Introducing gofannon

PIng, Portland OR, April 30 2025

About Me

Andrew Musselman works on data and analytics, and runs software teams for a living, most recently at Speedchain. He has contributed to open source projects for over a decade, mostly focusing on machine learning and AI, is a member of the Apache Software Foundation and the AI Alliance, and lives in Portland with two kids and two dogs.

In his free time he plays music and has a project car in the garage.

Contact info: akm@apache.org,
https://www.linkedin.com/in/andrewmusselman

Agenda

- How LLMs, Tools, and Agents Work
- Gofannon Overview
- Cross Framework Interoperability
- Contributor Friendly
- Conclusion

How LLMs, Tools, and Agents Work

A Primer

Function/Tool Example

- An AI assistant that can analyze vehicle safety data.
- The National Highway Traffic Safety Administration (NHTSA) provides a REST API to access this data by vehicle make, model, and year.
- “Please give me an issue summary of the 2024 Tesla Cybertruck”
- If the LLM knows about NHTSA’s HTTP REST endpoint (/SafetyRatings/modelyear/2024/make/Tesla/model/Cybertruck) and can fill in the parameters, then it can retrieve the data and analyze it.
- ”Functions” are a format to bridge the gap between APIs and LLMs, using the function to enrich the prompt with the needed data.
- Many LLMs and LLM APIs support functions
- Foundation for more complex agent LLM applications

Function/Tool Example

- **<u>This is not using a search engine to search data previously staged, cleaned, and maintained.</u>**
- This is pulling the query together live in real time

Working Example NHSTA Complaints

“Can you please tell me about complaints dealing with the 2024 Cybertruck?”

Step 1 – Define the function (REST call):

```
def get_nhsta_complaints_by_vehicle(make: str, model: str, modelYear: str):
  """https://api.nhtsa.gov/complaints/complaintsByVehicle?make=acura&model=rdx&modelYear=2012"""
  base_url = "https://api.nhtsa.gov/complaints/complaintsByVehicle"
  payload = {
      "make": make,
      "model": model,
      "modelYear": modelYear
  }
  r = get(base_url, params=payload)
  return json.dumps(r.json())
```

Working Example NHSTA Complaints

Step 2 - Wrap the function so the LLM can use it:

```
# here is the definition of our function
tools = [
    {
      "type": "function",
      "function": {
          "name": "get_nhsta_complaints_by_vehicle",
          "description": "Get NHSTA Complaints by vehicle and model year",
          "parameters": {
              "type": "object",
              "properties": {
                  "make": {
                      "type": "string",
                      "description":
                          "The make of the vehicle, e.g. Acura"
                  },
                  "model": {
                      "type": "string",
                      "description": "The model of the vehicle, e.g. ILX"
                  },
                  "modelYear": {
                      "type": "string",
                      "description": "The model year of the vehicle, e.g. 2022",
                  }
              },
              "required": ["make", "model", "modelYear"]
          },
      }
    }
]
```

Working Example NHSTA Complaints

Step 3 – First call to the LLM: prompt gets successfully parsed into the function:

```python
# here is the user request
messages = [
    {
        "role": "user",
        "content": "What's up with the 2024 Tesla Cybertruck?"
    }
]

# let's send the request and print the response
response = openai.chat.completions.create(
    model="meta-llama/Meta-Llama-3.1-70B-Instruct",
    messages=messages,
    tools=tools,
    tool_choice="auto",
)
tool_calls = response.choices[0].message.tool_calls
for tool_call in tool_calls:
    print(json.dumps(tool_call.model_dump(), indent=3))
```

```
{
   "id": "call_xI7WB2OF6J6I9R0w6Ex9eimA",
   "function": {
      "arguments": "{\"make\": \"Tesla\", \"model\": \"Cybertruck\", \"modelYear\": \"2024\"}",
      "name": "get_nhsta_complaints_by_vehicle"
   },
   "type": "function"
}
```

Working Example NHSTA Complaints

Step 4 –Set up a second call to the LLM. Execute the function locally and add the response to the prompt of the second call.

```
for tool_call in tool_calls:
  function_name = tool_call.function.name
  if function_name == "get_nhtsa_complaints_by_vehicle":  #"get_current_weather":
      function_args = json.loads(tool_call.function.arguments)
      function_response = get_nhtsa_complaints_by_vehicle(
          make=function_args.get("make"),
          model=function_args.get("model"),
          modelYear=function_args.get("modelYear"),
      )

  # extend conversation with function response
  messages.append({
      "role": "tool",
      "content": function_response,
  })
```

Working Example [NHSTA Complaints]

