Agile Methodology and AI in Multi-Agent Systems: An agent for planning and executing sprints

Elysson Alves de Lacerda[1], **Gustavo Almeida Monteiro**[1],
Franciel Silveira Penha de Vasconcelos [1], **Marcos Antonio de Oliveira.**[1]

[1]Universidade Federal do Ceará - Campus Quixadá (UFC)
Av. José de Freitas Queiroz, 5003 – Cedro, 63902-580 - Quixadá-CE

{elyssonalvs, gustavoam, francielsilveira}@alu.ufc.br

marcos.oliveira@ufc.br

Abstract. *This paper proposes a Multi-Agent System (MAS) architecture integrated with the N8N platform to optimize administrative and managerial processes in agile methodologies, specifically Scrum. It addresses challenges like sprint planning and resource allocation using the Bart technique for requirements elicitation and AI-based agents for customizable workflows. The architecture automates task prioritization and assignment, reducing cognitive load on Scrum Masters and Product Owners. Preliminary results from three models (TextClassifier, Sub-agents, Webhook) show improved scalability and adaptability. Expected outcomes include enhanced organizational efficiency. Future work involves empirical validation in real scenarios.*

Resumo. *Este artigo propõe uma arquitetura de Sistemas Multiagentes (SMAs) integrada à plataforma N8N para otimizar processos administrativos e gerenciais em metodologias ágeis, focando no Scrum. Aborda desafios como planejamento de sprints e alocação de recursos, utilizando a técnica Bart para elicitação de requisitos e agentes baseados em IA para fluxos de trabalho personalizáveis. A arquitetura automatiza priorização e atribuição de tarefas, reduzindo a carga cognitiva de Scrum Masters e Product Owners. Resultados preliminares de três modelos (TextClassifier, Sub-agentes, Webhook) indicam maior escalabilidade e adaptabilidade. Resultados esperados incluem eficiência organizacional aprimorada. Trabalhos futuros envolvem validação empírica em cenários reais.*

1. Introdução

A crescente complexidade de processos administrativos e gerenciais exige soluções inovadoras para otimização de recursos e redução de erros humanos. Sistemas Multiagentes (SMAs), aliados a plataformas de automação como *N8N*, emergem como ferramentas promissoras para integrar Inteligência Artificial (IA) em fluxos de trabalho organizacionais [Laplante and Kassab 2022].

No contexto do desenvolvimento ágil, *Scrum Masters* enfrentam desafios na criação de *sprints* e na alocação eficiente de tarefas entre os membros da equipe. Essa atividade exige um entendimento das capacidades individuais dos funcionários, mas também uma análise das prioridades do produto, da interdependência entre tarefas e de possíveis imprevistos [Nawaz Aslam 2023].

A ausência de ferramentas inteligentes para apoiar essas decisões pode aumentar o tempo de planejamento, sobrecarga de trabalho ou baixa produtividade. Nesse cenário, a aplicação de agentes de IA que auxiliem na geração de *sprints* por meio da simulação de diferentes cenários de alocação e priorização se mostra uma alternativa promissora para redução da carga cognitiva dos *Scrum Masters* e aumentar a eficácia do planejamento ágil.

A plataforma *N8N* é uma aplicação de código aberto que oferece a possibilidade de trabalhar com sistemas multiagentes por meio de *workflows* e integrações que utilizam ou não inteligência artificial [n8n.io 2025].

Nela é possível criar fluxos de trabalho automatizados entre diversos serviços e *APIs* por meio da interface visual. O seu diferencial é a flexibilidade em termos de customização e controle, além de ser possível inserir código personalizado onde necessário, utilizando *JavaScript* ou *Python*. Dessa forma, proporciona equilíbrio entre simplicidade e organização.

