

Model Context Protocol e Agentes de Inteligência Artificial: Arquiteturas, Integração e Desafios para Sistemas Autônomos

Marcos Vinicius Tavares¹, Thiago Winicius Storqui¹, Gabriel Motta Becker¹,
João Vitor Rosário Marquardt¹, Nicolas Heling¹, Pedro Henrique Schutz¹,
Jefferson Vorpapel¹

¹ Faculdade Donaduzzi – 85.920-025 – Toledo – PR – Brasil

{marcos.tavares, thiago.storqui, gabriel.becker, joao.marquardt,
nicolas.heling, pedro.schutz}@alunos.bpkedu.com.br
jefferson.vorpapel@bpkedu.com.br

Abstract. *This paper reviews the Model Context Protocol (MCP), introduced by Anthropic in November 2024, which standardizes communication between large language models (LLMs) and external tools via JSON-RPC 2.0, reducing the $N \times M$ integration problem to $O(N+M)$. The paper analyzes MCP's architecture, the ReAct-based agentic loop, security challenges identified in 2025, and adoption by OpenAI, Google DeepMind, and Microsoft. Results include an empirical experiment confirming $O(N+M)$ complexity reduction, showing that MCP transforms language models into interoperable cognitive agents enabling complex multi-step workflows across heterogeneous systems.*

Resumo. *Este artigo revisa o Model Context Protocol (MCP), lançado pela Anthropic em novembro de 2024, que padroniza a comunicação entre grandes modelos de linguagem (LLMs) e ferramentas externas via JSON-RPC 2.0, reduzindo o problema $N \times M$ para $O(N+M)$. O artigo analisa a arquitetura do MCP, o laço agêntico baseado no paradigma ReAct, os desafios de segurança de 2025 e a adoção pelo ecossistema. Os resultados incluem um experimento empírico confirmando a redução $O(N+M)$ na prática, demonstrando que o MCP transforma modelos isolados em agentes cognitivos interoperáveis, viabilizando fluxos de trabalho complexos em sistemas heterogêneos.*

1. Introdução

Os Grandes Modelos de Linguagem (LLMs) revolucionaram a Inteligência Artificial, mas sua natureza estática impõe restrições críticas a aplicações que exigem interação dinâmica com sistemas externos e acesso a dados em tempo real [Lewis et al. 2020]. Para superar essas limitações, emergiu o conceito de agente de IA: um sistema que envolve um LLM em um ciclo de raciocínio e ação, capaz de planejar, tomar decisões e invocar ferramentas externas conforme a necessidade da tarefa [Yao et al. 2023]. Nesse contexto, a proliferação de integrações proprietárias gerou o chamado problema $N \times M$: à medida que aumentam o número de modelos e de ferramentas disponíveis, cresce de forma quadrática a quantidade de conectores necessários, tornando sua manutenção complexa e pouco escalável.

Como resposta, o Model Context Protocol (MCP), lançado pela Anthropic em novembro de 2024, propõe uma interface aberta baseada em JSON-RPC 2.0 e inspirada no Language Server Protocol, permitindo que agentes de IA se comuniquem com diferentes ferramentas sem conectores proprietários [Anthropic 2024]. Este trabalho investiga: *de que forma o MCP contribui para reduzir a complexidade de integração entre agentes de IA e ferramentas externas, favorecendo sistemas mais interoperáveis e escaláveis?*

2. Objetivo

O objetivo deste trabalho é analisar o Model Context Protocol (MCP) como infraestrutura de comunicação para sistemas de agentes de Inteligência Artificial, descrevendo sua arquitetura, seu funcionamento no laço de execução agêntica, os recursos expostos por seus servidores, os principais desafios de segurança e suas perspectivas de adoção em arquiteturas multiagente escaláveis. Adicionalmente, busca-se validar empiricamente a redução de complexidade de $O(N \times M)$ para $O(N+M)$ por meio de um experimento comparativo entre integração tradicional e integração via MCP, medindo linhas de código e latência de execução.

3. Referencial Teórico

3.1. Agentes de IA e o Paradigma ReAct

Um agente de IA percebe seu ambiente, raciocina e executa ações para atingir objetivos predefinidos [Russell e Norvig 2020]. O paradigma *ReAct* (*Reasoning + Acting*) formaliza o laço agêntico: o modelo alterna entre raciocínio e ação, invocando ferramentas externas. Estudos empíricos demonstraram que agentes ReAct superam abordagens de raciocínio puro, pois a intercalação reduz alucinações ao ancorar inferências em evidências recuperadas dinamicamente [Yao et al. 2023].