The prompt now consists of the following components:

```
{
    "role": "user",
    "content": "What's up with the 2024 Tesla Cybertruck?"
}
```

```
{
    "id": "call_toSFenKR0zMl4nGBkNzqnxVX",
    "function": {
      "arguments": "{\"make\": \"Tesla\", \"model\": \"Cybertruck\", \"modelYear\": \"2024\"}",
      "name": "get_nhtsa_complaints_by_vehicle"
    },
    "type": "function"
}
```

```
{
    "tool_call_id": "call_toSFenKR0zMl4nGBkNzqnxVX",
    "role": "tool",
    "content": function_response
}
```

Working Example NHSTA Complaints

Step 5 –Second call to the LLM, execute the function and evaluate the response:

```
# get a new response from the model where it can see the function responses
second_response = openai.chat.completions.create(
  model="meta-llama/Meta-Llama-3.1-70B-Instruct",
  messages=messages
)

print(second_response.choices[0].message.content)
```

```
Based on the NHTSA complaints, there are several issues with the 2024 Tesla Cybertruck, including:

1. Tire blowouts: There have been reports of tire blowouts, particularly with the Pirelli Scorpion ATR tires.
2. Electrical system issues: Some owners have reported electrical system failures, including critical error messages and loss of system redundancy.
3. Steering issues: There have been reports of steering problems, including power reduced and critical steering issue detected messages.
4. Suspension issues: Some owners have reported issues with the air suspension system, including inability to adjust and error messages.
5. Structure issues: There have been reports of structural issues, including misaligned doors and loose trim pieces.
6. Visibility issues: Some owners have reported visibility issues, including windshield wiper failures and snow accumulation on the headlights.
7. Fire risks: There have been reports of fire risks, including a complaint about the large driver's touchscreen getting extremely hot.
8. Latch issues: Some owners have reported issues with the door latches, including a complaint about the passenger side door swinging open while driving.
9. Propulsion system issues: There have been reports of propulsion system issues, including a complaint about the accelerator pedal slipping and lodging
```

In summary we have sent two messages to our LLM, the question and where to find more info

Working Example NHSTA Complaints

Step 5 –Second call to the LLM, execute the function and evaluate the response:

```
# get a new response from the model where it can see the function responses
second_response = openai.chat.completions.create(
  model="meta-llama/Meta-Llama-3.1-70B-Instruct",
  messages=messages
)

print(second_response.choices[0].message.content)
```

```
Based on the NHTSA complaints, there are several issues with the 2024 Tesla Cybertruck, including:

1. Tire blowouts: There have been reports of tire blowouts, particularly with the Pirelli Scorpion ATR tires.
2. Electrical system issues: Some owners have reported electrical system failures, including critical error messages and loss of system redundancy.
3. Steering issues: There have been reports of steering problems, including power reduced and critical steering issue detected messages.
4. Suspension issues: Some owners have reported issues with the air suspension system, including inability to adjust and error messages.
5. Structure issues: There have been reports of structural issues, including misaligned doors and loose trim pieces.
6. Visibility issues: Some owners have reported visibility issues, including windshield wiper failures and snow accumulation on the headlights.
7. Fire risks: There have been reports of fire risks, including a complaint about the large driver's touchscreen getting extremely hot.
8. Latch issues: Some owners have reported issues with the door latches, including a complaint about the passenger side door swinging open while driving.
9. Propulsion system issues: There have been reports of propulsion system issues, including a complaint about the accelerator pedal slipping and lodging
```

In summary we have sent two messages to our LLM, the question and where to find more info

Working Example NHSTA Complaints

Step 5 –Second call to the LLM, execute the function and evaluate the response:

```
# get a new response from the model where it can see the function responses
second_response = openai.chat.completions.create(
  model="meta-llama/Meta-Llama-3.1-70B-Instruct",
  messages=messages
)

print(second_response.choices[0].message.content)
```

```
Based on the NHTSA complaints, there are several issues with the 2024 Tesla Cybertruck, including:

1. Tire blowouts: There have been reports of tire blowouts, particularly with the Pirelli Scorpion ATR tires.
2. Electrical system issues: Some owners have reported electrical system failures, including critical error messages and loss of system redundancy.
3. Steering issues: There have been reports of steering problems, including power reduced and critical steering issue detected messages.
4. Suspension issues: Some owners have reported issues with the air suspension system, including inability to adjust and error messages.
5. Structure issues: There have been reports of structural issues, including misaligned doors and loose trim pieces.
6. Visibility issues: Some owners have reported visibility issues, including windshield wiper failures and snow accumulation on the headlights.
7. Fire risks: There have been reports of fire risks, including a complaint about the large driver's touchscreen getting extremely hot.
8. Latch issues: Some owners have reported issues with the door latches, including a complaint about the passenger side door swinging open while driving.
9. Propulsion system issues: There have been reports of propulsion system issues, including a complaint about the accelerator pedal slipping and lodging
```

