou um relato de experiência da indústria sobre a utilização do Atlassian Rovo integrado ao Jira Automation para apoiar auditorias semiautomatizadas do processo de software da empresa sob análise.

A solução proposta utilizou agentes especializados para interpretar regras do processo de software, coletar evidências em ferramentas corporativas e gerar comentários estruturados de auditoria diretamente no Jira. Os resultados demonstraram que auditorias anteriormente executadas manualmente em períodos entre 4 e 8 horas puderam ter sua execução automatizada concluída em poucos minutos.

Além da redução significativa do esforço operacional, também foram observados benefícios relacionados à padronização das auditorias, melhoria da rastreabilidade das evidências e centralização das informações no ecossistema Atlassian.

A experiência demonstrou que agentes de IA generativa possuem forte potencial para apoiar atividades de governança, conformidade e Engenharia de Software em ambientes corporativos. Entretanto, os resultados também reforçam que a supervisão humana continua sendo essencial em análises dependentes de interpretação contextual e tomada de decisão organizacional.

Como trabalhos futuros, pretende-se evoluir a solução para suportar abertura automática de ações corretivas, geração consolidada de indicadores organizacionais e análise histórica de conformidade entre projetos.

Disponibilidade de Artefatos

Os artefatos deste trabalho estão disponíveis em um repositório GitHub. O repositório contém um exemplo de prompt utilizado pelos agentes Atlassian Rovo durante a execução das auditorias e um exemplo do resultado gerado pela solução proposta. Github: https://github.com/eldorado-institute/CBSoft2026-Eldorado-artifacts

Agradecimentos

Os autores agradecem ao Instituto de Pesquisas Eldorado pelo suporte e incentivo ao desenvolvimento deste artigo.

A ferramenta *MS Copilot* (GPT 5.5) foi utilizada para apoiar a elaboração das Figuras 2, 3 e 4 e para auxiliar na revisão textual. Todo o conteúdo gerado ou revisado com apoio da ferramenta foi analisado, validado e aprovado pelos autores, que assumem integral responsabilidade pelo conteúdo final do artigo.

Referências

[1] Atlassian. 2026. Agents | Rovo | Atlassian Support. https://support.atlassian.com/rovo/docs/agents/. Accessed: 2026-05-14.
[2] Atlassian. 2026. Bitbucket Overview. https://bitbucket.org/product/guides/getting-started/overview. Accessed: 2026-05-14.
[3] Atlassian. 2026. Jira Automation: Basics & Common Use Cases. https://www.atlassian.com/software/jira/guides/automation/overview. Accessed: 2026-05-14.
[4] Atlassian. 2026. Software Development Tools and Services. https://www.atlassian.com/br/software. Accessed: 2026-05-15.
[5] eazyBI. 2026. eazyBI for Jira Reports, Charts, and Dashboards. Atlassian Marketplace. https://marketplace.atlassian.com/apps/1211051/eazybi-for-jira-reports-charts-and-dashboards. Accessed: 2026-05-14.
[6] Angela Fan et al. 2023. Large Language Models for Software Engineering: Survey and Open Problems. *arXiv preprint arXiv:2310.03533* (2023).
[7] Brian Fitzgerald and Klaas-Jan Stol. 2017. Continuous Software Engineering: A Roadmap and Agenda. *Journal of Systems and Software* 123 (2017), 176–189.
[8] Yifan Fu et al. 2024. Towards AI-Assisted Software Compliance and Governance. *IEEE Software* (2024).
[9] Xinyi Hou et al. 2023. Large Language Models for Software Engineering: A Systematic Literature Review. *arXiv preprint arXiv:2308.10620* (2023).
[10] Jez Humble and David Farley. 2010. *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation.* Addison-Wesley.
[11] Roger S. Pressman and Bruce R. Maxim. 2020. *Software Engineering: A Practitioner's Approach.* McGraw-Hill.
[12] Ian Sommerville. 2016. *Software Engineering.* Pearson.
[13] Jules White et al. 2023. A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT. *arXiv preprint arXiv:2302.11382* (2023).