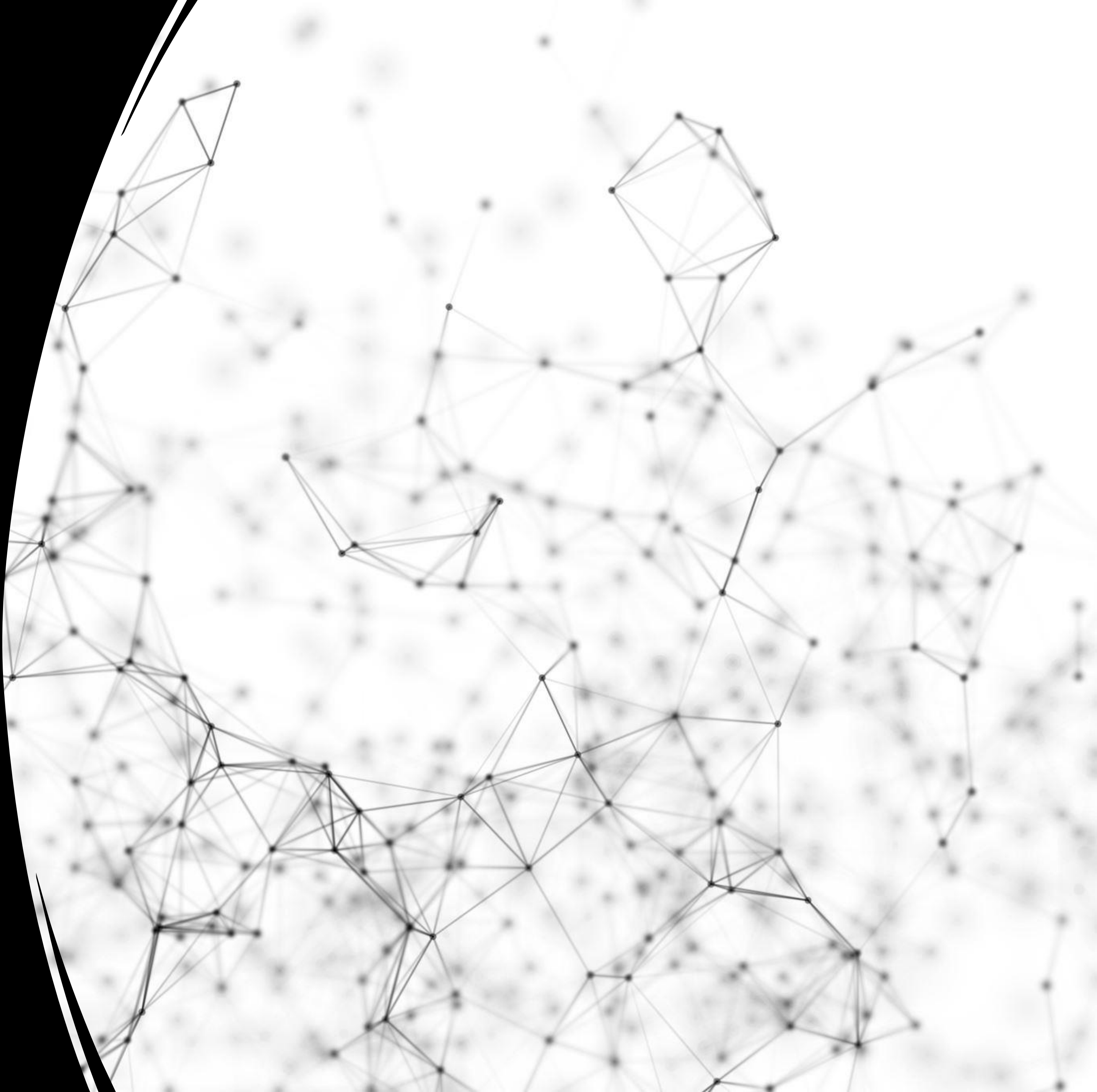
Tying your platforms together for action planning

Joshua Lochner



"excited to be presenting today"

BLUF : what will we cover in this talk?

I will share the process I went through to Build a system that could be used to query an infra for current issues across a few platforms. I will be discussing some lessons learned.

My examples are built using tools mostly in Azure.

I am cheap, the tools I use are emphasizing cost over performance.

If this doesn’t meet your needs, I would say now is a good time to dip into another talk.

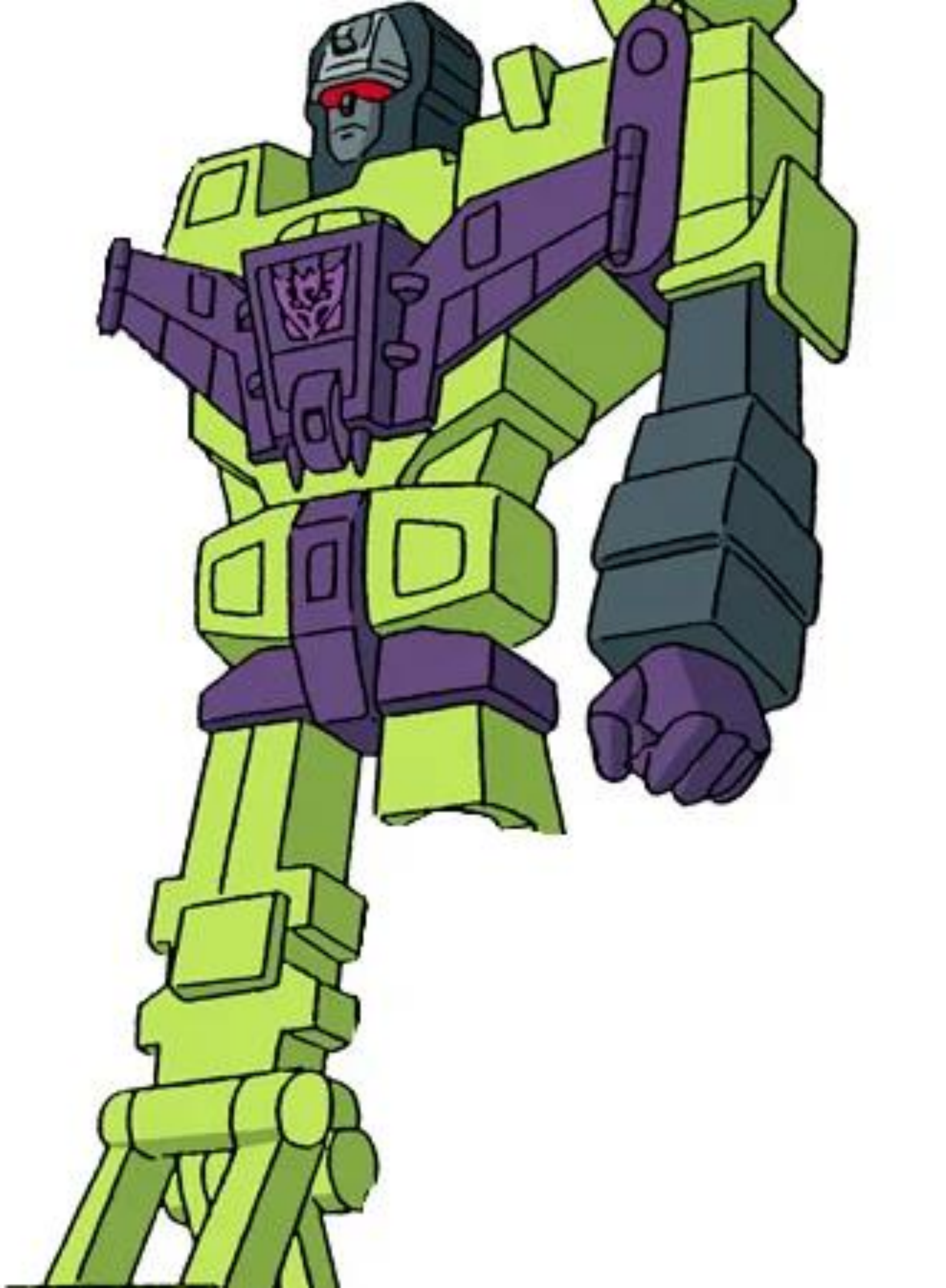
My feelings will probably not be hurt.