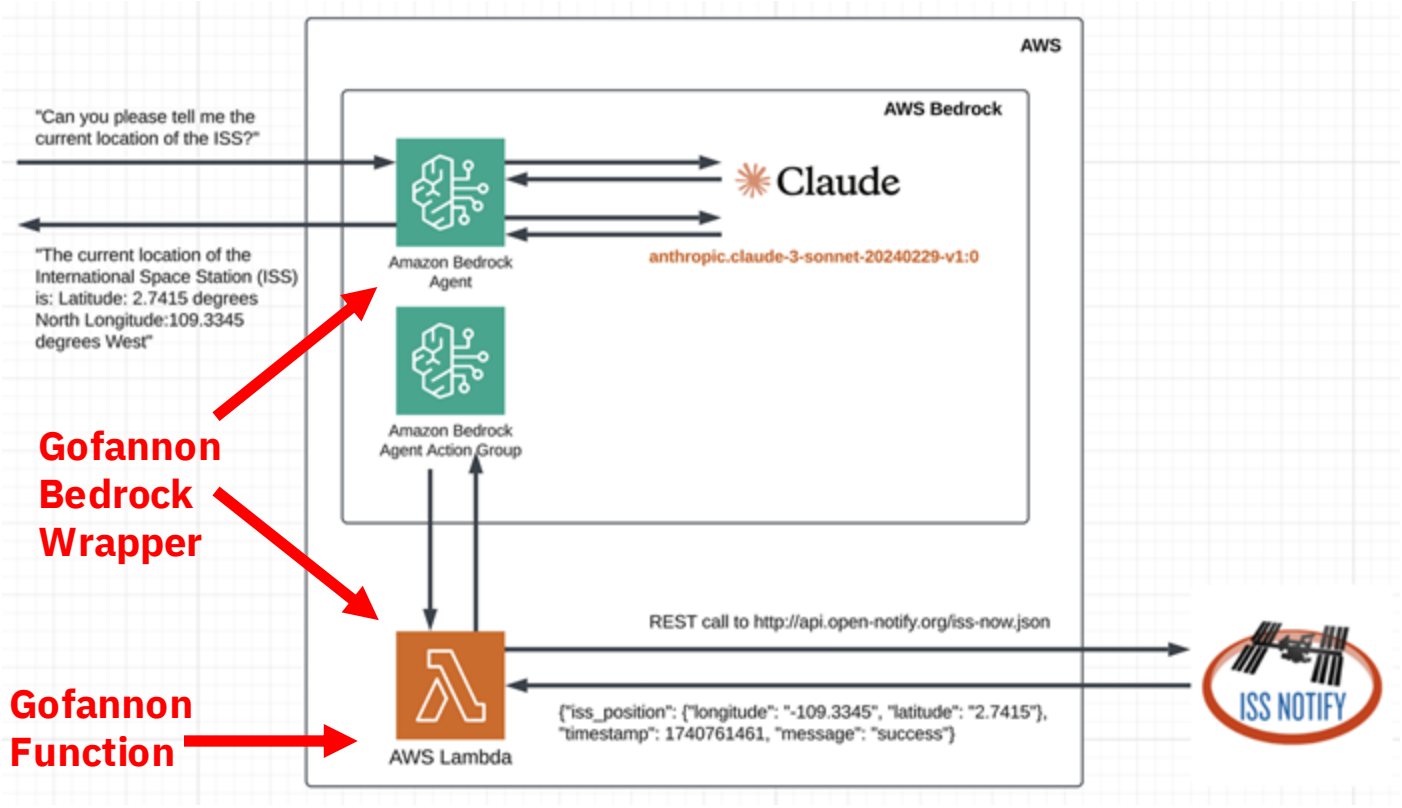
In summary we have sent two messages to our LLM, the question and where to find more info

Now Agents

Agents use tools + multiple LLM calls to create the illusion of magic. Let us now ruin the trick for you.

