AGNTCY

AGNTCY Integration Capabilities: An Overview

by Ramiz Polic, Luca Muscariello

Goals

Support Agentic Standards

Support widely-adopted standards for agentic development like MCP servers

Enhance Developer Experience

Support AI-assisted developer workflows via tools and IDEs like Visual Studio Code

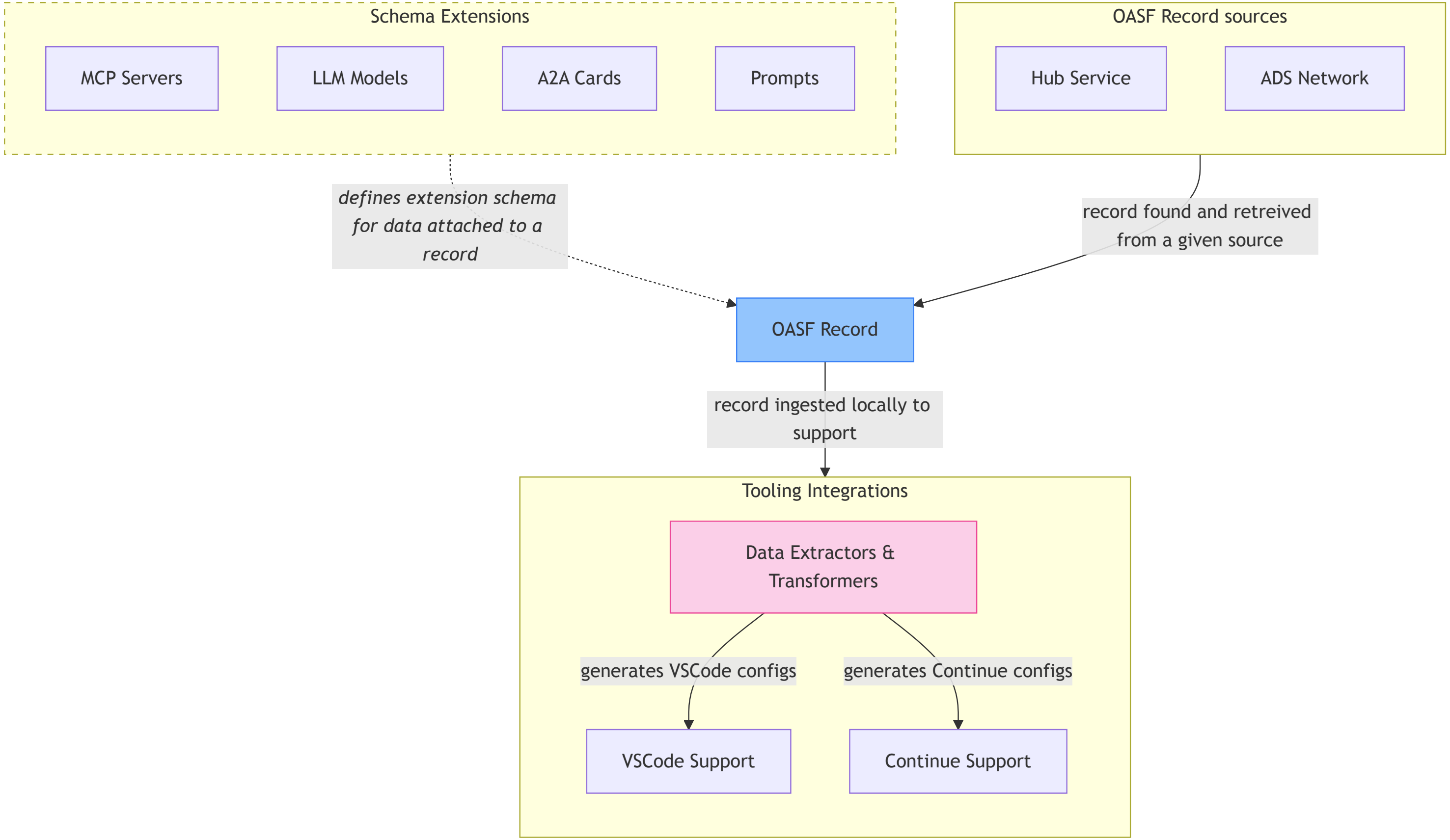
Simplified Integration

Easy integration and usage of AGNTCY components like OASF and ADS

Methodology

1. **Schema Extensions**
   - Use OASF records as data carriers for common tools
   - Add native support for **LLMs, Prompts, A2A Cards, MCP servers**
2. **Data Extractors and Transformers**
   - Generate tool-specific configuration files from OASF records
   - Support tool usage via **VS Code Copilot** and **Continue.Dev**
3. **Workflow Integration**
   - Provide simple setup instructions
   - Enable immediate productivity

Architecture



Schema Extensions

MCP Servers

LLM Models

A2A Cards

Prompts

OASF Record sources

Hub Service

ADS Network

defines extension schema for data attached to a record

record found and retreived from a given source

OASF Record

record ingested locally to support

Tooling Integrations

Data Extractors & Transformers

generates VSCode configs

generates Continue configs

VSCode Support

Continue Support

OASF Schema Extensions

OASF Schema: LLM Extension

```
{
  "extensions": [{
    "name": "schema.oasf.agntcy.org/features/runtime/model",
    "version": "v1.0.0",
    "data": {
      "models": [
        {
          "model": "deepseek-r1:1.5b",
          "provider": "ollama",
          "api_base": "http://localhost:11434",
          "prompt": "You are an expert software developer…"
