Relatório 09 - Leitura: Construindo Agentes Google ADK(III)

Lucas Augusto Nunes de Barros

Descrição da atividade

O card propõe duas leituras, uma delas sobre o que é importante para um engenheiro de IA saber sobre protocolos de comunicação de agentes, a outra é uma notícia sobre o lançamento de ferramentas da Google voltadas para a área de modelos de inteligência artificial.

1. Google ADK e o Protocolo A2A

Uma notícia veiculada pelo Canal Tech, apresenta na sua reportagem novas ferramentas do Google que reforçam uma das principais tendências de IA, o desenvolvimento de agentes inteligentes. Uma dessas ferramentas é o Agent Development Kit (ADK), um pacote open source, desenvolvido em Python e Typecript para oferecer flexibilidade ao criar agentes simples até fazer orquestrações complexas em sistemas com multi agentes. O ADK destaca-se por ser independente ao modelo, permitindo que os desenvolvedores usem os modelos de sua preferência, além de permitir a integração com soluções diretamente no Vertex AI do Google Cloud.

Para complementar o desenvolvimento, o Google anunciou a integração do ADK com o Agent Engine, um conjunto de serviços que permite aos desenvolvedores implementar, gerenciar e escalonar agentes em ambientes de produção. Outra novidade do Google foi o lançamento do AI Agent Marketplace, uma plataforma digital centralizada onde empresas e usuários podem comprar e vender seus agentes.

No campo da interoperabilidade, o protocolo Agent-to-Agent (A2A) foi o destaque, desenvolvido para eliminar barreiras entre ecossistemas distintos. Com o apoio de mais de 50 parceiros o A2A permite que agentes usando modelos diferentes publiquem as suas capacidades e negociem a forma de interação autonomamente entre si, garantindo uma comunicação fluida e assertiva.

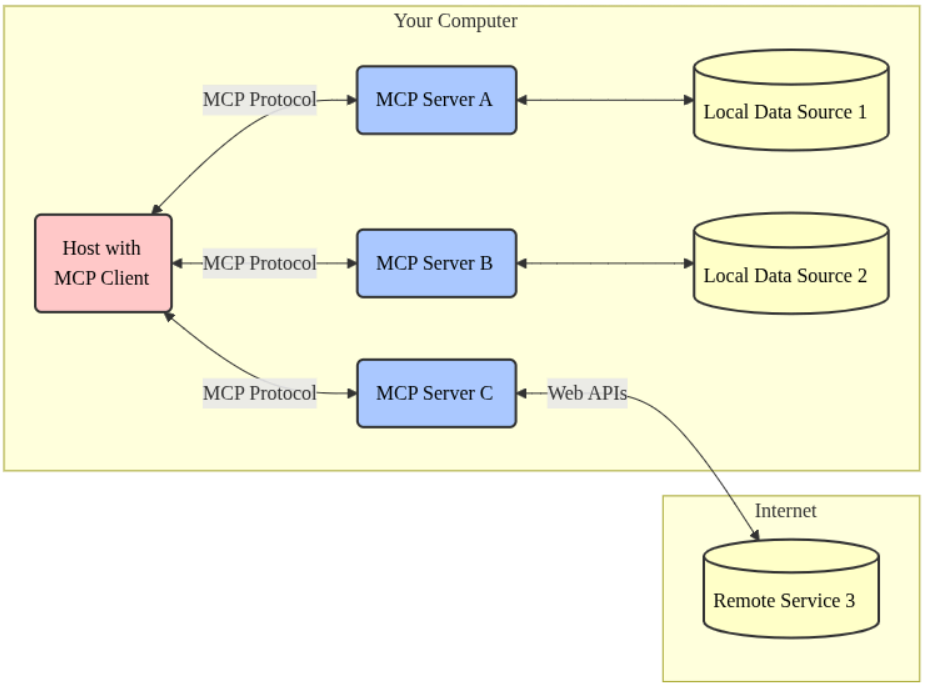
2. Protocolos de Comunicação para Agente de IA

Esse tópico é abordado em uma publicação da medium.com, ela traz como os principais protocolos de IA da atualidade ajudam os agentes a conversar, pensar e trabalhar juntos.

2.1 MCP - Model Context Protocol

O Model Context Protocol (MCP), desenvolvido pela Anthropic, atua conectando modelos de IA às ferramentas e dados disponíveis. A sua principal função é padronizar a conexão e o recebimento de contexto estruturado por parte das LLMs, além de padronizar também a invocação de ferramentas de forma dinâmica, permitindo aos modelos executar tarefas de forma mais independente.