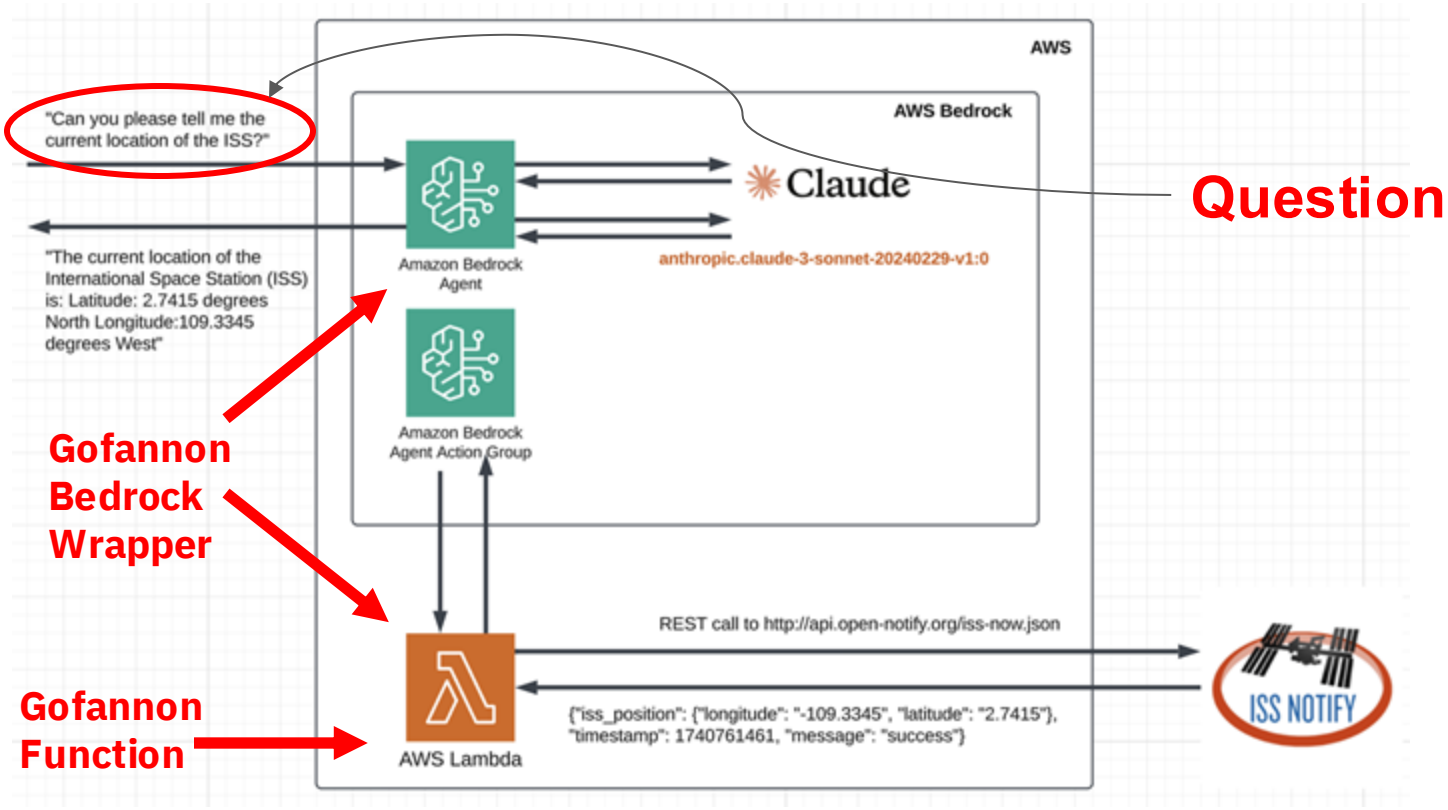
This example uses AWS Bedrock, which is the most complicated example. But it still fits on one slide, ergo the most complicated agents example is still pretty simple. **Ask questions if you don't understand!**

Now Agents


AWS
AWS Bedrock
"Can you please tell me the current location of the ISS?"
Amazon Bedrock Agent
Claude
anthropic.claude-3-sonnet-20240229-v1:0
"The current location of the International Space Station (ISS) is: Latitude: 2.7415 degrees North Longitude:109.3345 degrees West"
Amazon Bedrock Agent Action Group
Gofannon Bedrock Wrapper
REST call to http://api.open-notify.org/iss-now.json
ISS NOTIFY
{"iss_position": {"longitude": "-109.3345", "latitude": "2.7415"}, "timestamp": 1740761461, "message": "success"}
Gofannon Function
AWS Lambda