        },
        {
          "model": "gpt-4o",
          "provider": "azure",
          "api_base": "${input:AZURE_OPENAI_API_BASE}",
          "api_key": "${input:AZURE_OPENAI_API_KEY}",
        }
      ]
    }
  }]
}
```

Features

Describes LLM support and its configuration for a given agent.

- Multi-model support
- Local and cloud providers
- Model tuning parameters
- Compatible with existing tools

OASF Schema: LLM Prompt Extensions

```
{
  "extensions": [{
    "name": "schema.oasf.agntcy.org/features/runtime/prompt",
    "version": "v1.0.0",
    "data": {
      "prompts": [
        {
          "name": "PR Summary",
          "description": "PR analysis",
          "prompt": "Summarize the pull request in detail..."
        },
        {
          "name": "PR Review",
          "description": "PR review",
          "prompt": "Review the pull request in detail..."
        }
      ]
    }
  }]
}
```

Features

Describes common LLM interaction prompts to use the agent.

- Structured LLM Prompts
- Categorized by purpose
- Descriptive metadata
- Task-specific instructions
- Compatible with existing tools

OASF Schema: MCP Server Extension

```
{
  "extensions": [{
    "name": "schema.oasf.agntcy.org/features/runtime/mcp",
    "version": "v1.0.0",
    "data": {
      "servers": {
        "github": {
          "command": "docker",
          "args": [
            "run", "-i", "--rm",
            "-e", "GITHUB_PAT",
            "ghcr.io/github/github-mcp-server"
          ],
          "env": {
            "GITHUB_PAT": "${input:GITHUB_PAT}"
          }
        }
      }
    }
  }]
}
```

Features

Describes MCP servers required to run and interact with the agent.

