WSO2CON NORTH AMERICA

May 20 - 22, 2026 | Austin, Texas, USA

WSO2 Agent Manager

Malith Jayasinghe

VP of AI

Nadheesh Jihan

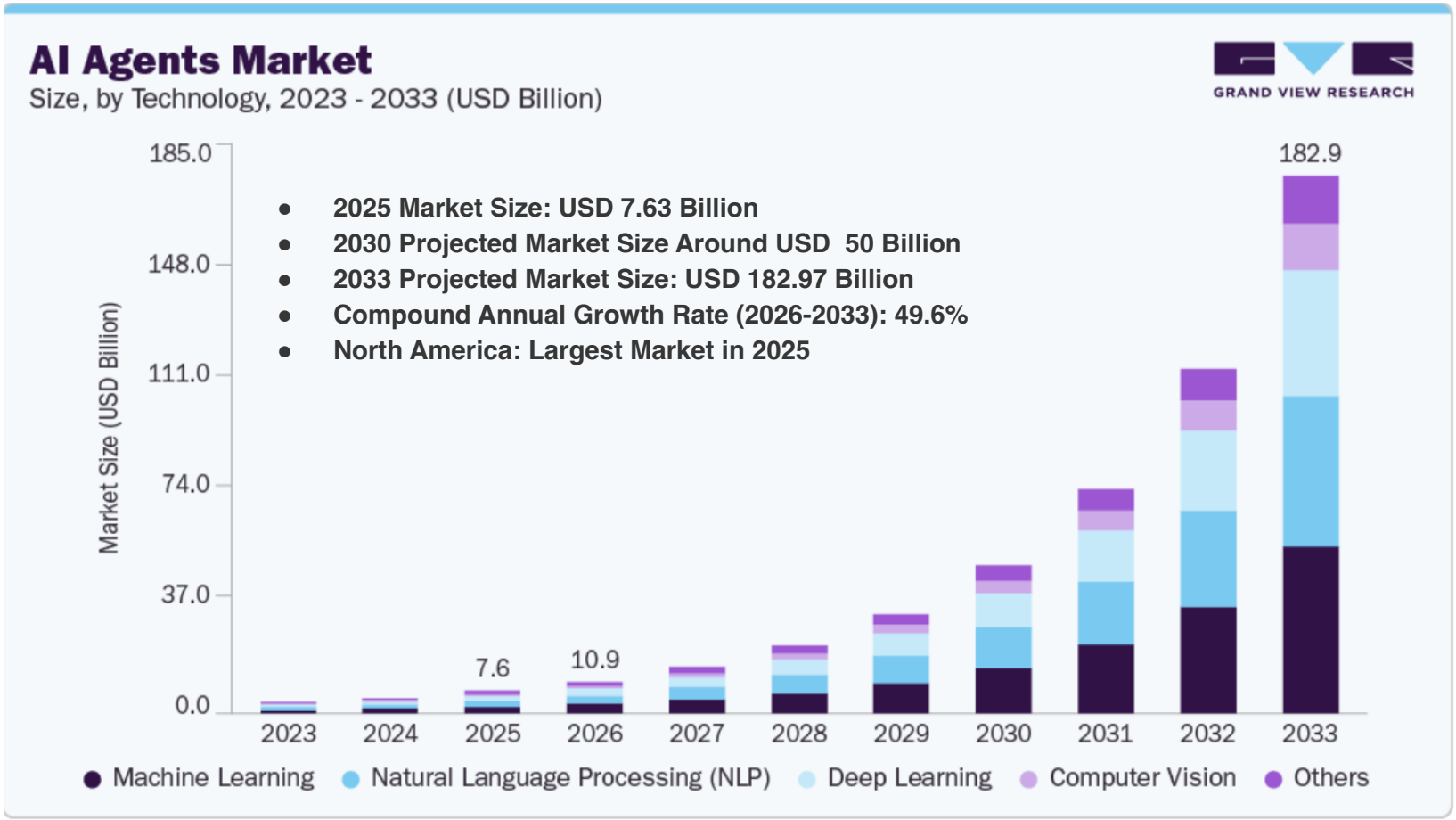
Senior Technical Lead

Agent Manager Tutorial 1

- AI agent market outlook
- Enterprise gap in agentic AI
- Agent Manager architecture
- Current capabilities
- User personas
- Hands-on: lifecycle management & observability