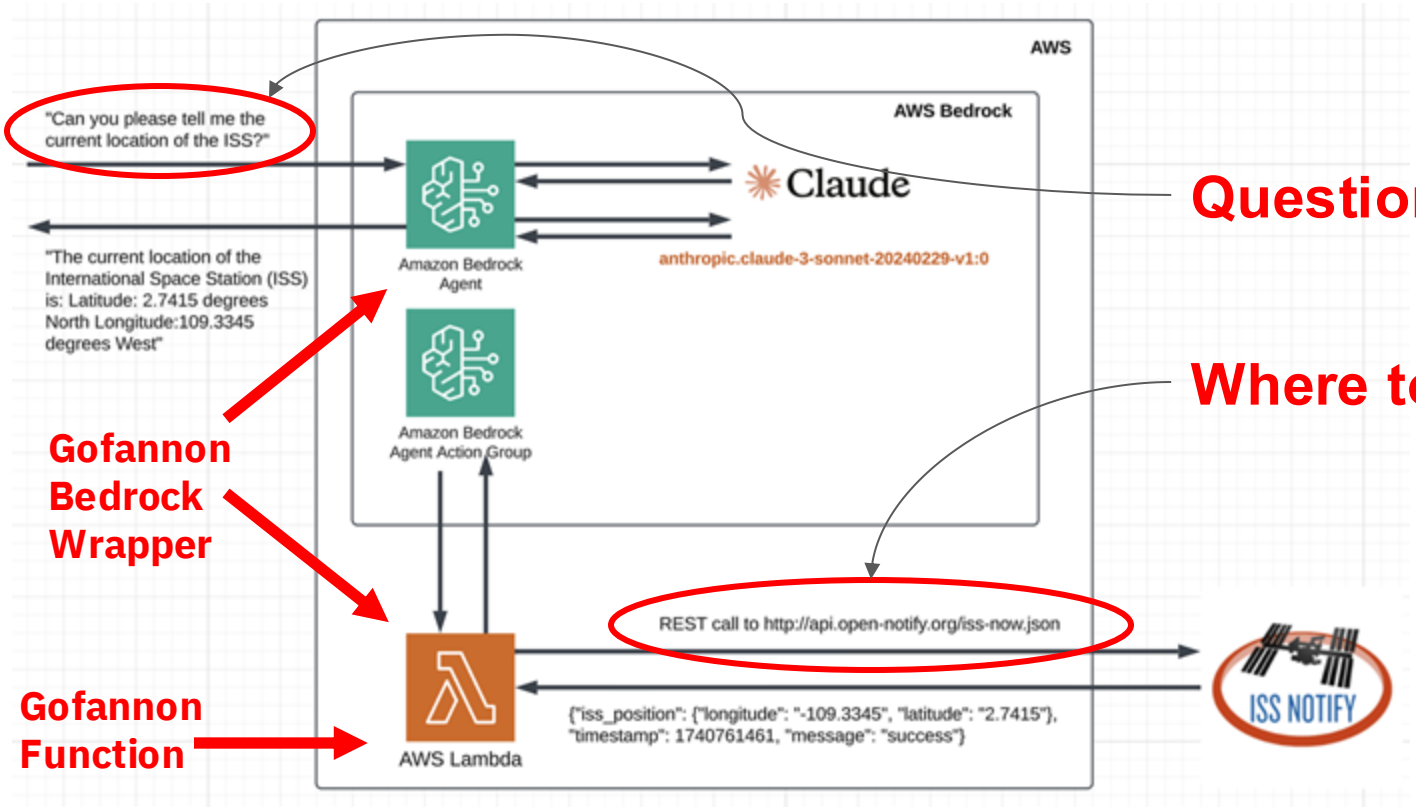

Gofannon
Gofannon: Empowering AI with legendary tools

Now Agents


AWS
AWS Bedrock
"Can you please tell me the current location of the ISS?"
Claude
Question
"The current location of the International Space Station (ISS) is: Latitude: 2.7415 degrees North Longitude:109.3345 degrees West"
Amazon Bedrock Agent
anthropic.claude-3-sonnet-20240229-v1:0
Amazon Bedrock Agent Action Group
Gofannon Bedrock Wrapper
REST call to http://api.open-notify.org/iss-now.json
ISS NOTIFY
{"iss_position": {"longitude": "-109.3345", "latitude": "2.7415"}, "timestamp": 1740761461, "message": "success"}
Gofannon Function
AWS Lambda


Gofannon
Gofannon: Empowering AI with legendary tools

Now Agents


AWS
AWS Bedrock
"Can you please tell me the current location of the ISS?"
Claude
anthropic.claude-3-sonnet-20240229-v1:0
Amazon Bedrock Agent
"The current location of the International Space Station (ISS) is: Latitude: 2.7415 degrees North Longitude:109.3345 degrees West"
Amazon Bedrock Agent Action Group
Question
Where to find more
Gofannon Bedrock Wrapper
REST call to http://api.open-notify.org/iss-now.json
ISS NOTIFY
{"iss_position": {"longitude": "-109.3345", "latitude": "2.7415"}, "timestamp": 1740761461, "message": "success"}
Gofannon Function
AWS Lambda


Gofannon
Gofannon: Empowering AI with legendary tools

Now Agents

lambda_function.py

```python
import json
from gofannon.open_notify_space.iss_locator import IssLocator

def lambda_handler(event, context):
    tool = IssLocator()
    result = tool.fn()
    action_response = {
        "actionGroup": event["actionGroup"],
        "apiPath": event["apiPath"],
        "httpMethod": event["httpMethod"],
        "httpStatusCode": 200,
        "responseBody": {"application/json": {"body": json.dumps(result)}},
    }
    api_response = {"messageVersion": "1.0", "response": action_response}

    return api_response
```

Gofannon Function

Bedrock Wrapper

Now Agents


```
    "traceId": "5c0289ef-b927-40ef-9617-f21fc3c2be37-0"
  },
  "rationale": {
    "text": "To get the current location of the International Space
      Station (ISS), I will need to call the GET
      ::gofannon_demo_ISSLocator_ag::/iss_locator function. This
      function does not require any parameters, so I can invoke it
      directly.",
    "traceId": "5c0289ef-b927-40ef-9617-f21fc3c2be37-0"
  },
  "invocationInput": [
    {
      "actionGroupInvocationInput": {
        "actionGroupName": "gofannon_demo_ISSLocator_ag",
        "apiPath": "/iss_locator",
        "executionType": "LAMBDA",
        "verb": "get"
      },
      "invocationType": "ACTION_GROUP",
      "traceId": "5c0289ef-b927-40ef-9617-f21fc3c2be37-0"
    }
  ],
  "observation": [
    {
```