Goal Repo state

- Step 0 – build the repo/github
- Step 1 – build the infra
- Step2 – Build SPs and connections for MCP servers
- Step3 - convert the AI project from click ops to Terraform deployments
- Step4 – Create demo resources
- Step5 – Clean Up

Current state

- Step 0 – build the repo/github
- Step 1 – build the infra
- Step2 – Build SPs and connections for MCP servers
- Step3 - convert the AI project from click ops to Terraform deployments
- ~~Step4 – Create demo resources~~
- ~~Step5 – Clean Up~~



Overview and Objectives

Platform Integration
Combining Azure Policy, MCP servers, SQL, and Jira Cloud to build a scalable action planning workflow.

Agentic Workflow Benefits
Using workflows to normalize compliance signals and enrich them with contextual data effectively.

Automated Remediation
Generating actionable Jira tickets for project managers to streamline compliance remediation pipelines.

Target Audience
Presentation tailored for senior engineers and project managers focused on automated remediation improvements.

Still cool to dip

I have security center and/or Security hub…

“Here are our current vulns…”

But the backlog is horrendous, so I'm just
going to go ahead and have you red bag it.

You got a Dashboard of all your stuff that is bad, but….

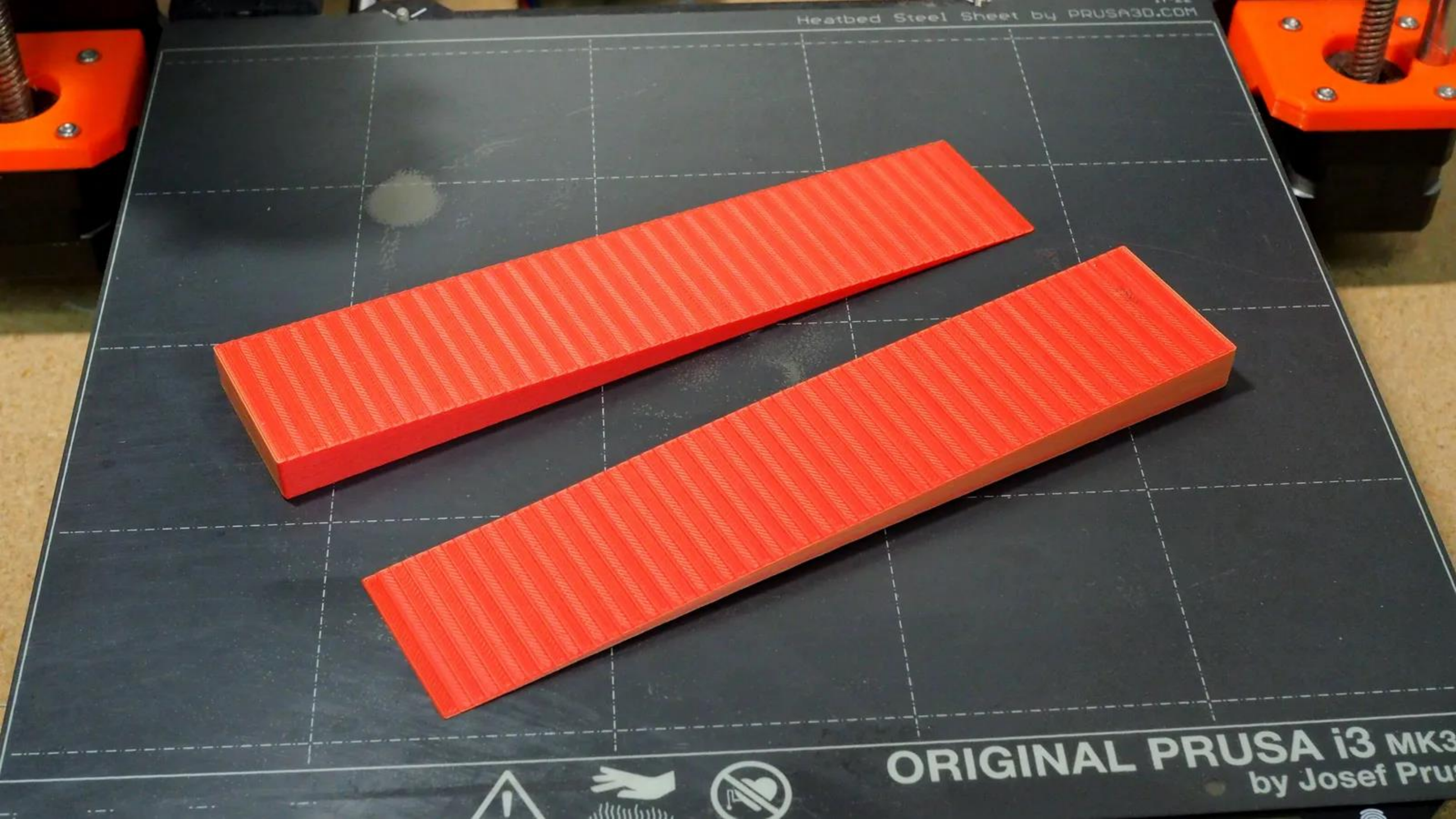
The Devastating building (and learning) process

