Google Cloud - Co... · Follow publication

Multi-Agent Madness: Demystifying the Architecture of Salesforce Agentforce + A2A + MCP

4 min read · Jul 10, 2025

Gaurav Kheterpal Follow

In my last post, I briefly wrote about how we can invoke CRUD operations on Salesforce objects via Google Gemini. In this post, I dig deeper into the exploring the architecture of what Agentforce, combined with A2A (Agent-to-Agent) protocols & MCP (Multi-Agent Collaboration Protocol) support looks like.

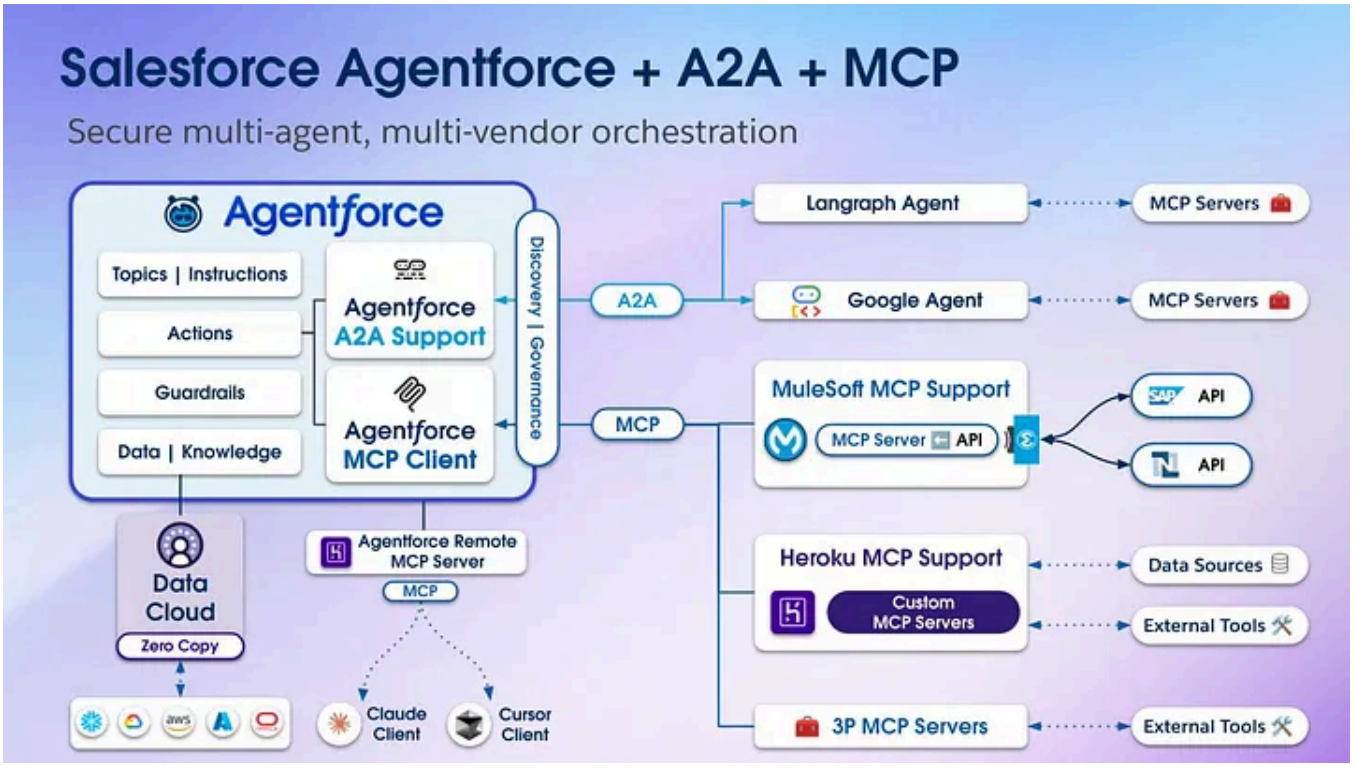

Image from Salesforce

So, first things first, let's start off with some basics.

What Exactly Is Agentforce?