Fluxograma de Operação do Protocolo MCP



2.1.1 Funcionalidades do MCP

- Injeção de Dados Contextuais: O MCP permite integrar recursos externos diretamente via prompt ou direto na memória do modelo. Esse processo é realizado em uma interface padrão, garantindo que o modelo permaneça eficiente e com uma arquitetura limpa.

- Encaminhamento e Execução de Funções: O protocolo permite os modelos acionar ferramentas de forma dinâmica. É possível fornecer capacidades específicas (as tools do agente) e o LLM poderá invocá-las conforme necessário.

- Orquestração de Prompts: Em vez de sobrecarregar o prompt com todos os detalhes possíveis, o MCP auxilia na compilação apenas do contexto que é estritamente relevante. Resultando em um contexto mais inteligente, menor consumo de tokens e resultados de maior qualidade.

2.1.2 Características de Implementação

Em termos de implementação, o protocolo opera sobre HTTP(S) utilizando JSON e foi desenhado para ser independente ao modelo, permitindo que qualquer LLM com um ambiente de execução compatível use servidores MCP. A sua arquitetura assegura ainda a compatibilidade com gateways de API e suporta padrões de autenticação como OAuth2 e mTLS.

2.1.3 Casos de Uso

- Integração de LLMs com APIs Internas: Viabiliza o acesso seguro a dados de negócio estruturados sem a necessidade de expor diretamente os endpoints.

- Agentes Corporativos: Instrumentaliza agentes autônomos com contexto em tempo real.

- Construção Dinâmica de Prompts: Permite a personalização de prompts com base na sessão do utilizador, no estado atual do sistema ou na lógica do fluxo de tarefas.

2.2 ACP - Agent Communication Protocol

O Agent Communication Protocol (ACP), proposto pela BeeAI e IBM, foi desenvolvido com a premissa de ser "local-first", ou seja, orientado para conexões locais. Pensado para otimizar a coordenação de agentes que operam no mesmo ambiente ou em borda, sem depender de conexões com nuvem. Este protocolo é ideal para cenários de baixa latência e alta privacidade, onde os agentes utilizam um barramento local para interagirem diretamente, garantindo privacidade dos dados e execução eficiente.

2.2.1 Arquitetura do Protocolo

O ACP define um ambiente de agentes descentralizado caracterizado pelo seguinte:

- Descoberta e Identidade: Cada agente anuncia a sua identidade, capacidades e estado utilizando uma camada local de transmissão (broadcast) e descoberta.

- Comunicação: Os agentes comunicam através de mensagens orientadas a eventos, frequentemente utilizando um barramento local ou sistema de IPC (comunicação entre processos).

- Orquestração (Opcional): Um controlador de execução (runtime) pode ser utilizado para orquestrar o comportamento dos agentes, agregar telemetria e impor políticas de execução.

- Estrutura Operacional: Tipicamente, os agentes ACP operam como serviços ou contêineres leves e sem estado (stateless), partilhando uma infraestrutura de comunicação comum.

2.2.2.Características de Implementação

- Ambientes de Baixa Latência: Projetado para cenários que exigem resposta rápida.

- Flexibilidade de Transporte: A sua implementação pode ser realizada sobre protocolos eficientes como gRPC, ZeroMQ ou barramentos de execução (runtime buses) personalizados.

- Soberania Local: Foca na independência do sistema, eliminando a necessidade de conexão com nuvem ou serviços externos.

- Roteamento Inteligente: Permite a delegação automatizada de tarefas com base nas competências de cada sub agente.

2.2.3 Casos de Uso

- Orquestração Multi-Agente em Dispositivos de Borda: Facilita a gestão de agentes em dispositivos de borda (edge devices), como drones, clusters de IoT e frotas autônomas.

- Sistemas de LLM com Prioridade Local (Local-First): Coordena as chamadas ao modelo, a leitura de informações externas e a atuação por meio de ações diretas.