250
#F0ECE1
9010

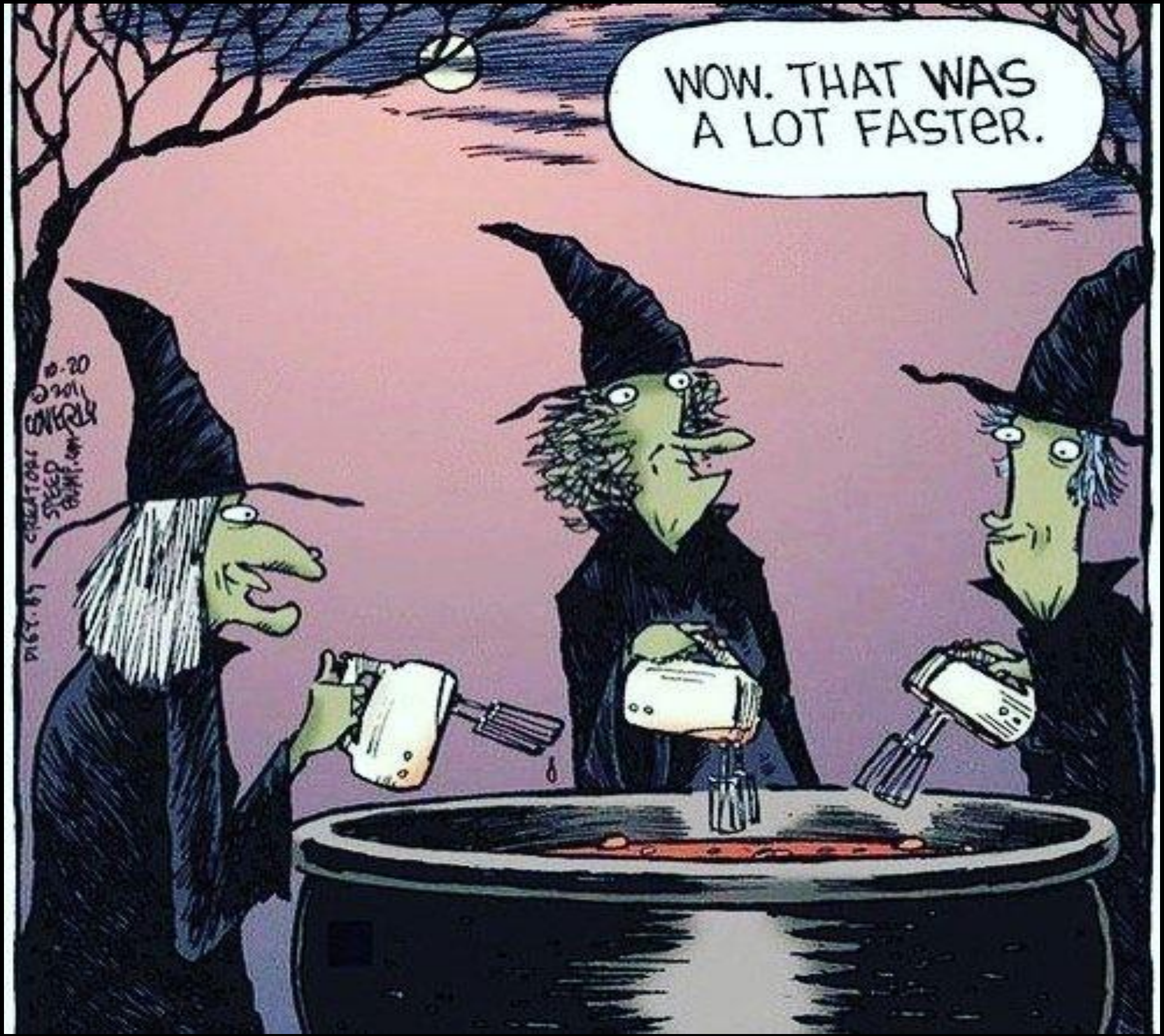
Lesson Learned: Are we really, measurably better?

- Put metrics of success on paper before starting.
- Get measurements now.
- It will help in objective review of what is being created
- How many hours are we talking about saving here?

NBC
SNL | hulu


Heatbed Steel Sheet by PRUSA3D.COM
ORIGINAL PRUSA i3 MK3
by Josef Pru

Boil Boil Toil and trouble



Lesson: Verify data for value proposal
Long ticket time != lots of real man hours

Time to dive in

https://azure.microsoft.com/en-us/pricing/purchase-options/azure-account?icid=azurefreeaccount

Free? Well, free adjacent. Or not at all free.

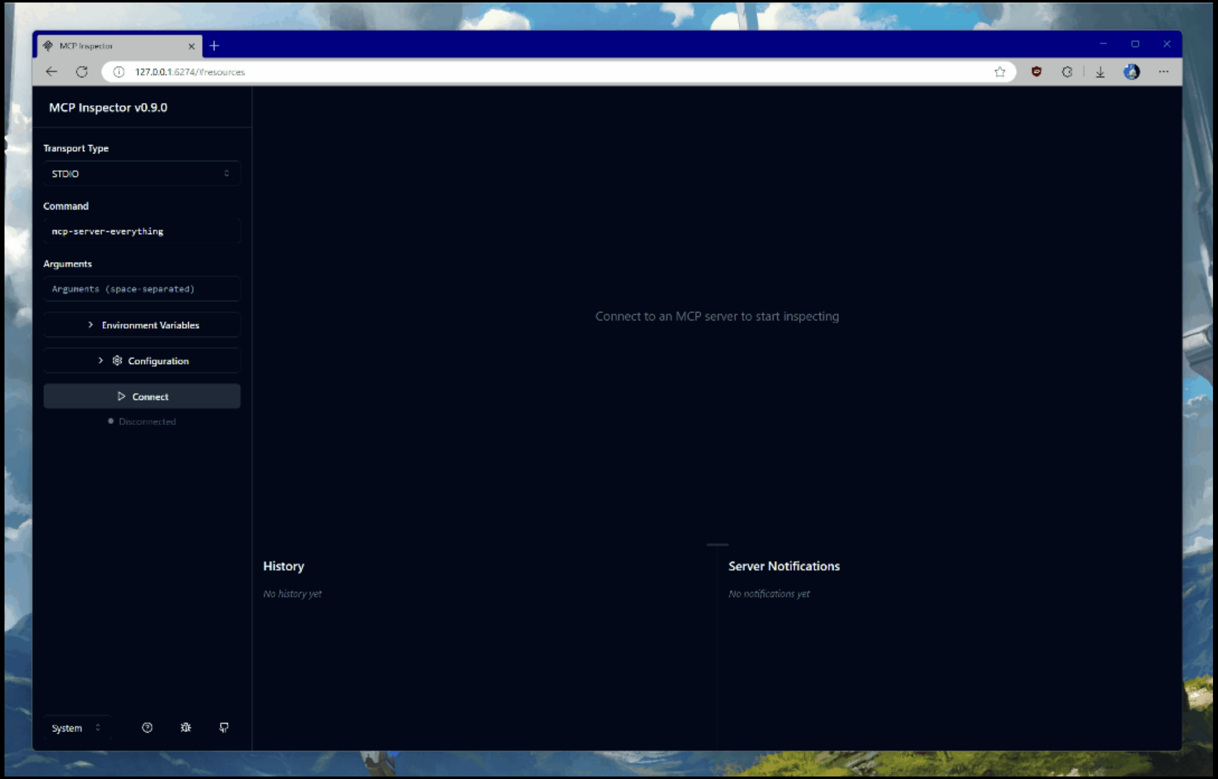
Nice place to start:
https://github.com/alejandro-ao/mcp-server-example best video

https://learn.microsoft.com/en-us/azure/developer/azure-mcp-server/get-started/languages/python?tabs=azure-cli

https://github.com/Azure-Samples/AI-Gateway/tree/main/labs

Then: https://den.dev/blog/remote-mcp-server/

Secure Remote MCP Servers With Entra ID And Azure API Management



MCP Inspector v0.9.0

Transport Type: STDIO

Command: mcp-server-everything

Arguments: Arguments (space-separated)