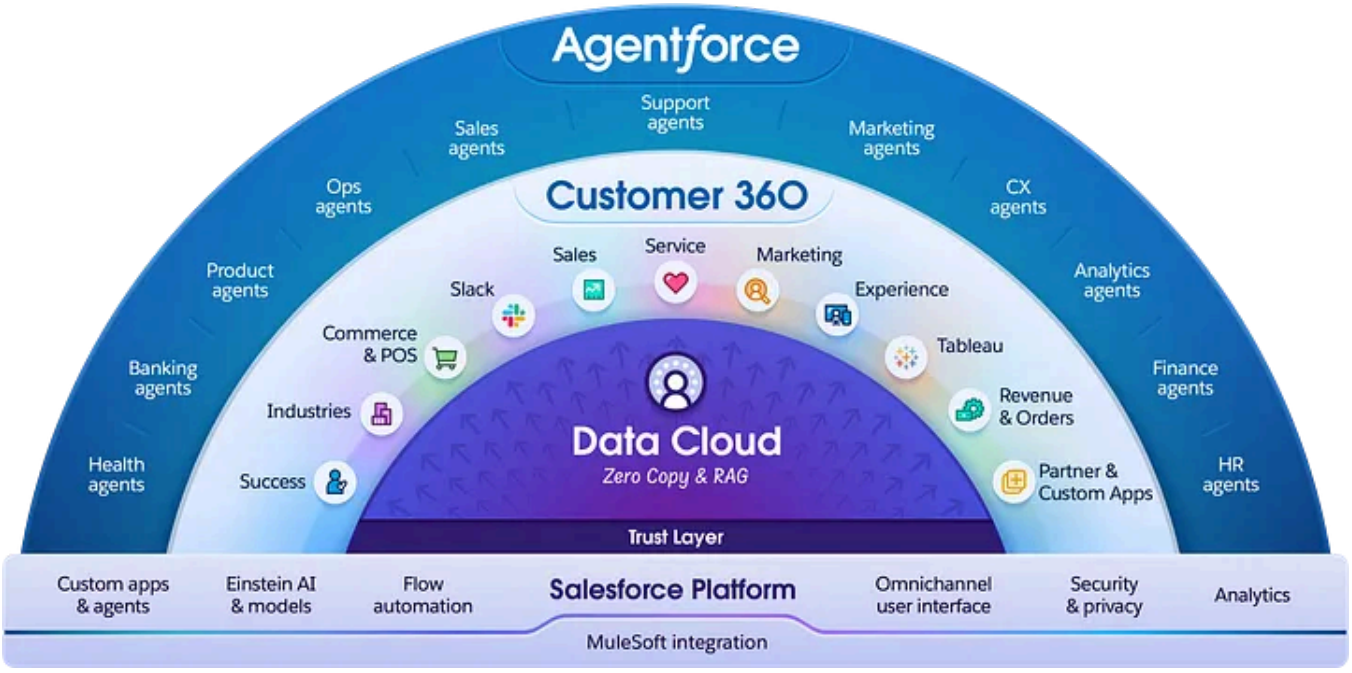

Image from Salesforce

At its core, Agentforce is Salesforce's AI orchestration layer — kind of like the brain that coordinates:

- What tasks agents should do (topics & instructions)
- How they do them (actions & guardrails)
- And what they're working with (data & knowledge)
- The true power of Agentforce is that it's deeply connected with Salesforce Data Cloud — meaning we're not copying or duplicating data. It's all **zero-copy.** Data stays where it is, and Agentforce sends the intelligence to it.

A2A — Standardising Agent to Agent Communication

So, how does Agentforce talk to other Agents — for example, a ServiceNow support agent or a SAP inventory agent? Enter A2A — I can now have Agentforce talk directly to a Google Agent, a Langraph Agent, or any other LLM-powered tool using the A2A protocol.

Image from Google

MCP — Deeper, Smarter Integrations

Now, when I need to connect to enterprise systems — like SAP, NetSuite, or internal APIs — that's where MCP (Multi-Agent Collaboration Protocol) comes in.

Agentforce acts as an MCP client, and connects to:

- MuleSoft MCP Servers — to talk to APIs via standard integration patterns
- Heroku MCP Servers — for custom logic, external tools, or data sources
- Even 3rd-party MCP Servers — giving us full flexibility across the ecosystem

This lets me build secure agent workflows that talk to literally anything: legacy systems, new APIs, or even AI-driven endpoints.

Image from Anthropic

How does it all come together?

What truly excites me about this architecture is its openness. It's not a walled garden. We designed it to be an ecosystem.

- Third-Party Agents: The ability to plug in agents like Langgraph Agent or Google Agent is a game-changer. I don't have to build everything myself. I can bring in best-in-class agents for specialized tasks and have them collaborate seamlessly with my own.
- Integration with MuleSoft: This is where we bridge the gap between AI and our core business systems. Through MuleSoft MCP Support, I can empower an agent to interact with an SAP API or other legacy systems. This turns AI from an analytical tool into a "doer" that can execute real business processes.

Image from MuleSoft

- Customization with Heroku: For unique challenges, I need flexibility. Heroku MCP Support gives my team the power to build and deploy Custom MCP Servers. This lets us connect to any number of proprietary Data Sources or External Tools, making the possibilities virtually limitless.