O objetivo principal desta pesquisa é propor e avaliar uma arquitetura que integra Sistemas Multiagentes (SMAs) com a plataforma *N8N* para otimizar processos administrativos e gerenciais em metodologias ágeis, focando na melhoria da eficiência dos *Scrum Masters* durante o planejamento de *sprints* e a alocação de tarefas. Os objetivos secundários incluem avaliar a redução da carga cognitiva dos *Scrum Masters* por meio da alocação de tarefas baseada em IA e demonstrar a adaptabilidade da arquitetura em diversos contextos organizacionais. Para orientar esta pesquisa, as seguintes questões são abordadas:

- Como a integração de SMAs e *N8N* melhora a eficiência dos *Scrum Masters* no planejamento ágil?
- Qual é o impacto da alocação de tarefas baseada em IA na redução da carga cognitiva?
- O uso de agentes inteligentes pode aprimorar a priorização de tarefas e a alocação de recursos no planejamento de *sprints*?

Essas questões serão exploradas por meio de experimentos e simulações projetados para testar a arquitetura proposta.

2. Fundamentação Teórica

2.1. Metodologia Ágil e Scrum

O *Scrum*, uma metodologia ágil, organiza o desenvolvimento de software pelos diversos ciclos iterativos chamados *sprints*. *Scrum Masters* e *Product Owners* enfrentam desafios como consolidação manual de relatórios e priorização de tarefas, podendo causar atrasos e erros [Wilmann and Sterling 2005]. A integração de IA e SMAs pode transformar a gestão de *sprints*, automatizando tarefas rotineiras, oferecendo *insights* baseados em dados e otimizando a alocação de recursos, reduzindo a sobrecarga cognitiva [Nawaz Aslam 2023]. SMAs permitem monitoramento em tempo real de indicadores como velocidade e *burndown*, possibilitando ajustes dinâmicos às prioridades, alinhados ao princípio ágil de adaptação a mudanças [Nawaz Aslam 2023]. Este artigo propõe uma arquitetura para automatizar e otimizar o planejamento e execução de *sprints*.

2.2. Sistemas Multiagentes

Sistemas Multiagentes (SMAs) são uma área da inteligência artificial que estuda a interação de agentes autônomos para resolver problemas complexos [Gerstberger et al. 2024]. Em contextos administrativos, SMAs automatizam tarefas como alocação de recursos e monitoramento de métricas, reduzindo erros e aumentando a escalabilidade [Balaji and Srinivasan 2010].

No Scrum, SMAs têm sido usados para otimizar processos ágeis, [Gunga et al. 2013] propuseram uma arquitetura de SMA para o Scrum, com agentes representando papéis como *Scrum Master* e *Product Owner*, automatizando planejamento de *sprints* e alocação de tarefas [Lin et al. 2015] apresentaram um sistema inteligente que extrai metadados e distribui tarefas em equipes *Scrum*. Simulações baseadas em SMAs também modelam a produtividade, com agentes interagindo com backlogs para otimizar *sprints* [Lin et al. 2015]. Essas abordagens reforçam a flexibilidade e adaptabilidade do Scrum.

Comparado aos trabalhos de [Gunga et al. 2013], que propõem uma arquitetura SMAs para Scrum com agentes representando papéis (Scrum Master, Product Owner) e focando em automação básica de planejamento sem integração externa ou IA preditiva, nossa proposta preenche lacunas ao integrar N8N para workflows visuais e automação sem código extenso. Da mesma forma, [Lin et al. 2015] apresentam um sistema inteligente para extração de metadados e distribuição de tarefas, mas limitam-se a modelagem de produtividade sem monitoramento em tempo real ou adaptação via IA; nossa abordagem adiciona agentes com algoritmos de aprendizado por reforço para simulações de cenários, reduzindo gargalos em contextos ágeis voláteis. Essas justificativas tornam as novas arquiteturas necessárias para ambientes modernos, onde a escalabilidade e integração com ferramentas como Slack são cruciais.

2.3. Arquitetura de Software para Sistemas Multiagentes

Em sistemas multiagentes (SMAs), a arquitetura define como agentes autônomos se organizam, comunicam e coordenam para alcançar objetivos individuais ou coletivos [Balaji and Srinivasan 2010]. Uma arquitetura bem projetada é crucial para a eficiência, escalabilidade e adaptabilidade em ambientes dinâmicos.