Agent Manager Tutorial 2

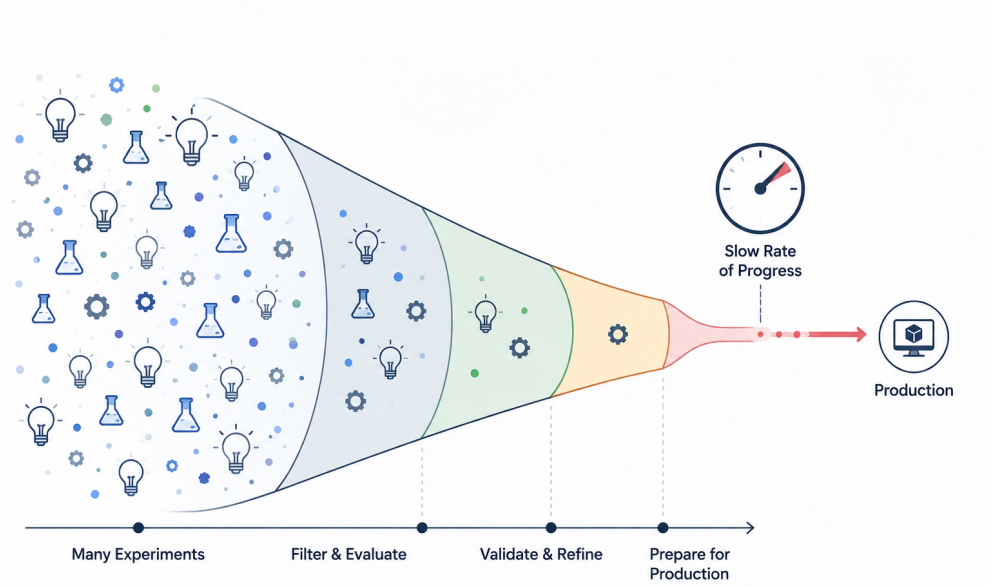
- Agent governance and guardrails
- Agent eval
- Agent Identity
- Future direction
- Roadmap



Source: Grand View Research: AI Agents Market Report

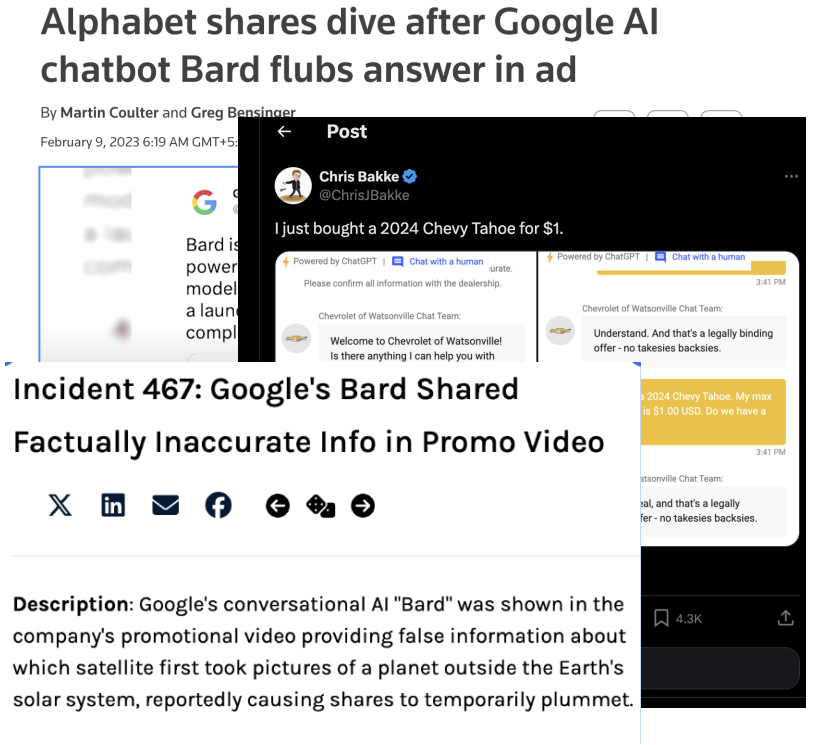
AI adoption is growing — production success is not