Image from Heroku

Governance

With such a hetrogenous architecture, Governance is key here or else it will truly be multi-agent madness :) — That's why I'm glad Salesforce built **Governance & Discovery** into the fabric of this architecture:

- I can define what an agent can see, access, or do
- Set guardrails so nothing goes rogue
- And track every step of the agent orchestration lifecycle

Challenges

While on paper this looks powerful and I'm still wrapping my head around getting a working proof of concept for this, there's definitely a lot of challenges to be considered.

While powerful, this architecture introduces significant challenges that require careful consideration:

- **Technical Complexity:** Managing and debugging a process that hops between multiple agents, platforms (like MuleSoft and Heroku), and vendors is incredibly complex. Identifying the root cause of failures or performance bottlenecks in this distributed system is a major hurdle.
- **Security and Governance Risks:** The open, multi-vendor nature expands the system's attack surface. Enforcing your internal safety "Guardrails" on third-party agents is difficult, and securely managing data and permissions as they are passed between different vendors creates significant security and privacy challenges.
- **Ecosystem Adoption:** The success of the A2A and MCP protocols depends on widespread adoption by all agent vendors. Without a critical mass of participants adhering to this standard, the promised interoperability will not materialize.
- **Operational Overhead and Cost:** Managing the costs, contracts, updates, and potential breaking changes from numerous different vendors creates a high operational burden. A simple update to one agent could require re-testing the entire workflow, increasing maintenance complexity and costs.

Despite all the challenges and the fact that it's still early days in the world of Multi-Agent orchestrations, I believe this will well and truly be the future of Agentic AI.

More on this from yours truly in the next post!

Signing off,

Agent Gaurav :-}

Follow

Published in Google Cloud - Community

69K followers · Last published 5 hours ago

A collection of technical articles and blogs published or curated by Google Cloud Developer Advocates. The views expressed are those of the authors and don't necessarily reflect those of Google.

Follow

Written by Gaurav Kheterpal

532 followers · 932 following

Salesforce MVP Hall of Fame | Google Developer Expert - AI | Founder & CEO - Vanshiv Technologies

Responses (2)

Bgerby

What are your thoughts?

Abhishek Simgekar
Jul 10

Great read. 👏🏻
Thank you for sharing.

1 Reply

Suchit Tripathy
Sep 7

very informative article Gaurav. Thanks for sharing this. Please let us know once you have a workable POC, using A2A and MCP with AgentForce.

Thanks

Suchit

Reply

More from Gaurav Kheterpal and Google Cloud - Community

In Google Cloud - Community by Gaurav Kheterpal

My AI Agent Experiments — Connecting Salesforce Agentforce with Google Cloud's ADK

Before anything else, a quick disclaimer — I do a lot of work on Salesforce so my opinions on Agentforce may be biased.

Jun 30 2 1

In Google Cloud - Community by Romin Irani

Gemini CLI Tutorial Series

Welcome to the Gemini CLI Tutorial Series. It is an open-source AI agent that brings the power of Gemini directly into your terminal. You...

Jun 27 768 10

In Google Cloud - Community by Arjun Prabhulal

Model Context Protocol(MCP) with Google Gemini LLM — A Deep Dive (Full Code)

A step-by-step guide with code, architecture, and real-world use case

Apr 4 1.3K 22

Gaurav Kheterpal

My AI Agent Experiments — Connecting Salesforce Agentforce with Google Cloud'sADK

Before anything else, a quick disclaimer — I do a lot of work on Salesforce so my opinions on Agentforce may be biased.

Jun 30 3

See all from Gaurav Kheterpal

See all from Google Cloud - Community

Recommended from Medium

Hangsik Shin

Getting Started with Google ADK: A Step-by-Step Local Setup and Testing Guide

Ready to start building powerful AI agents with Google's Agent Development Kit (ADK)? This step-by-step guide walks you through the entire...

Aug 15 72

Velu Sankaran

The Agent Architecture Blueprint: LangGraph + A2A Server Integration

Bridging Stateful Agent Logic with Distributed Communication Protocols

Jun 5 2 1

Heemeng Foo

Playing around with A2A — LangGraph & CrewAI

And using chocolate cake to explain how this works

Sep 2 15

In Another Integration Blog by Sravan Nerella

MuleSoft MCP: Building an AI Agent with Anypoint Platform APIs

Build an MCP Server to Interface with Anypoint Design Center via Natural Language or plain english. Sounds exciting right?

May 29 6

Anton Kutishevsky

How to build a simple Salesforce MCP server with Node.js, a step-by-step guide.

This article will guide you through building a simple yet functional MCP server that integrates with Salesforce, enabling Claude Desktop to...

Jun 16 12 1

Todd Greco

How to Give GitHub Copilot a Photographic Memory (and a Kiro-Style Brain)

Give Copilot a persistent memory and a goal-oriented workflow by combining kiro and a memory bank.

Jul 19 42 1

See more recommendations