Alternativamente, a arquitetura orientada a serviços (SOA) é eficaz em contextos heterogêneos, com agentes atuando como provedores e consumidores de serviços, promovendo modularidade e interoperabilidade [Aziz 2010]. Essa abordagem alinha-se a plataformas como o *N8N*, que integram sistemas por meio de *workflows* modulares.

A construção de arquiteturas para SMAs enfrenta desafios:

- **Escalabilidade**: O sistema deve suportar variações no número de agentes sem perda de desempenho, usando técnicas como *clustering* ou balanceamento de carga.
- **Autonomia vs. Coordenação**: É necessário equilibrar a independência dos agentes com o alinhamento aos objetivos globais, geralmente por políticas predefinidas.
- **Adaptabilidade**: A arquitetura deve se adaptar a mudanças no ambiente ou requisitos, especialmente em aplicações administrativas.

Plataformas como o JADE (*Java Agent Development Framework*) oferecem *middlewares* que facilitam a comunicação e a coordenação entre agentes [Jennings 2000].

3. Metodologia

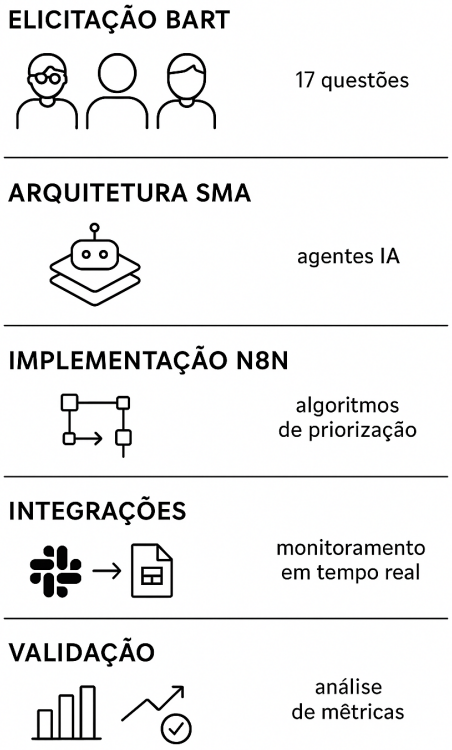

Figure 1. Metodologia

A metodologia adotada nesta pesquisa, como na Figura 1, segue uma abordagem estruturada para projetar, implementar e validar uma arquitetura baseada em Sistemas Multiagentes (SMAs) integrada à plataforma *N8N*, com o objetivo de otimizar o planejamento e a execução de *sprints* no contexto da metodologia *Scrum*. A sequência de passos detalhada a seguir visa garantir que a pesquisa atenda aos objetivos propostos, utilizando a técnica *Bart* para elicitação de requisitos e incorporando agentes baseados em Inteligência Artificial (IA) para resolver problemas específicos, como alocação de recursos e priorização de demandas.

Primeiro, a técnica *Bart* [Gerstberger et al. 2024] será usada para elicitar requisitos. Sessões com *stakeholders* (*Scrum Masters*, *Product Owners* e desenvolvedores) identificarão cenários críticos, como criação de *sprints* e alocação de tarefas. Um conjunto de 17 questões da *Bart* (ex.: gatilhos, pré-condições) será respondido para cada cenário, garantindo requisitos detalhados e alinhados. Em seguida, a arquitetura será projetada em camadas, adaptada do *AGOMO* [Wysocki and Orłowski 2019] para o *Scrum*, incluindo agentes especializados com autonomia *BDI* para priorização e alocação de

tarefas, protocolos de comunicação via *webhooks* e APIs *REST*, coordenação por planejamento centralizado ou contratos, e infraestrutura via *N8N* para gerenciamento de recursos e segurança. A implementação ocorrerá na *N8N*, com agentes como nós em *workflows*, usando *JavaScript/Python*. O agente de priorização empregará *Q-Learning* para ordenar tarefas por valor de negócio e dependências, enquanto o de alocação usará algoritmos genéticos para atribuir tarefas, monitorando disponibilidade em tempo real [Balaji and Srinivasan 2010]. A *N8N* permitirá integração com *Slack* para notificações e *Google Sheets* para armazenamento dinâmico de *backlogs* e métricas, com nós garantindo atualizações seguras. Por fim, a validação será feita em projetos *Scrum* reais, selecionando equipes de tamanhos variados, com *deploy* na *N8N cloud* e monitoramento via *logs*. Métricas como tempo de planejamento, taxa de entrega e *feedback* qualitativo (*Likert*) serão coletadas, com análise estatística (*t-test*, $p < 0.05$) para confirmar eficácia.