- Support for multiple servers
- Input variable mapping and templating
- Compatible with existing tools

OASF Schema: A2A Extensions

```json
{
  "extensions": [{
    "name": "schema.oasf.agntcy.org/features/runtime/a2a",
    "version": "v1.0.0",
    "data": {
      "name": "example-agent",
      "description": "An agent that performs web searches",
      "url": "http://localhost:8000",
      "capabilities": {
        "streaming": true,
        "pushNotifications": false
      },
      "defaultInputModes": ["text"],
      "defaultOutputModes": ["text"],
      "skills": [
        { "id": "browser", "name": "browser automation" }
      ]
    }
  }]
}
```

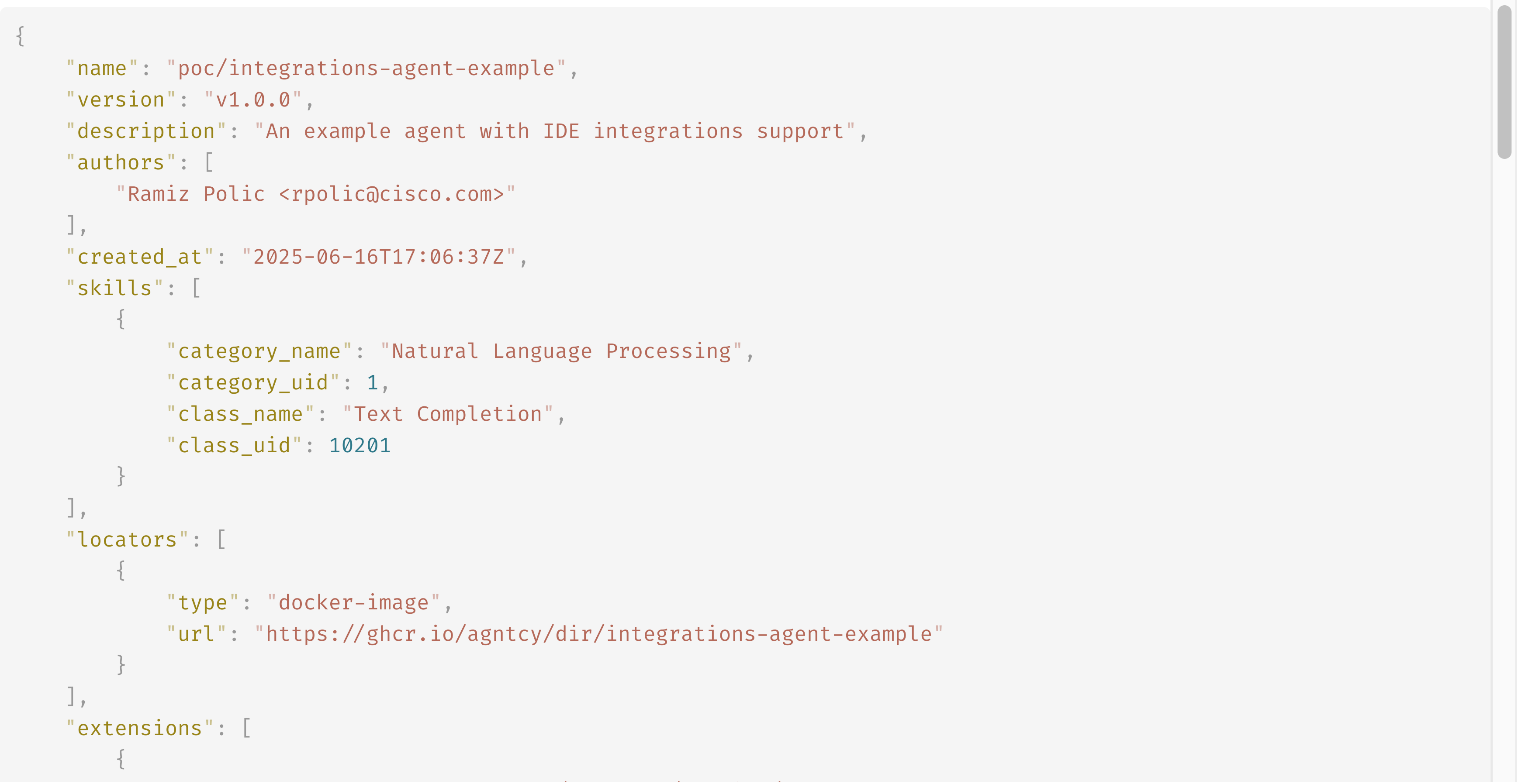
Features

Describes A2A card details for communication and its usage with A2A protocol.