3.2. Arquitetura do Model Context Protocol

O MCP define três camadas: host (ambiente de execução), cliente MCP (gerencia conexões com múltiplos servidores) e servidores MCP (processos leves que expõem capacidades via protocolo padronizado por stdio ou SSE/HTTPS). A camada de transporte usa JSON-RPC 2.0, suportando chamadas síncronas, notificações assíncronas e streaming. Assim como o LSP reduziu de $N \times M$ para $N+M$ as integrações em editores de código, o MCP elimina a fragmentação de conectores no ecossistema de agentes [Anthropic 2024].

3.3. Recursos Expostos por Servidores MCP

Servidores MCP expõem três categorias: ferramentas (*tools*), funções executáveis descobertas em tempo de execução via *handshake*; recursos (*resources*), dados estáticos ou dinâmicos; e templates de prompt, pré-construídos no servidor. O modelo raciocina sobre o catálogo de schemas JSON para decidir quais ferramentas invocar, em qual sequência e com quais parâmetros [Robison 2025].

4. Metodologia

Este trabalho combina revisão bibliográfica sistemática com experimento empírico comparativo. A revisão foi conduzida nas bases arXiv, IEEE Xplore e Google Acadêmico,

complementadas por documentações técnicas da Anthropic, relatórios do BCG e publicações da Equinix, cobrindo novembro de 2024 a abril de 2026. Foram identificados 47 documentos candidatos, dos quais 12 foram selecionados após aplicação dos critérios de inclusão: (a) tratar diretamente do MCP ou arquiteturas agênticas baseadas em LLMs; (b) disponibilidade de texto completo; (c) publicados após novembro de 2024. Documentos duplicados, sem relação com o tema central ou sem autoria identificada foram excluídos. A análise comparativa foi estruturada em quatro dimensões: (1) complexidade de integração; (2) interoperabilidade; (3) composição multiagente; e (4) perfil de riscos de segurança.

Para a validação empírica, foi conduzido um experimento comparativo entre duas abordagens de integração com $N=2$ modelos (Llama3 e Mistral, executados localmente via Ollama) e $M=2$ ferramentas (consulta a banco de dados SQLite e leitura de arquivos). A Abordagem A implementou integração tradicional, com conectores específicos por modelo. A Abordagem B utilizou um servidor MCP único via JSON-RPC 2.0 e cliente genérico reutilizável. As métricas coletadas foram: linhas de código (LOC), número de funções de conector criadas e latência de execução em milissegundos. Os scripts foram implementados em Python 3.11 em ambiente Windows, sem uso de APIs externas.

5. Resultados

5.1. O Laço Agêntico e Casos de Uso

A adoção do MCP transforma sistemas de IA de respondedores de consultas isoladas em agentes executores de tarefas multietapas. O laço típico envolve: (1) recebimento de tarefa pelo host; (2) descoberta dinâmica de ferramentas; (3) raciocínio sobre qual ferramenta invocar; (4) execução da chamada JSON-RPC; (5) incorporação do resultado ao contexto; e (6) repetição até conclusão [Yao et al. 2023, Anthropic 2024]. Casos documentados incluem agentes de gestão de projetos integrando Asana e Jira, agentes de engenharia executando testes e criando pull requests via GitHub MCP, e agentes de suporte consultando bases de conhecimento automaticamente [BCG – Boston Consulting Group 2025].

5.2. MCP versus Integrações Proprietárias

No modelo $N \times M$ tradicional, 10 modelos e 50 ferramentas demandam até 500 conectores proprietários. Com o MCP, o mesmo ecossistema requer apenas 10 clientes e 50 servidores, redução de $O(N \cdot M)$ para $O(N+M)$, traduzindo-se em menor custo de desenvolvimento e menor risco de incompatibilidades. A padronização também viabiliza sistemas multiagente hierárquicos, de modo análogo à arquitetura orientada a serviços (SOA) na camada de raciocínio de LLMs [Robison 2025].

5.3. Validação Empírica: LOC e Latência

Os resultados do experimento comparativo são apresentados nas Tabelas 1 e 2.