Disconnected

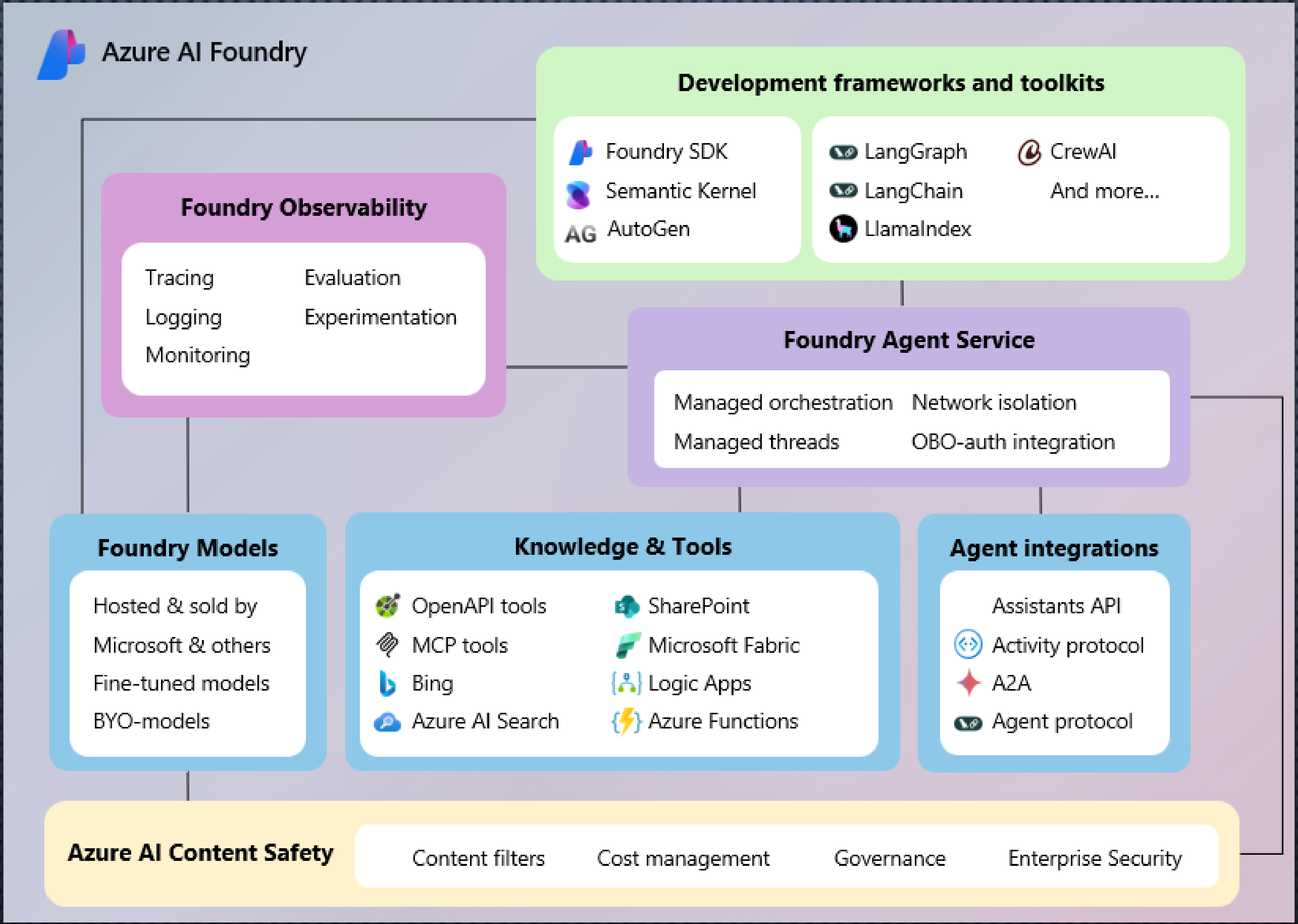
Connect to an MCP server to start inspecting

History

No history yet

Server Notifications

No notifications yet



“Whats in the box”

MCP servers and other useful repos (beware imposter MCP servers)

- https://github.com/atlassian/atlassian-mcp-server
- https://github.com/microsoft/mcp/tree/main/servers/Azure.Mcp.Server
- https://github.com/Azure-Samples/remote-mcp-functions-python

Estimated Data Flow

STEP	DESCRIPTION
1	PolicyStateChanged event or scheduled pull
2	Adapter queries PolicyStates latest/queryResults
3	Emit to Kafka topic policy.events
4	Normalizer maps and enriches data
5	Planner selects runbook and generates action plan
6	Orchestrator upserts into Jira ticket
7	Ack and update state; export metrics/logs

Lesson: Break down what you are going to want to do, and make sure you keep endpoints accessible as you go.

Finding Commonality Across Policies

Why: Reduce noise, improve PM efficiency

How:

- Normalize → tag with resourceId, policyId, policyAssignment
- Use MCP tools for semantic similarity or keyword match
- Build clusters for consolidated remediation

Output: Jira epics for clusters, subtasks for individual resources

Lesson: validate your results frequently and with sets that you can understand

CAN'T YOU DO ANYTHING RIGHT?
Copyright 2003 Randy Glasbergen. www.glasbergen.com

Validation

Field Mappings and Idempotency

Field Mapping Overview

Normalized fields are mapped to Jira fields including project key, issue type, and custom fields for integration.

Custom Fields Usage

Custom fields like External Correlation ID and Severity enable detailed issue tracking and integration robustness.

Idempotency Mechanism

- Idempotency is ensured by searching existing issues via External Correlation ID before creating new ones. Idempotency-Key.

Assignee and Routing

Assignees are resolved by accountId and components route tickets to appropriate teams efficiently.

Expectations of data

+

=

Upon closer validation of the data

+

=

Jira Cloud Integration

Azure Policy Compliance Signals

Compliance Data Interfaces

Azure Policy offers compliance data via the PolicyStates REST API and Event Grid event notifications.

PolicyStates Querying

PolicyStates API allows querying compliance at subscription and resource scopes using filters and time windows.