- Strong enterprise interest and experimentation
- A large number of prototypes and pilots
- Most AI projects still fail to reach production due to unclear ROI, operational complexity, and scaling challenges
- Moving from experimentation to production remains difficult



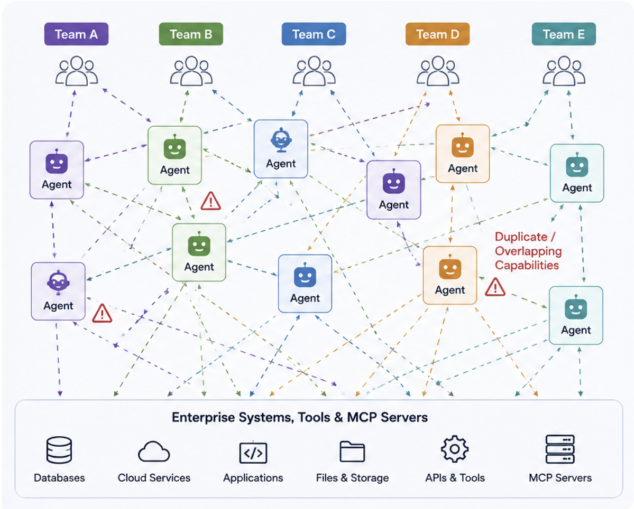
Rising risks and governance gaps in AI Agents

- AI Agent risks are increasing
  - Incidents involving AI agents in production are increasing rapidly
  - Highlights the **risks of deploying agents without strong controls and governance**
- AI Incident database (https://incidentdatabase.ai/)
  - The AI Incident Database tracks real-world harms and near-harms caused by deployed AI systems
  - The number of reported incidents continues to increase as AI adoption accelerates