Na abordagem tradicional, conectar $N=2$ modelos a $M=2$ ferramentas exigiu 4 funções de conector dedicadas com duplicação explícita de lógica. Na abordagem MCP, o mesmo cenário foi implementado com 2 componentes independentes: servidor (143 linhas) e cliente genérico (171 linhas). O ponto central é a escalabilidade: adicionar um terceiro modelo ao sistema MCP requer zero linhas adicionais nas ferramentas, enquanto

Tabela 1. Complexidade de código: Tradicional vs. MCP (N=2, M=2)

Métrica	Tradicional $O(N \cdot M)$	MCP $O(N+M)$
LOC total	198	314
Funções de conector	$N \times M = 4$	$M = 2$
Código duplicado por modelo	Sim	Não
LOC extra para 3º modelo	~99 linhas	0 linhas

Tabela 2. Latência de execução por modelo (ms)

Modelo	Tradicional	MCP	Diferença
Llama3	108.073ms	47.608ms	-60.465ms
Mistral	114.965ms	41.208ms	-73.757ms
Média	111.519ms	44.408ms	-67.111ms

na abordagem tradicional exigiria $M=2$ novos conectores (~99 linhas), confirmando empiricamente a redução $O(N \cdot M) \rightarrow O(N+M)$. A diferença de latência é atribuída principalmente ao tamanho do contexto enviado, e não ao *overhead* do JSON-RPC 2.0, que opera em escala de milissegundos para comunicação local via socket TCP.

5.4. Ecossistema, Adoção e Segurança

A velocidade de adoção do MCP é notável: OpenAI em março de 2025, Google DeepMind e Microsoft na sequência, com integração ao GitHub Copilot e Microsoft 365 Copilot. Em dezembro de 2025, a governança foi transferida para a AAIF sob a Linux Foundation [Linux Foundation 2025]. Em termos de segurança, dois vetores foram identificados: injeção de prompt via ferramentas (*tool prompt injection*) e substituição silenciosa de ferramentas (*tool shadowing*). A especificação MCP 1.1 incorporou autenticação OAuth 2.0 como mitigação [Anthropic 2025].

6. Conclusão

O Model Context Protocol representa uma inflexão arquitetural relevante ao padronizar a comunicação entre LLMs e ferramentas externas via JSON-RPC 2.0. O experimento empírico conduzido confirmou que a abordagem MCP elimina a duplicação de código de integração: enquanto a abordagem tradicional exige M novos conectores para cada modelo adicionado, o MCP mantém custo constante, validando a redução de $O(N \cdot M)$ para $O(N+M)$. Entretanto, o protocolo apresenta limitações relevantes: a ausência de mecanismos nativos de controle de acesso granular, a dependência de implementações corretas por parte dos servidores e a ausência de padrões consolidados para ambientes multiagente distribuídos são pontos que requerem atenção. A adoção em larga escala amplia a superfície de ataque, exigindo validação criptográfica, isolamento de ambientes e princípio do menor privilégio. No contexto brasileiro, o MCP apresenta oportunidades na integração com ERPs nacionais, plataformas fiscais e sistemas regulatórios. Como trabalhos futuros, recomenda-se a avaliação empírica com N e M maiores, considerando métricas de custo por tarefa, taxa de sucesso e desempenho em fluxos operacionais complexos.

Referências

Anthropic (2024). Introducing the model context protocol. <https://www.anthropic.com/news/model-context-protocol>. Acesso em: abr. 2026.

- Anthropic (2025). Model context protocol specification — version 1.1. <https://spec.modelcontextprotocol.io>. Acesso em: abr. 2026.
- BCG – Boston Consulting Group (2025). Put AI to work faster using model context protocol. <https://www.bcg.com/publications/2025/put-ai-to-work-faster-using-model-context-protocol>. Acesso em: abr. 2026.
- Lewis, P. et al. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 9459–9474.
- Linux Foundation (2025). Anthropic transfers model context protocol governance to the agentic AI foundation under the Linux foundation. <https://www.linuxfoundation.org/press/anthropic-transfers-model-context-protocol-governance>. Acesso em: abr. 2026.
- Robison, G. (2025). The model context protocol: The architecture of agentic intelligence. <https://gregrobison.medium.com>. Acesso em: abr. 2026.
- Russell, S. e Norvig, P. (2020). *Artificial Intelligence: A Modern Approach*. Pearson, Hoboken, 4 edition.
- Yao, S. et al. (2023). ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.