JSON Ln 28, Col 1 Errors: 0 Warnings: 0

Rationale deduced by LLM

Tools: The Real Stars of the Agentic Show

Retrievers - Get your local weather, or how badly your DOGE coin bombed out overnight

Actuators - Open/close your windows, or stop the bleeding and sell your DOGE coin

Calculators - Calculate your energy savings, or the net loss on your DOGE debacle

Descriptions - Let the LLM know about the function

gofannon

Overview

https://the-ai-alliance.github.io/gofannon

About Gofannon

Gofannon is a set of tooling for building agents that simplifies pulling functions, LLMs, and frameworks together into working systems

From the AI Alliance

Born out of a need to wrangle the explosion of competing and complementary models and tools into harmony in order to get work done without costly rewrites and rework when things need to change

About the Name

“Gofannon is a name in Welsh mythology, specifically the name of a divine smith and craftsman, a counterpart to the Irish god Goibniu and the Gaulish deity Gobannos.” - Google AI

“God of the Forge”

Goibniu too hard to pronounce.

VCs can fund Goibniu or Gobannos 😎

Full Mythos Here

A (Constantly Growing) Collection of Tools

- arXiv
  - `get_article`
  - `search`
- Basic Math
  - `addition`
  - `subtraction`
  - `multiplication`
  - `division`
  - `exponential`
- Github
  - `clone_repo`
  - `commit_files`
  - `create_issue`
  - `get_repo_contents`
  - `list_issues`
  - `Read_issue`
  - `search`
- Local File System
  - `read_file`
  - `write_file`
  - `list_directory`
- Google
  - `search`
- Headless Browser
  - `get` (Get A Page with Javascript Rendering)
- NASA
  - `apod` (Photo of the Day
- Open Notify
  - `iss_locator`
- Many More…

Cross Framework Compatibility

- Write once - run anywhere
- Covered at length in next section

Contributor Friendly / Open Source Education

- Robust Library of Examples
- Curated Contribution Paths

Cross-Framework Interoperability

Tradeoff

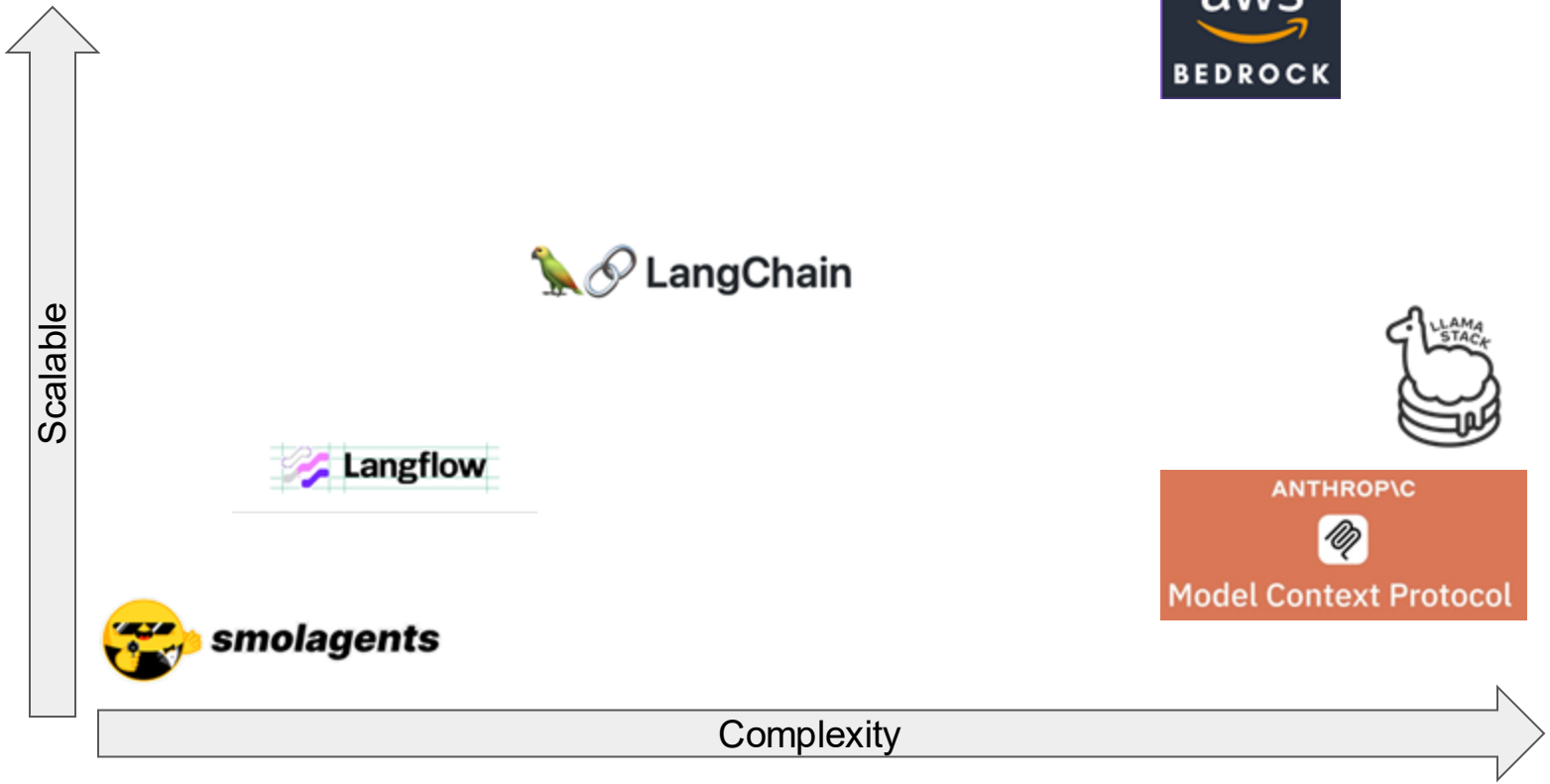
Concentrate on Functionality / Defer Scaling