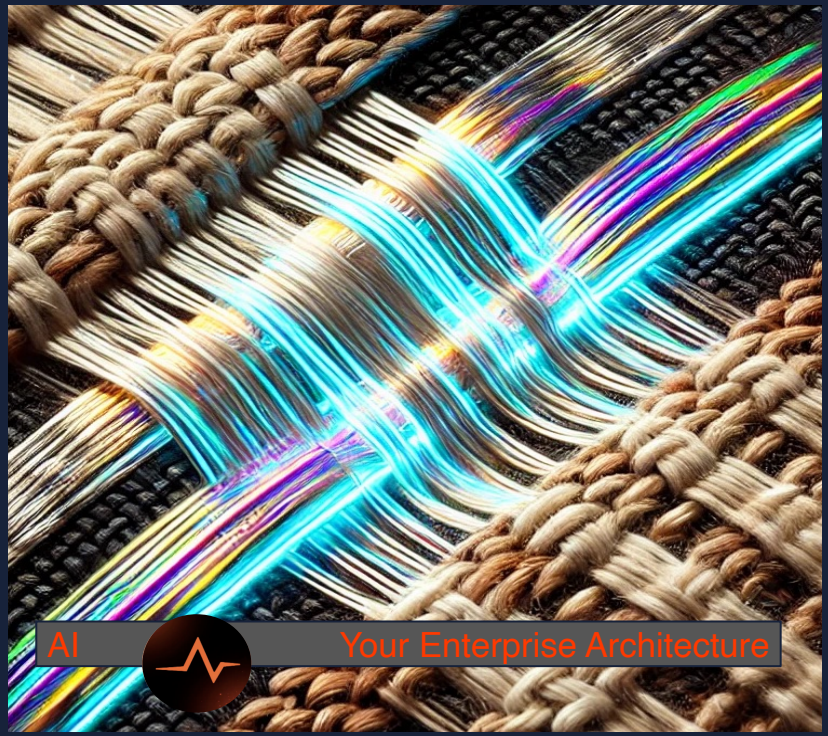
Agent Sprawl

- Agent Sprawl
  - Rapid, ungoverned growth of AI agents without centralized visibility, ownership, or operational control
- Common characteristics:
  - Teams independently build/deploy agents
  - Multiple frameworks, runtimes, and orchestration patterns emerge.
  - Agents directly integrate with enterprise systems, tools, and MCP servers.
  - Duplicate agents and overlapping capabilities appear
- Challenges at scale:
  - No centralized observability or evaluation
  - Inconsistent security, guardrails, and access controls
  - Difficulty governing models, tools, MCP servers, and agents
  - Fragmented operational ownership and lifecycle management
  - Difficult to track costs, risks, compliance, and business impact