Event Grid Notifications

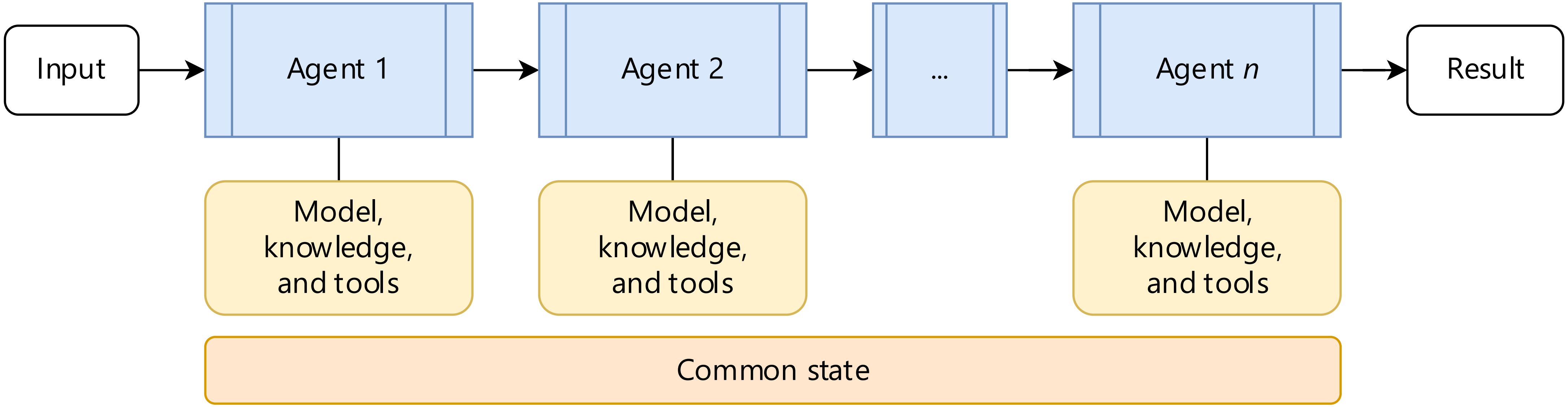
Event Grid emits events such as PolicyStateCreated and PolicyStateChanged after evaluations for real-time updates. Jira Idempotent key.

MCP adapter Processing

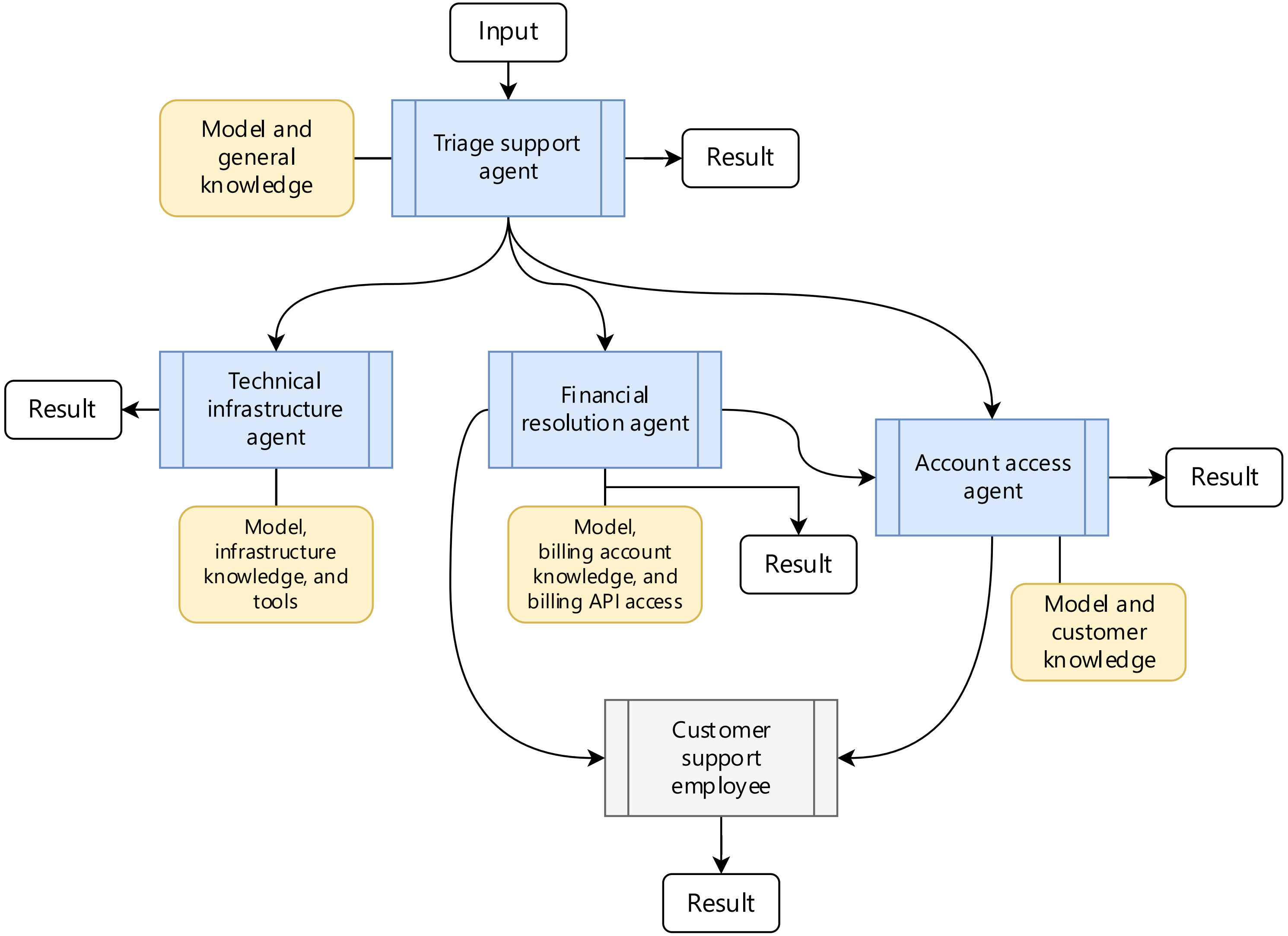
MCP adapters normalize data, handle pagination and idempotency, and emit to Kafka for downstream processing.