- May be difficult to retrofit scaling in

Concentrate on Scalability / Defer Functionality

- Worried about productionalizing before testing market fit

A **Very** Opinionated Quadrant



Write Once - Run Anywhere

Solves the Tradeoff: export tools and functions to multiple frameworks

Example: Prototype with `smolagents`, deploy to production on Bedrock

Roster of Current Agentic Frameworks

Based on Merged and Open PRs on April 23, 2025

LangChain

smolagents

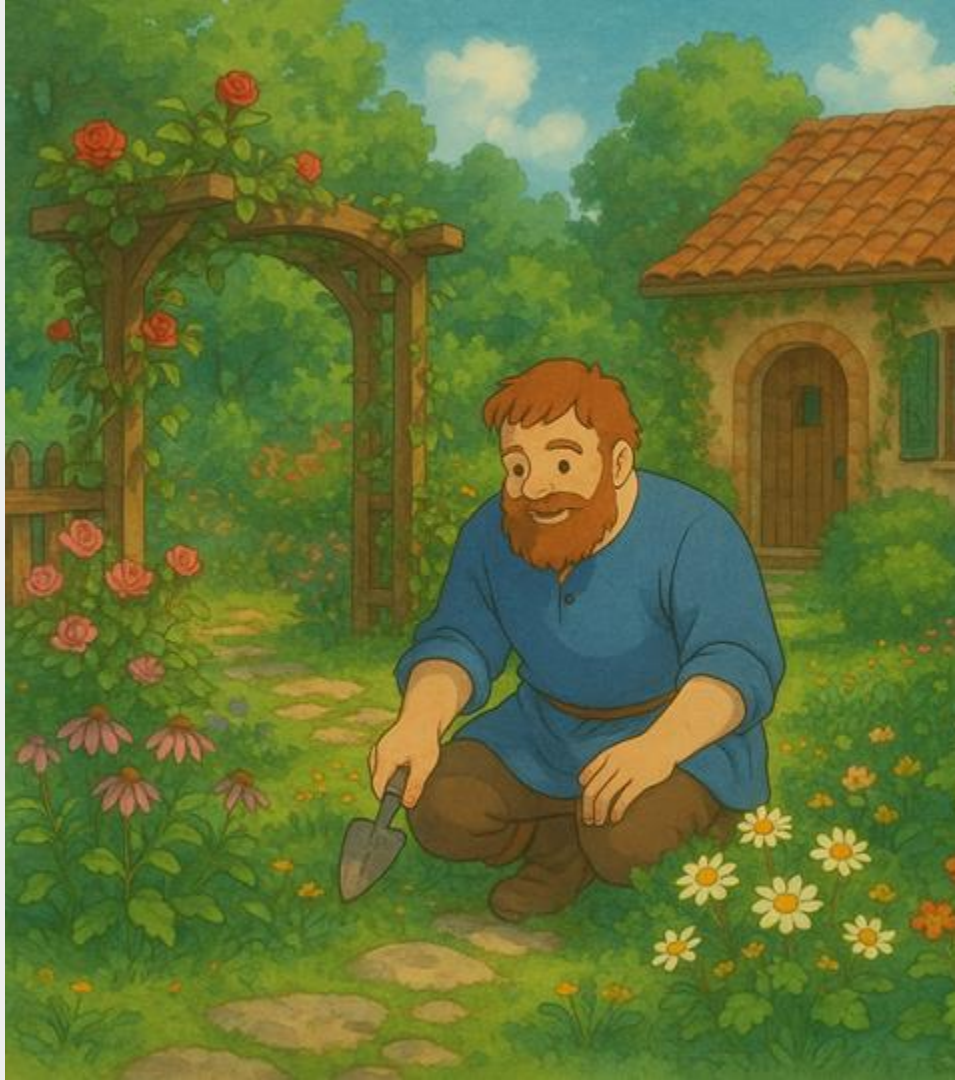
aws BEDROCK

Langflow

ANTHROP\C Model Context Protocol

LLAMA STACK

Contributor Friendly



Curated Contribution Pathways