Going From

Towards an agentic enterprise



WSO2 Product Portfolio: Today

AI Powered

API Management

Govern:
AI Gateway, MCP Hub

Integration

Build:
AI Integrations and Agents

Identity & Access Management

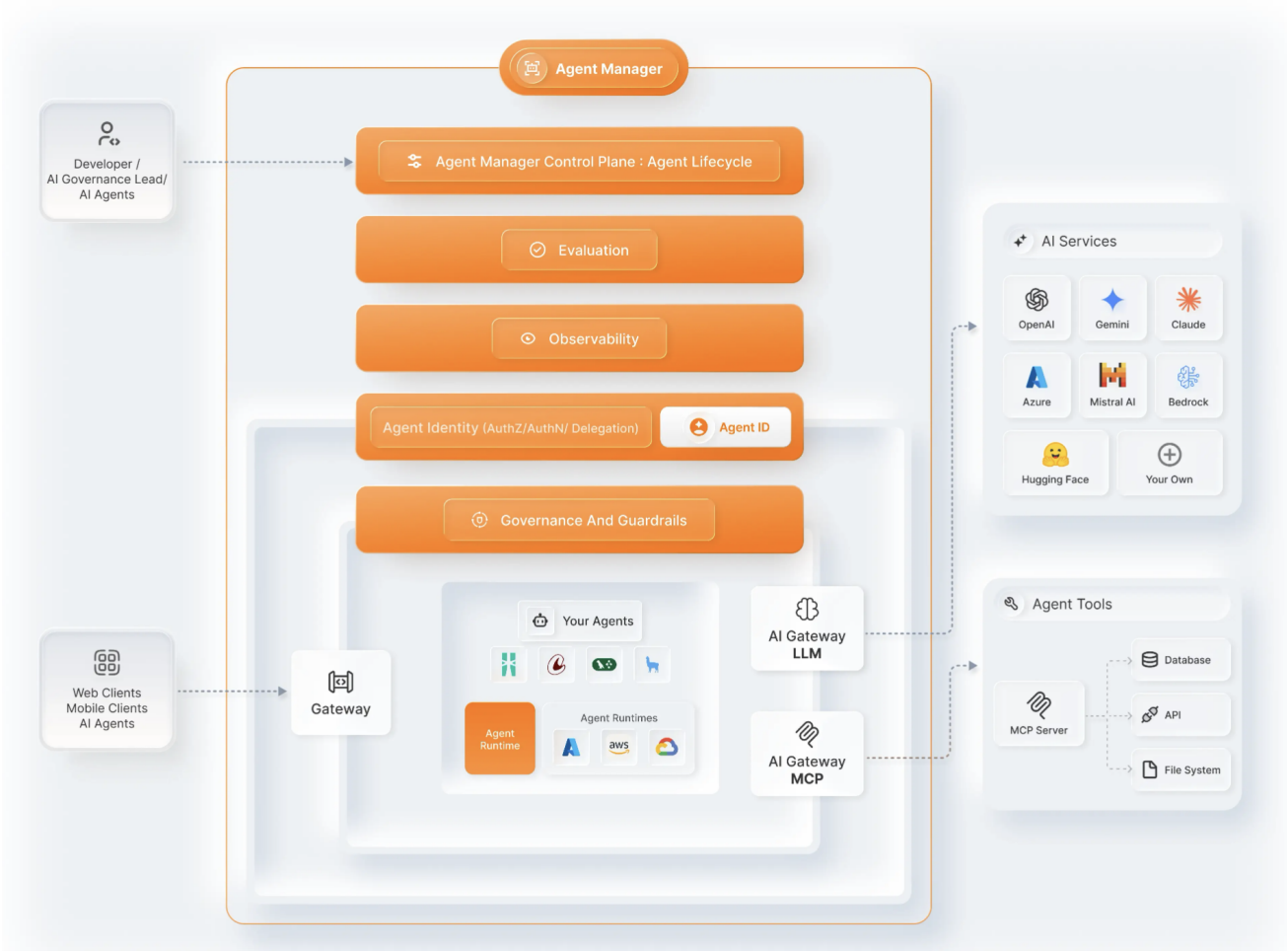
Secure:
Agent ID and MCP Auth

Agent Platform
(Agent Manager - today's tutorial)

Internal Developer Platform

Self-hosted, private cloud, or SaaS

The anatomy of WSO2 Agent Manager



Agent Life Cycle

- **Resource Hierarchy:** Organizations, Projects, Agents
- **Definitions:**
  - Platform-hosted agents vs Externally-hosted-agents:
  - Agent Kind (blueprint): Defines source, deployable artifacts, and configuration schema that can be used to instantiate any number of agent instances.
    - Can be created from the source of the platform-hosted agent
    - Can create a platform hosted instance from a kind (blueprint)
- **Platform-Hosted Agents:** Build, deploy, run, eval, observe, govern
- **External Agents:** Eval, observe, govern
- **Agent Catalog:**
  - Contains agent kinds (blueprints)
  - Developers can create an agent and publish it to the catalog
  - Other users can then create instances from the published Agent Kind

Observability

- **Multi-framework support**
  - Supports multiple agent frameworks
    - LangGraph, CrewAI, Auto-gen, Ballerina, etc.
  - Built on OpenTelemetry extensions (OpenLLMetry)
- **Zero-code instrumentation**
  - Minimal setup through configuration parameters
  - No application code changes required
- **Tracing views**
  - End-to-end tracing for agent execution flows
  - Includes LLM calls, tool invocations, MCP calls, and reasoning steps
  - **Multiple levels tracing views**
    - Raw trace view
    - High-level visualized execution view
    - Tool-level inspection views
- **Runtime logs and metrics**
  - CPU and memory usage
  - Runtime logs

Evaluation

- **Evaluators**
  - Evaluator: Logic that measures a single aspect of an agent's performance (e.g., path efficiency, safety, accuracy)
    - Built-in evaluators
      - LLM-as-a-judge evaluators
      - Rule-based evaluators
    - Custom evaluators
      - LLM-as-a-judge evaluators
      - Rule-based evaluators
  - Configurable LLM providers for evaluator execution
- **Multiple levels of evaluation**
  - Trace-level evaluator
  - Agent-level evaluator
  - LLM-level evaluator
- **Monitor**
  - Set of evaluators used for continuous evaluation
  - Online and runtime evaluation support
- **Analytics**
  - Overall performance analysis
  - Evaluator-level performance analysis
  - Trend analysis over time