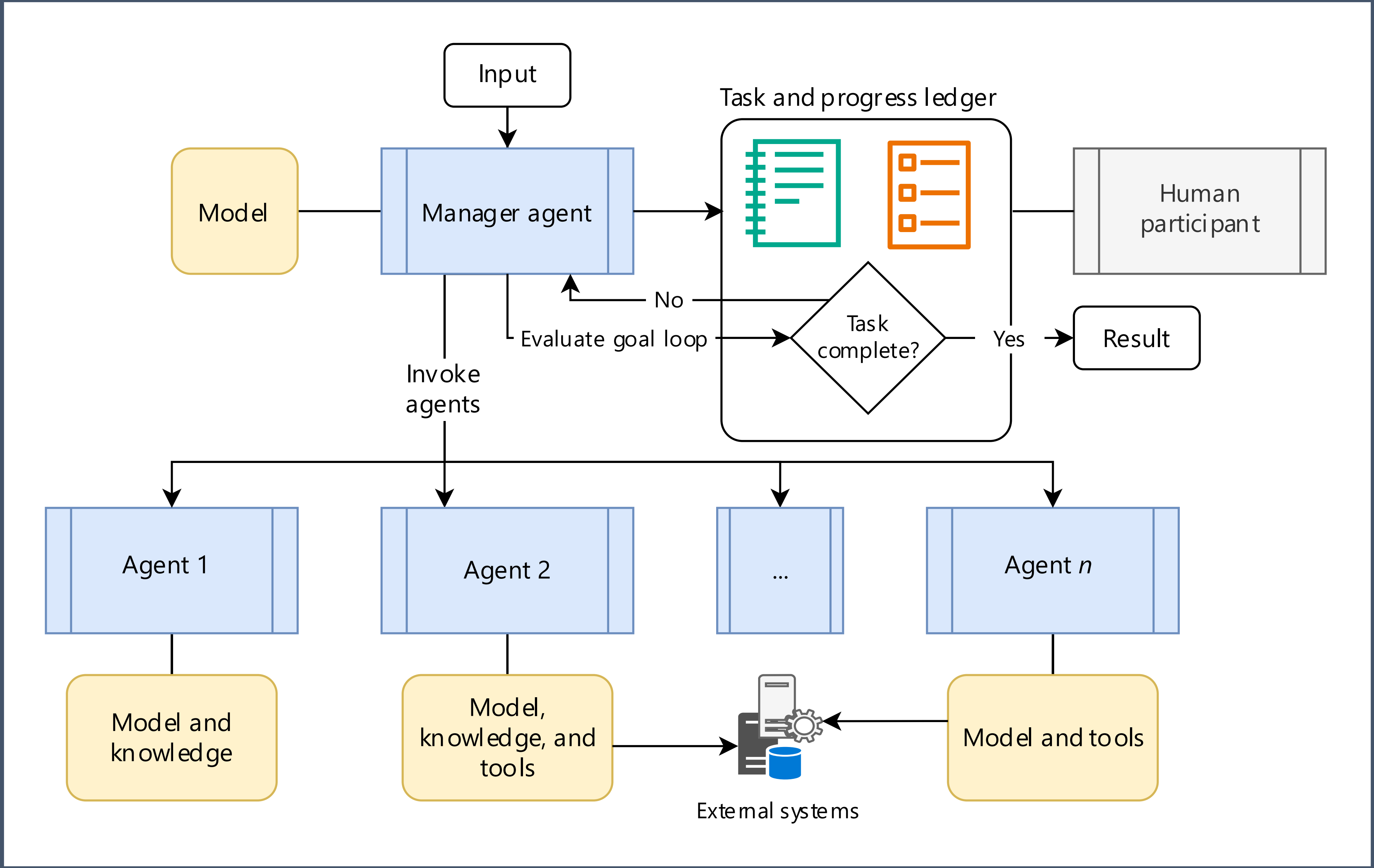
```
[{
  "id": "5829794FCB5075FCF585476619577B5A5A30E52C84842CBD4E2AD73996714C4C",
  "topic": "/subscriptions/<SubscriptionID>",
  "subject": "/subscriptions/<SubscriptionID>/resourceGroups/<ResourceGroup>/providers/<ProviderNamespace>/<ResourceType>/<ResourceName>",
  "data": {
    "timestamp": "2021-03-27T18:37:42.4496956Z",
    "policyAssignmentId": "<policy-assignment-scope>/providers/microsoft.authorization/policyassignments/<policy-assignment-name>",
    "policyDefinitionId": "<policy-definition-scope>/providers/microsoft.authorization/policydefinitions/<policy-definition-name>",
    "policyDefinitionReferenceId": "",
    "complianceState": "NonCompliant",
    "subscriptionId": "<subscription-id>",
    "complianceReasonCode": ""
  },
  "eventType": "Microsoft.PolicyInsights.PolicyStateCreated",
  "eventTime": "2021-03-27T18:37:42.5241536Z",
  "dataVersion": "1",
  "metadataVersion": "1"
}]
```

Unexpected results


Input
Agent 1
Agent 2
...
Agent n
Result
Model, knowledge, and tools
Model, knowledge, and tools
Model, knowledge, and tools
Common state

BY DIABLITO 666


Input
Model and general knowledge
Triage support agent
Result
Result
Technical infrastructure agent
Financial resolution agent
Account access agent
Result
Model, infrastructure knowledge, and tools
Model, billing account knowledge, and billing API access
Result
Model and customer knowledge
Customer support employee
Result


Input
Task and progress ledger
Model
Manager agent
Human participant
No
Task complete?
Evaluate goal loop
Yes
Result
Invoke agents
Agent 1
Agent 2
...
Agent n
Model and knowledge
Model, knowledge, and tools
Model and tools
External systems

RBAC

API Endpoints and Authentication -> MCP Server permission sets

Core API Endpoints

Key Jira API endpoints enable issue creation, searching, updating, and property management effectively.