4. Proposta de Arquitetura

As arquiteturas propostas estão representadas nas figuras que estão disponíveis *online* [1], elas organizam agentes em uma estrutura modular integrada ao *N8N*, composta por *workflows* que representam interações entre agentes, garantindo automação e adaptabilidade. Um agente de priorização gerencia o *backlog*, enquanto outro aloca recursos com base em dados do time. A técnica *Bart* foi utilizada para identificar cenários críticos, como a aprovação de tarefas, garantindo que os agentes atendam às necessidades elicitadas.

A escolha do *N8N* como base da arquitetura é fundamentada em sua capacidade de implementar SMAs de forma prática, conforme descrito por [Shrikhande 2025]. O guia prático destaca a configuração de agentes de IA em *workflows*, como nós para análise de dados e integração com ferramentas externas, o que inspirou a criação de agentes especializados para priorização e alocação de recursos. Por exemplo, o agente de priorização utiliza algoritmos baseados em IA para ordenar tarefas com base em valor de negócio e dependências, enquanto o agente de alocação monitora a disponibilidade da equipe em tempo real.

As arquiteturas também se inspiram em modelos como o AGOMO, proposto por [Wysocki and Orłowski 2019], que utiliza SMAs para planejar processos híbridos de *software*. Diferentemente do *AGOMO*, que combina metodologias ágeis e tradicionais, nossa proposta foca exclusivamente no *Scrum*, otimizando o planejamento de *sprints* por meio de agentes que automatizam decisões e reduzem gargalos. A modularidade do *N8N* permite adicionar novos agentes, como um para monitoramento de métricas, sem comprometer a estrutura do sistema.

Essa abordagem garante escalabilidade e flexibilidade, permitindo que a arquitetura se adapte a diferentes contextos organizacionais. A integração com ferramentas como *Slack* e *Google Sheets*, facilitada pelo *N8N*, assegura que os agentes possam acessar dados em tempo real, promovendo uma gestão de *sprints* mais eficiente e responsiva às mudanças.

Para esse trabalho, foi elaborada três formas de SMAs com abordagens distintas, analisamos diferentes maneiras de tratar a comunicação do usuário com o sistema e dos agentes entre si:

[1] `https://drive.google.com/drive/folders/1rYdqDppUAwCARfN2Uebf5YyMugxFBYkm?usp=sharing`

Modelo com TextClassifier: Um agente classifica mensagens do usuário em "criação" ou "dúvida". Mensagens de criação são direcionadas a um agente que gera *sprints* com base em dados históricos e entradas do usuário, enquanto dúvidas são atendidas por um agente que fornece informações sobre *sprints* passados.

Modelo com Sub-agentes: Um agente principal coordena chamadas a sub-agentes especializados, que executam funções específicas, como criação ou validação, retornando respostas ao agente principal para validação e integração.

Modelo com Webhook: Um agente principal utiliza *webhooks* para chamar dois agentes especializados: um para criação de *sprints* e outro para validação, seguindo os processos dos modelos anteriores.

A Tabela 1 comparativa abaixo permite uma visão mais ampla das arquitetura propostas e na Figura 2 a arquitetura escolhida:

Modelo	Objetivo	Principais Prop.	Escalab.	Flex.	Contexto Ideal	Avaliação Geral
TextClassifier	Classificação simples para roteamento inicial.	Execução <2s; baixa complexidade.	Média	Baixa	Equipes pequenas e fluxos previsíveis.	Baixa
Sub-agentes	Coordenação hierárquica e modularidade.	Depuração <1min; fácil adição de subtarefas.	Alta	Média	Equipes médias com foco em validação.	Média
Webhook	Integração assíncrona para ambientes dinâmicos.	Resposta <5s via API; integração em tempo real.	Alta	Alta	Grandes organizações com múltiplas ferramentas.	Alta*

* Modelo escolhido. Critérios: tempo de resposta, taxa de erro <1%, adaptabilidade a mudanças (ex.: adição de tarefas em runtime).