- First Issue
- Adding a new Tool
- Adding a new Framework
- Adding a new Example (Under Construction)

Community Communication

- https://the-ai-alliance.github.io/gofannon/about
- Async: Github Discussions
- Sync: Discord

The Leaderboard

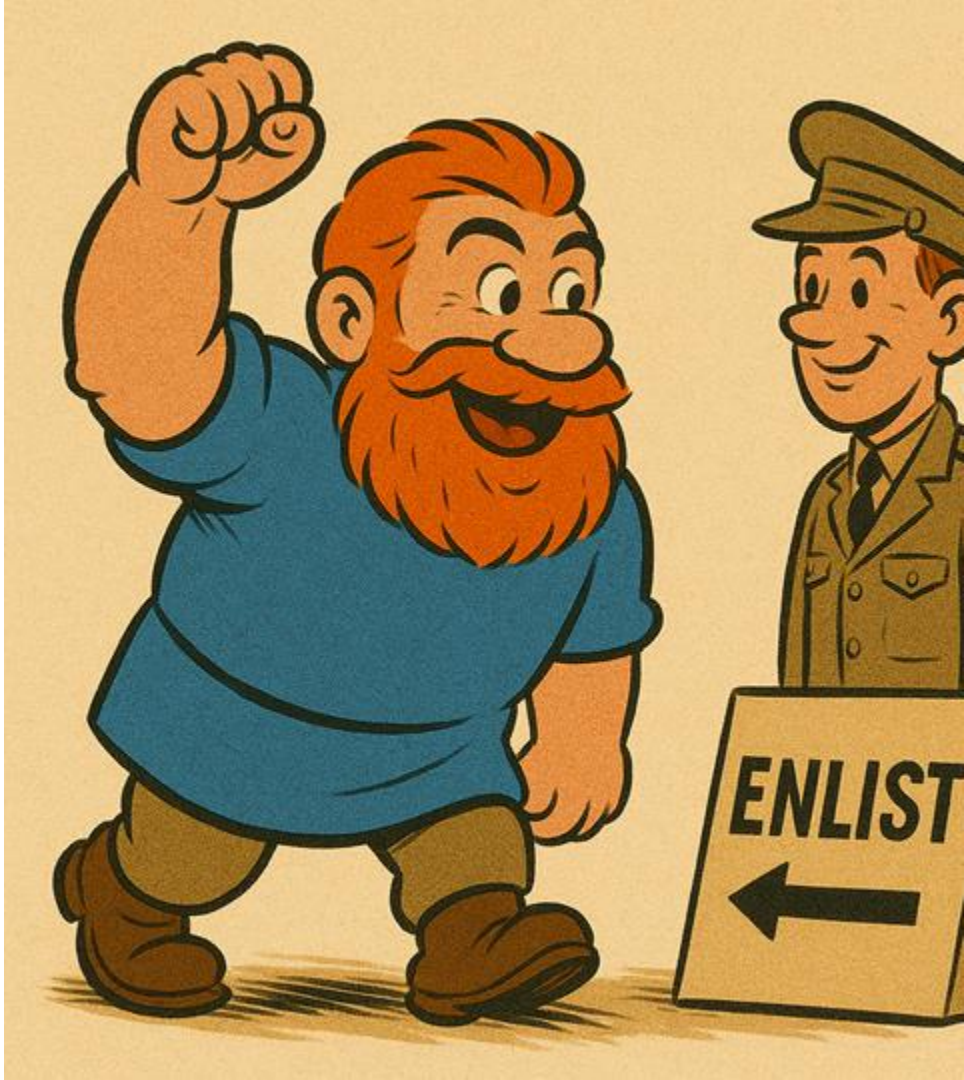
Conclusion

Time to land this plane

Recap of What We've Covered

- How LLMs, Tools, and Agents Work
- Gofannon Overview
- Cross Framework Interoperability
- Contributor Friendly

Join Our Community



Star Us on Github

Give a Talk

Contribute a Tool

Examples at:

https://github.com/The-AI-Alliance/gofannon/pulls?q=is%3Apr+label%3Atool

Thank You