- Common A2A Card schema
- Compatible with existing tools

Agentic Workflow: IDE Integrations

Agentic Workflow: Example OASF record



```
{
    "name": "poc/integrations-agent-example",
    "version": "v1.0.0",
    "description": "An example agent with IDE integrations support",
    "authors": [
        "Ramiz Polic <rpolic@cisco.com>"
    ],
    "created_at": "2025-06-16T17:06:37Z",
    "skills": [
        {
            "category_name": "Natural Language Processing",
            "category_uid": 1,
            "class_name": "Text Completion",
            "class_uid": 10201
        }
    ],
    "locators": [
        {
            "type": "docker-image",
            "url": "https://ghcr.io/agntcy/dir/integrations-agent-example"
        }
    ],
    "extensions": [
        {
```

VS Code Integration

Explores ways to use OASF records to enable its usage in agentic workflows with VS Code.

Implementation

1. Load and Process OASF records
2. Extract data from OASF record about:
   - MCP servers
   - NOTE: other components are not yet supported
3. Generate `.vscode/mcp.json` configuration file

Features

- **MCP Server Configuration**
  - Automatic server setup
  - Secure credential handling
  - Input variable templating
- **Copilot Agent Mode Support**
  - Native integration
  - Interactive credential prompts

VS Code Integration: Live Demo

Continue Integration

Explores ways to use OASF records to enable its usage in agentic workflows with VS Code Continue.dev extension.

Implementation

1. Load and Process OASF records
2. Extract data from OASF record about:
    - MCP servers
    - LLM Data
    - Prompts
    - A2A information
3. Generate `.continue/assistants/` configuration file

Features

- **Quick Provisioning**
    - Multi-model configuration
    - Pre-defined prompts
    - MCP server integration
- **Dev Experience**
    - Assistant gallery integration
    - Model provider selection
    - Prompt usage

Continue Integration: Live Demo

MCP to OASF Example: Agentic App

Demonstrate the usage of OASF agents by loading them into IDEs and performing an agentic workflow to scan MCP server records from GitHub and create matching OASF records.

VSCode Workflow

1. Open VSCode chat console
2. Switch to LLM (e.g., Claude)
3. Enable Agent mode
4. Enter conversion generation prompt
5. Check results

Continue Workflow

1. Open Continue chat console
2. Refresh Assistants tab
3. Select OASF-generated assistant
4. Switch to configured LLM (Azure GPT-4o)
5. Select conversion generation prompt
6. Check results

MCP to OASF Example: OASF Record

Connecting everything together to showcase an example application.
Built on top of OASF and ADS with native IDE support.

```json
{
    "name": "poc/mcp-scanner",
    "version": "v1.0.0",
    "description": "An agent that scans MCP server registries and generates OASF records for each MCP server.",
    "authors": [
        "Ramiz Polic <rpolic@cisco.com>"
    ],
    "created_at": "2025-06-16T17:06:37Z",
    "skills": [
        {
            "category_name": "Natural Language Processing",
            "category_uid": 1,
            "class_name": "Text Completion",
            "class_uid": 10201
        }
    ],
    "locators": [
        {
            "type": "docker-image",
            "url": "https://ghcr.io/agntcy/dir/mcp-scanner"
        }
```

MCP to OASF Example: Live Demo

Outcomes

Extensible Schema: OASF can easily support third-party integrations via extensions

Minimal Impact: No significant changes needed to ADS or OASF projects

Simple Integration: Straightforward integration of AGNTCY components

Developer Experience: Leverages familiar tools with minimal configuration

Transformation Pattern: Data extraction and transformation approach works well

Thank You

github.com/agntcy/dir