Authentication Methods

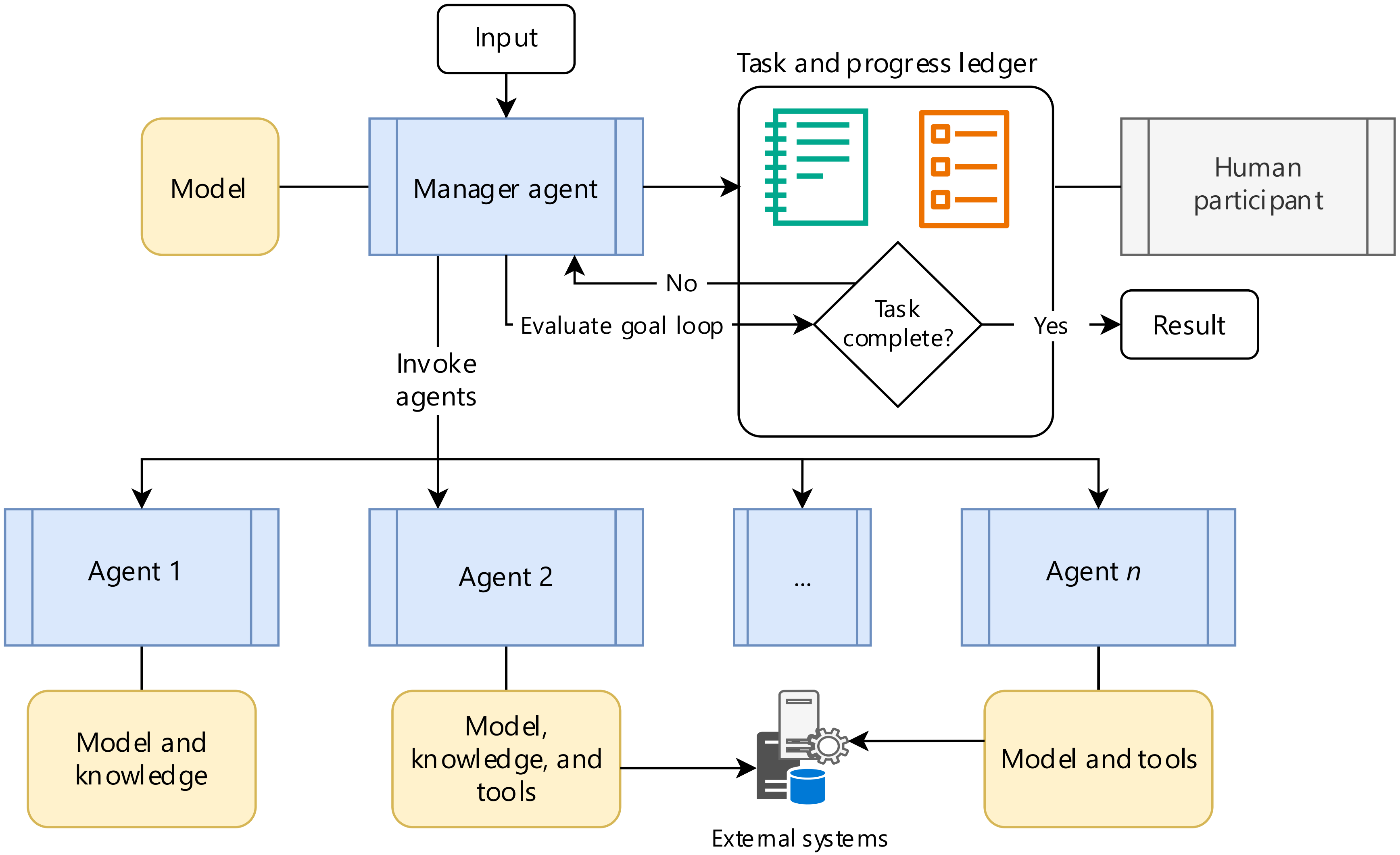
OAuth 2.0, Connect JWT, and API tokens provide secure authentication for accessing Jira API endpoints.

Rich Content Format

Atlassian Document Format supports rich descriptions within issue bodies enhancing content presentation.

Scoped Permissions

Properly scoped permissions ensure users can create and edit issues securely and efficiently.


Input
Task and progress ledger
Model
Manager agent
Human participant
No
Task complete?
Evaluate goal loop
Yes
Result
Invoke agents
Agent 1
Agent 2
...
Agent n
Model and knowledge
Model, knowledge, and tools
Model and tools
Extemal systems

SHIM in some ad data https://learn.microsoft.com/en-us/windows/win32/adschema/a-department

AD Query and Result:

```
get-aduser -filter * -properties Department, DepartmentNumber, mail -searchbase
"OU=<USERS>,DC=<COMPANYNAME>,DC=com" | Select Enabled, Name, SamAccountName, Department,
@{Label=”BUSINESSUNIT";e={$_.departmentNumber}}, @{Label="Email";e={$_.mail}}
```

Some Contexts?

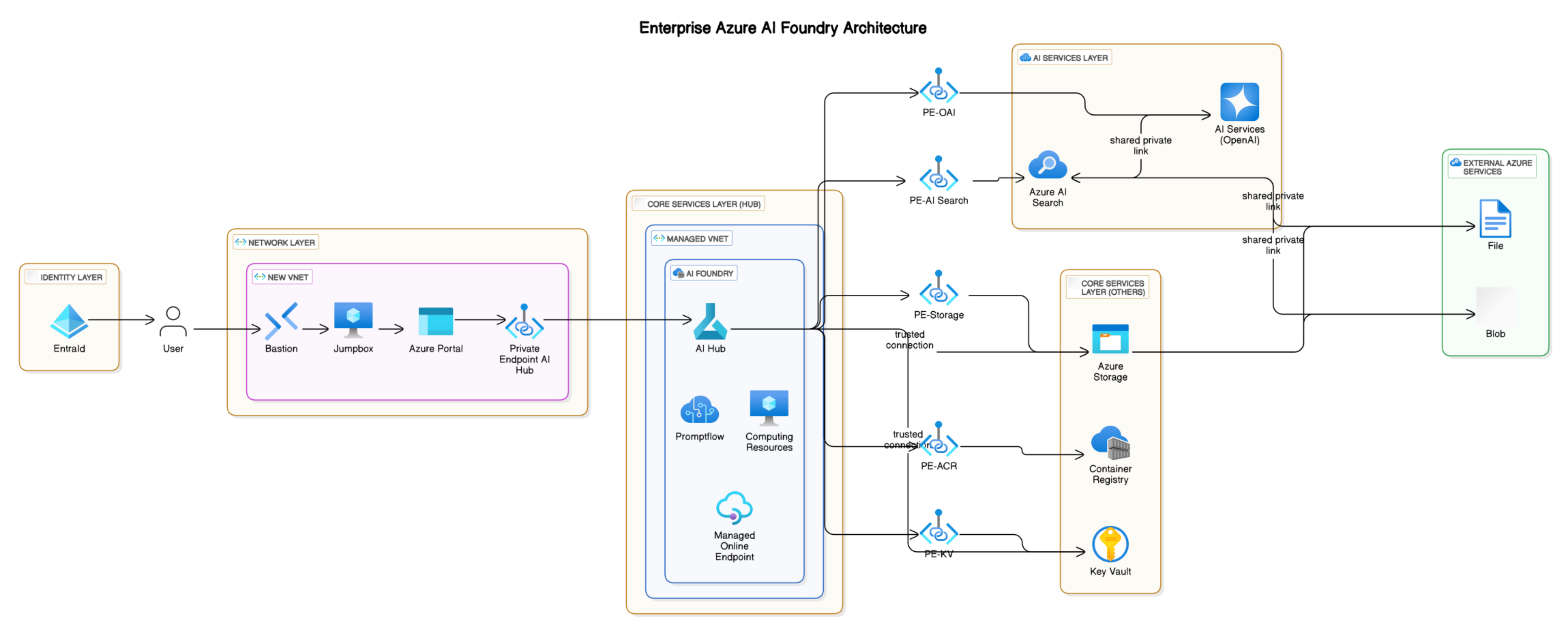
SP 800-128, *Guide for Security-Focused Configuration Management*

SP 800-40, *Guide to Enterprise Patch Management Planning*

Define Maximum Tolerable Downtime (MTD): As part of contingency planning (SP 800-34)

800-204C Implementation of DevSecOps for a Microservices-based Application with Service Mesh

Also pretty darn good for a reference: https://github.com/Azure/terraform-azurerm-avm-ptn-ai-foundry-enterprise/tree/main



Enterprise Azure AI Foundry Architecture

LESSON: KEEP IT SIMPLE

Work through one plan.

Validate (are you creating a ticket, what is the schema?) are you a hub and spoke topology, what are you expecting re: gateways?