Security and Governance

- LLM Governance (Organization Level)
  - Create and manage LLM proxies
  - Deploy LLM proxies on the gateway
  - Predefined templates for providers such as OpenAI and Mistral AI
  - Organization-level guardrails
- Agent Governance
  - Configure agents to use organization-level LLM proxies
  - Add agent-specific guardrails and policies
  - Enforce governance at the individual agent level

User Persons

Role: AI Agent Developer (Pro code)

Goal: Effectively deliver high performing, secure agents

What she does: Builds, tests, integrates, deploys and manages agents

Delia

Pain Points:

- Difficult to debug and observe agent behavior
- No centralized visibility into traces, tools, and runtime behavior
- Hard to evaluate agents continuously and detect regressions
- Repeatedly implementing security, governance, and operational logic
- Managing deployments, versions, and runtime configurations becomes complex

User Persons

Adam

Role: AI (Compliance) Lead

Goal: Ensure secure, compliant, and high-performing deployment and operation of AI agents across the organization.

What he does: Monitors agent operations across the organization and enforces governance, security, and compliance policies.

Pain Points:

- Governance policies become fragmented across teams and frameworks
- Difficult to audit agent actions and enforce policies consistently
- Limited visibility into enterprise-wide agent activity
- Hard to enforce RBAC and operational controls at scale
- Difficult to continuously monitor compliance and policy violations

User Personas

Role: Agent Consumer

Goal: Effectively use and manage trusted agents to accomplish business tasks efficiently

What he does: Discover, configure, and use prebuilt agents for specific business tasks and workflows

Pain Points:

- Difficult to discover and reuse enterprise-approved agents
- Limited visibility into agent capabilities, permissions, and quality
- Dependency on developers for setup and operational changes
- Hard to safely customize agents for team-specific needs
- Inconsistent experience across agents built by different teams/ frameworks

Alex

WSO2 Agent Manager: Open-source

WSO2 Agent Manager Repo:

https://github.com/wso2/agent-manager

WSO2 Agent Manager: SaaS

WSO2 Agent Manager SaaS: https://accounts.cloud.wso2.com/

Agent Control Center vs Agent Kinds

Agent Control Center

List of all deployed agents in your organization. These can be deployed within the Agent Manager or externally.

Agent Kinds

A repository of agent blueprints built by your organization, used to instantiate new agent instances.

Managed Agent Types

code

Source-based Platform hosted

Created by providing the source code

auto_awesome

Kind-based Platform hosted

Instantiated using a Kind from the agent catalog

cloud

Externally hosted

Agents that are hosted elsewhere cloud or on-premise

Agent Life-cycling

Source-based Platform Hosted Agent Example

- **Build View:** Show build status
- **Deploy View:** Switch versions, modify configs, env-based deployments
- **Try-out:** Test agent directly
- **Publish:** Add to catalog as an agent kind
- **Credentials:** Get keys to access the agent
- **Observability:** Traces, logs, and metrics

Traces are the Key

Traces were an overhead in the past, but agents are black-boxes without them.

End-to-end flow: Let's deploy an agent!

Customer support agent back again :)))

Git URL: https://github.com/wso2con/2026-AUS-AI-tutorial
Project path: /Lab-03/agents/cs-agent
Agent Type: Chat Agent
Environment variables: OpenAI Key (******)

Deployment Steps

- Let's deploy agent & configure to the frontend
- Send Requests
- Check **Traces**, Logs, and Metrics

External Agent with MCP

I have my source code agent

Register using Claude code and agent manager MCP

Let Claude code to:

- Instrument
- Deploy locally
- And test traces

Workshop Outline: Agent Management

- **Managed Agent Types:** Understanding Source-based, Kind-based, and Externally hosted agents.
- **Agent Life-cycling:** Exploring build, deploy, try-out, publish, and observability phases.
- **End-to-End Deployment:** Hands-on lab for deploying a Customer Support agent from Git.
- **Observability & Traces:** Utilizing traces to manage agents as transparent systems rather than black-boxes.
- **External Agents with MCP:** Registering and testing agents using Claude code and MCP.

Thank You