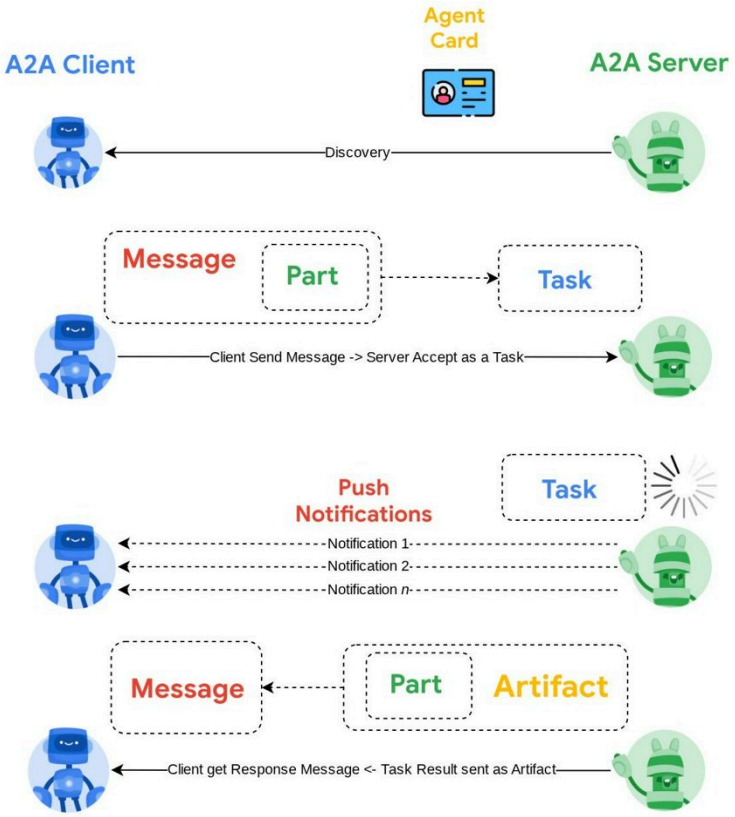
- Ambientes de Execução Autônomos: Permite que agentes operem e se coordenem de forma independente, sem qualquer necessidade de infraestrutura de nuvem centralizada.

2.3 A2A - Agent-to-Agent Protocol

O Agent-to-Agent Protocol (A2A), criado pela Google, foca na interoperabilidade horizontal através da internet, permitindo que agentes de diferentes sistemas, fornecedores ou plataformas colaborem e deleguem tarefas entre si.

Ao contrário do foco local do ACP, o A2A utiliza o padrão web (HTTP e JSON) para permitir que agentes exponham as suas identidades e capacidades através dos "Cartões de Agente", permitindo a orquestração de fluxos de trabalho complexos em ecossistemas distribuídos.

Forma de Operação do Protocolo A2A



2.3.1 Componentes

- Cartões de Agente: São arquivos JSON que informam características do agente, descrevendo suas capacidades, endpoints, mensagens suportadas, métodos de autenticação e metadados de execução.

- Interface Cliente/Servidor A2A: Oferece flexibilidade permitindo que cada agente possa atuar como cliente (iniciando uma tarefa) ou como servidor (executando a tarefa solicitada). Isso permite a delegação dinâmica de tarefas e uma interação mais fluida.

- Troca de Mensagens: O sistema suporta tarefas complexas divididas em subtasks, mantendo o contexto. Inclui capacidades para saídas em tempo real e a troca de estados persistentes, como ficheiros ou fragmentos de conhecimento.

- Negociação de Experiência: Os agentes podem adaptar o formato de suas mensagens, a granularidade do conteúdo e outras características para corresponderem exatamente às expectativas do agente receptor, garantindo uma compatibilidade entre diferentes modelos.

2.3.2 Arquitetura de segurança

- Autenticação Padrão: O acesso é gerido através de protocolos de segurança robustos e amplamente adotados, como OAuth 2.0 e autorização baseada em chaves de API.

- Omissão de Endpoints: O sistema permite que os agentes exponham apenas os endpoints necessários para a interação.

- Modo "Opaco": Os agentes têm a capacidade de operar em modo de "caixa preta". Isto significa que podem disponibilizar os seus serviços como ferramentas, mantendo a sua lógica interna oculta.

2.3.3 Características de Implementação

- Web Nativo: A sua arquitetura foi desenvolvida sobre o padrão web moderno, utilizando HTTP e JSON, além de seguir as normas de segurança.

- Agnóstico ao Modelo: É universal, funcionando independente do LLM escolhido, o modelo deve apenas ser compatível com o protocolo.

- Eficiência Colaborativa: Suporta o processamento de tarefas em tempo real e a colaboração contínua em múltiplos turnos, mantendo a troca de dados leve.

2.3.4 Casos de Uso

- Ecossistemas de Agentes Multiplataforma: Viabiliza a interoperabilidade segura entre agentes desenvolvidos com LLMs diferentes.

- Orquestração Distribuída de Agentes: Facilita a coordenação de agentes nos ambientes em nuvem, integrando-se em ecossistemas como Vertex AI, LangChain e etc.