Find examples from more trusted providers

https://github.com/Azure-Samples/remote-mcp-functions-python

System Message Framework

- https://techcommunity.microsoft.com/blog/educatordeveloperblog/ai-agents-building-trustworthy-agents--part-6/4399202?wt.mc_id=studentamb_258691

- **Meta System Message:** A template prompt used by the LLM to generate agent-specific system prompts. This meta prompt sets the overall tone and expectations for agent behavior.
- **Basic Prompt:** A concise description of the agent's role, tasks, and responsibilities.
- **LLM-Generated System Message:** Combine the meta system message and the basic prompt to generate a more refined and structured system message for the agent. The example in the full blog post demonstrates the output of this process.
- **Iterate and Improve:** Refine the basic prompt and regenerate the system message until it effectively guides the agent's behavior.

LESSON: …BUT LOG LIKE ITS GOING OUT OF STYLE

Capture contexts; requests/replies; dump it all into storage somewhere to reference

MCP Inspector

Agents

How many is too many?

Check your logic and your tool count. Azure's MCP Server, for example lets you define service namespaces.

Azmcp server start –namespace <service-name>

(Like keyvault or storage)

Found that the broad use of one MCP server for a work set seems to perform better than trying to manage state files for a bunch of small really specific ones, or having one monolithic one.

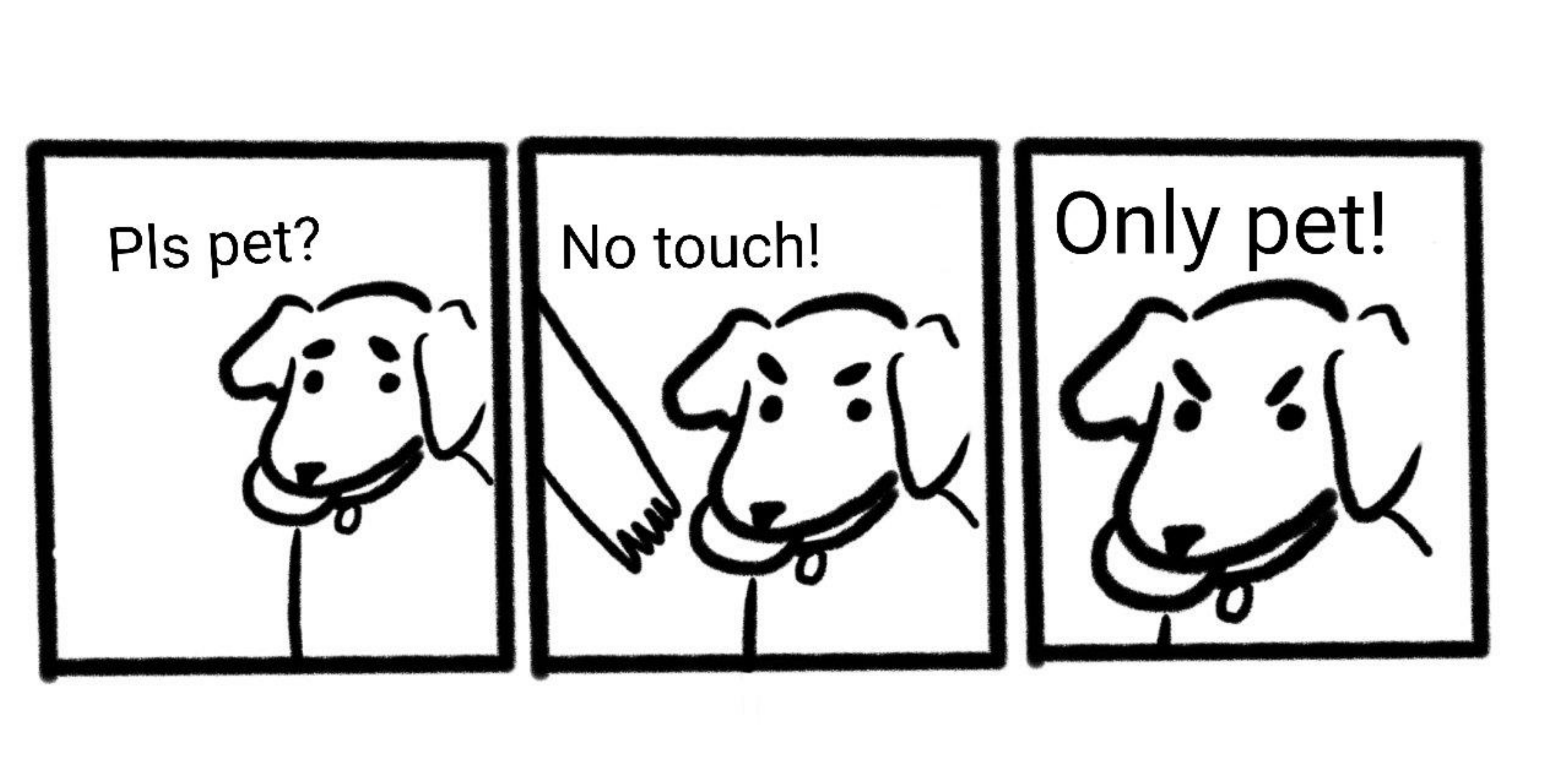
Duplicate MCPs might not be good.

You can tell because they are hanging out with the Brotherhood of Evil Mutants.

Purposeful agents and smaller functional MCP

- Don’t over-scope

Azure MCP Servers role permissions
Checking for READ ACCESS (no changes)



Check your API keys, check what they can do.

Reminder : Secrets/Environment Variables

Test locally then send it up

Cost will be incurred, depending on the resources you use to roll this out.

That’s on you.

Be Aware!!!

Operations and Observability

Observability and Metrics

Operational Metrics Overview

Key operational metrics include Kafka lag, DLQ depth, and Jira API error rates for system monitoring.

Log Data Insights

Logs capture trace IDs, tool invocations, and planner decisions to provide detailed system insights.

Dashboard Visualizations

Dashboards visualize compliance burndown and SLA burnup for ticket and policy tracking.

Monitoring Tools Usage

Tools like Log Analytics and Grafana monitor system health and aid in pipeline tuning.

Get your PAT (personal access token) down

Before you try to "azd up" or you will get nuget errors. Note, some of us use github actions not azure devops so this will seem weird.

https://www.softwaredeveloper.blog/private-nuget-feed-in-docker

Really, probably:

https://github.com/microsoft/artifacts-credprovider

AZTFEXPORT
CONVERTING JSON TO TF BY HAND
imgflip.com

ADP up to an RG then back down to .tf?

Expensive models

Depending on the purpose and amount of tasks it increases your cost efficiency to get your model in use to be “just dumb enough”

Key Takeaways

Integration for Scalability

Combining MCP servers with Azure Policy, Kafka, and Jira Cloud supports scalable and reliable workflows for action planning.

Data Normalization and Guardrails

Early data normalization and applying guardrails during planning are essential to maintain data integrity and control.

Dumb Models can be fine

Log, log, and look at those logs