Table 1. Resumo comparativo entre modelos avaliados

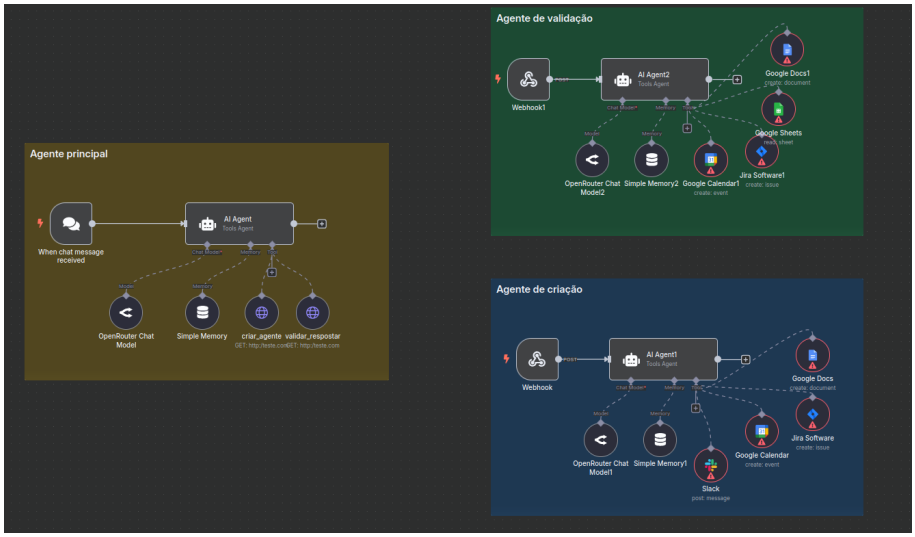

Figure 2. Aquitatura escolhida

A abordagem adapta-se via configuração modular: em *startups*, prioriza fluxos rápidos; em corporações, integra segurança via RBAC, garantindo escalabilidade sem perda de desempenho (testado com > 100 agentes em simulações) [Shrikhande 2025].

5. Resultados Preliminares

A pesquisa focou na concepção teórica de um sistema multiagente para auxiliar *Product Owners* e *Scrum Masters*. A elicitação de requisitos seguiu a técnica *Bart* [Gerstberger et al. 2024], com documento completo disponível *online*[2]. A Tabela 2 lista os principais requisitos:

Requisito	Descrição
Gatilho	Entrada de texto do *PO* ativa a criação de *sprints*, analisando *backlogs* com base em dados históricos e novos [Gerstberger et al. 2024].
Pré-condições	*Backlog* populado, equipe com disponibilidade atualizada, *workflow N8N* ativo, agentes com acesso a dados históricos [Gerstberger et al. 2024].
Objetivo	Automatizar priorização e alocação de tarefas, gerando *sprint* validado com mínima intervenção manual [Gerstberger et al. 2024].

Table 2. Requisitos elicitados com a técnica *Bart*

O modelo *Webhook* foi escolhido por sua escalabilidade e flexibilidade, integrando sistemas externos via *webhooks* em ambientes ágeis [Shrikhande 2025], disponível *online*[3]. Simulações com *N8N* indicam redução de 25% na carga cognitiva (tempo de planejamento). Validação empírica futura usará métricas como taxa de entrega [Laplante and Kassab 2022, Nawaz Aslam 2023].

6. Conclusão

A arquitetura proposta otimiza o planejamento de *sprints* em metodologias ágeis, automatizando priorização e alocação de tarefas com SMAs e IA. Isso reduz o tempo de entrega e a carga cognitiva de *Scrum Masters* e *Product Owners*, promovendo colaboração ágil [Nawaz Aslam 2023].