- Framework de Colaboração Multiagente: Suporta fluxos de trabalho complexos que utilizam de sistemas diferentes, permitindo a cooperação entre agentes de empresas/departamentos diferentes, independente do modelo ou estrutura utilizados.

Comparativo do Principais Protocolos

Característica	MCP	ACP	A2A
Foco Principal	Injeção de contexto para LLMs	Coordenação local de agentes	Comunicação de agentes multiplataforma
Arquitetura	Cliente-servidor (modelo host/servidor)	Descentralizada, runtime local	Cliente/servidor baseado em HTTP com "Agent Cards"
Escopo	Integração vertical (ferramentas → modelo)	Runtime de agente *local-first* (prioridade local)	Integração horizontal (agente ↔ agente)
Mecanismo de Descoberta	Registo de ferramentas no servidor	Transmissão local (*Broadcast*) / registo no runtime	"Agent Card" via HTTP(S)
Protocolo de Transporte	HTTP(S), JSON	IPC, ZeroMQ, gRPC (flexível)	JSON-RPC 2.0 via HTTPS
Modelo de Segurança	Autenticação na camada da App, OAuth2, APIs com escopo	*Sandboxing* de runtime, segurança de rede privada	OAuth2, exposição de *endpoint* com escopo
Ideal para	Apps de LLM com necessidade de dados/ferramentas externas	IA na borda (*Edge AI*), sistemas embarcados, agentes offline	Fluxos de trabalho multiagente entre diferentes plataformas
Exemplo de Caso de Uso	Ligar um LLM a APIs internas	Coordenação no dispositivo de múltiplos pequenos agentes	Agentes corporativos distribuídos a colaborar em conjunto

2.4 Complementares ou concorrentes?

É fácil perceber que o A2A e o MCP complementam-se muito bem, juntos eles fornecem uma base sólida para a construção de sistemas inteligentes e colaborativos. A forma mais simples de perceber isso é lembrar que, resumidamente, MCP conecta o modelo às ferramentas enquanto o A2A conecta um modelo a outro.

Já o ACP adota outra abordagem, focando na coordenação de agentes locais (local-first), no lugar do HTTP ou JSON, o ACP usa protocolos não orientados à conexão com servidores externos, permitindo aos agentes interagirem diretamente um com o outro.

Por isso o ACP não compete diretamente com o A2A, apesar de em certas configurações, o ACP ser um substituto do A2A, uma vez que elimina a sobrecarga dos protocolos nativos web e resolve a tarefa localmente com maior eficiência.

Dificuldades

Sem maiores dificuldades

Conclusões

A análise dos protocolos MCP, ACP e A2A deixa claro que a capacidade da inteligência artificial está além da capacidade individual dos modelos, mas na capacidade de integração e colaboração com outros modelos, aplicando o conceito de aplicações agênticas. Portanto, como os protocolos possuem características distintas, não devem ser vistos como concorrentes, mas como camadas de uma única arquitetura, que permite integração de modelos diferentes nas mais diversas situações e das mais diversas formas.

Referências

[1] CANALTECH.COM.BR. **Google revela ferramentas para criar agentes de IA e automatizar tarefas do dia - Canaltech**

Disponível em: <https://canaltech.com.br/inteligencia-artificial/google-revela-ferramentas-para-criar-agentes-de-ia-e-automatizar-tarefas-do-dia/>

Acesso em 21 de janeiro de 2026

[2] TUTORIALS.BOTSFLOOR.COM. **What Every AI Engineer Should Know About A2A, MCP & ACP | by Edwin Lisowski | Dev Stash**

Disponível em: <https://tutorials.botsfloor.com/what-every-ai-engineer-should-know-about-a2a-mcp-acp-8335a210a742>

Acesso em 21 de janeiro de 2026

[3] DOCS.CLOUD.GOOGLE.COM. **Visão geral do Vertex AI Agent Engine**

Disponível em: <https://docs.cloud.google.com/agent-builder/agent-engine/overview?hl=pt-br>

Acesso em 22 de janeiro de 2026

[4] CLOUD.GOOGLE.COM. **Google Cloud AI Agent Marketplace**

Disponível em: <https://cloud.google.com/blog/topics/partners/google-cloud-ai-agent-marketplace>

Acesso em 22 de janeiro de 2026

[5] CODELABS.DEVELOPERS.COM. **Como começar a usar o protocolo Agent2Agent (A2A): interações de um concierge de compras e um agente de vendas remoto no Cloud Run e no Agent Engine | Google Codelabs**

Disponível em: <https://codelabs.developers.google.com/intro-a2a-purchasing-concierge?hl=pt-br#0>

Acesso em 22 de janeiro de 2026