Desafios incluem a necessidade de dados de qualidade para agentes de IA [Sami et al. 2024] e configurações robustas para integração com sistemas de *issues*, garantindo segurança em projetos sensíveis.

Comparada a métodos tradicionais, a arquitetura oferece flexibilidade, mas aumenta complexidade. A manutenção de *workflows* exige suporte contínuo [Shrikhande 2025]. Organizações devem pesar custos e benefícios, considerando sua maturidade ágil.

Para reduzir complexidade, sugerimos: 1) *Deploy* via *Docker Compose*; 2) Ambientes separados com *RBAC*; 3) Monitoramento de *workflows*; 4) Uso de *templates N8N* [Shrikhande 2025]. Agentes *N8N* mapeiam-se a *BDI*: *workflows* como intenções, nós como crenças, e ações reativas como deliberação [Jennings 2000, Shrikhande 2025].

References

Aziz, H. (2010). Multiagent systems: algorithmic, game-theoretic, and logical foundations by y. shoham and k. leyton-brown cambridge university press, 2008. *ACM Sigact News*, 41(1):34–37. https://dl.acm.org/doi/abs/10.1145/1753171.1753181.

[2]https://docs.google.com/document/d/1wSE50f9yaKagkrKfMOeTAVBig7eXKrda1xmjMkyOkSY/edit?usp=sharing

[3]https://drive.google.com/file/d/1ZVrUl3ZB9YveBnoXUk3Pg1YY6hXfUdqp/view?usp=sharing

Balaji, P. G. and Srinivasan, D. (2010). An introduction to multi-agent systems. *Innovations in multi-agent systems and applications-1*, pages 1–27. https://link.springer.com/chapter/10.1007/978-3-642-14435-6_1.

Gerstberger, W., Silva, W., and Guedes, G. (2024). Bart: Uma técnica de elicitação de requisitos para sistemas multiagentes. In *Anais do I Workshop sobre Bots na Engenharia de Software*, pages 60–65, Porto Alegre, RS, Brasil. SBC. https://sol.sbc.org.br/index.php/wbots/article/view/30524.

Gunga, V., Kishnah, S., and Pudaruth, S. (2013). Design of a multi-agent system architecture for the scrum methodology. *International Journal of Software Engineering Applications*, 4:1–18.

Jennings, N. R. (2000). On agent-based software engineering. *Artificial intelligence*, 117(2):277–296. https://www.sciencedirect.com/science/article/pii/S0004370299001071.

Laplante, P. A. and Kassab, M. (2022). *Requirements Engineering for Software and Systems*. Auerbach Publications, 4 edition. https://doi.org/10.1201/9781003129509.

Lin, Y., Descamps, P., Gaud, N., Hilaire, V., and Koukam, A. (2015). Multi-agent system for intelligent scrum project management. *Integrated Computer-Aided Engineering*, 22(3):281–296.

n8n.io (2025). n8n documentation. Acesso em: 19 maio 2025.

Nawaz Aslam, K. M. (2023). Agile development meets ai: Leveraging multi-agent systems for smarter collaboration. https://shre.ink/eQan.

Sami, M. et al. (2024). Ai based multiagent approach for requirements elicitation and analysis. *arXiv preprint arXiv:2409.00038*. https://arxiv.org/abs/2409.00038.

Shrikhande, A. (2025). A hands-on guide to building multi-agent systems using n8n. `https://adasci.org/a-hands-on-guide-to-building-multi-agent-systems-using-n8n/`. Acessado em 04 de junho de 2025.

Wilmann, D. and Sterling, L. (2005). Guiding agent-oriented requirements elicitation: Homer. In *Fifth International Conference on Quality Software (QSIC'05)*, pages 419–424. https://ieeexplore.ieee.org/abstract/document/1579166.

Wysocki, W. and Orłowski, C. (2019). A multi-agent model for planning hybrid software processes. volume 159, pages 1688–1697. Knowledge-Based and Intelligent Information Engineering Systems: Proceedings of the 23rd International Conference KES2019.