Dapr

Processing one million data points daily with Dapr

Siri Varma Vegiraju
Software Engineer

Siri Varma Vegiraju

- Software Engineer
- Freelance Contributor
- Love hiking
- LinkedIn: /sirivarma
- Email: siri.varma@outlook.com
- Twitter: @sirivegiraju

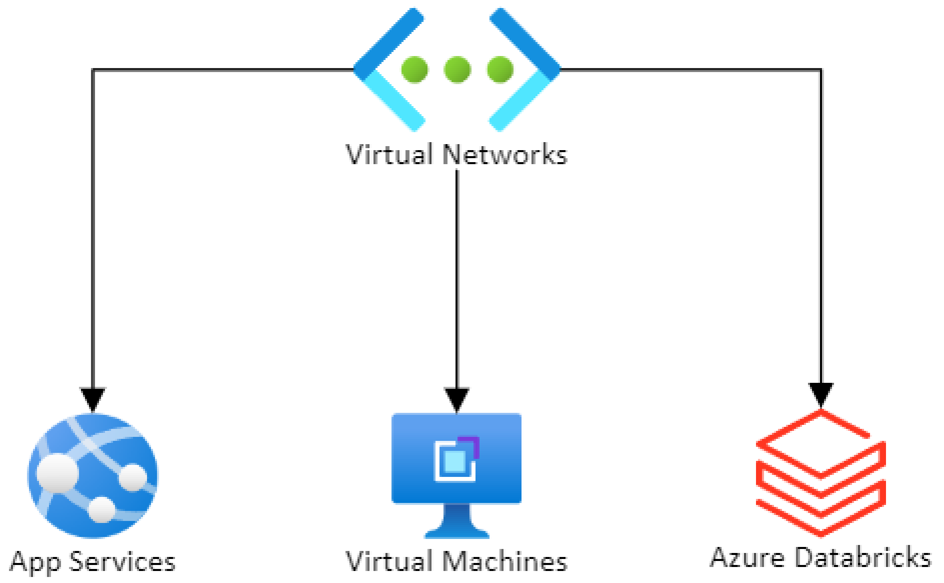
Agenda

- Overview of the business problem
- Architecture
  - Before Dapr.
  - After Dapr.
- Learnings from Dapr migration
- What's Next ?

Network Rules Recommendations for Protecting Azure workloads

Business Problem

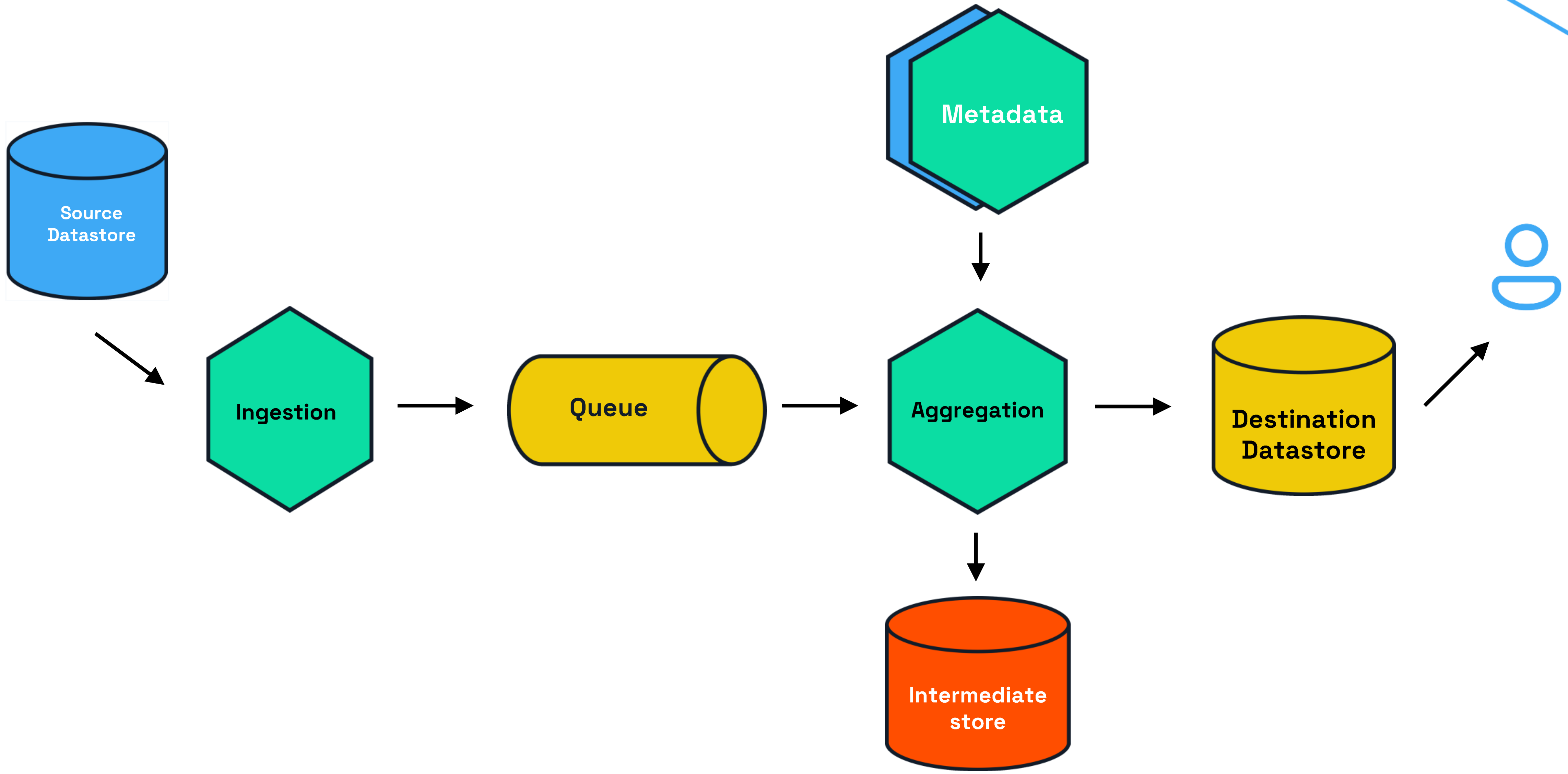
Apply rules at the Virtual Network level



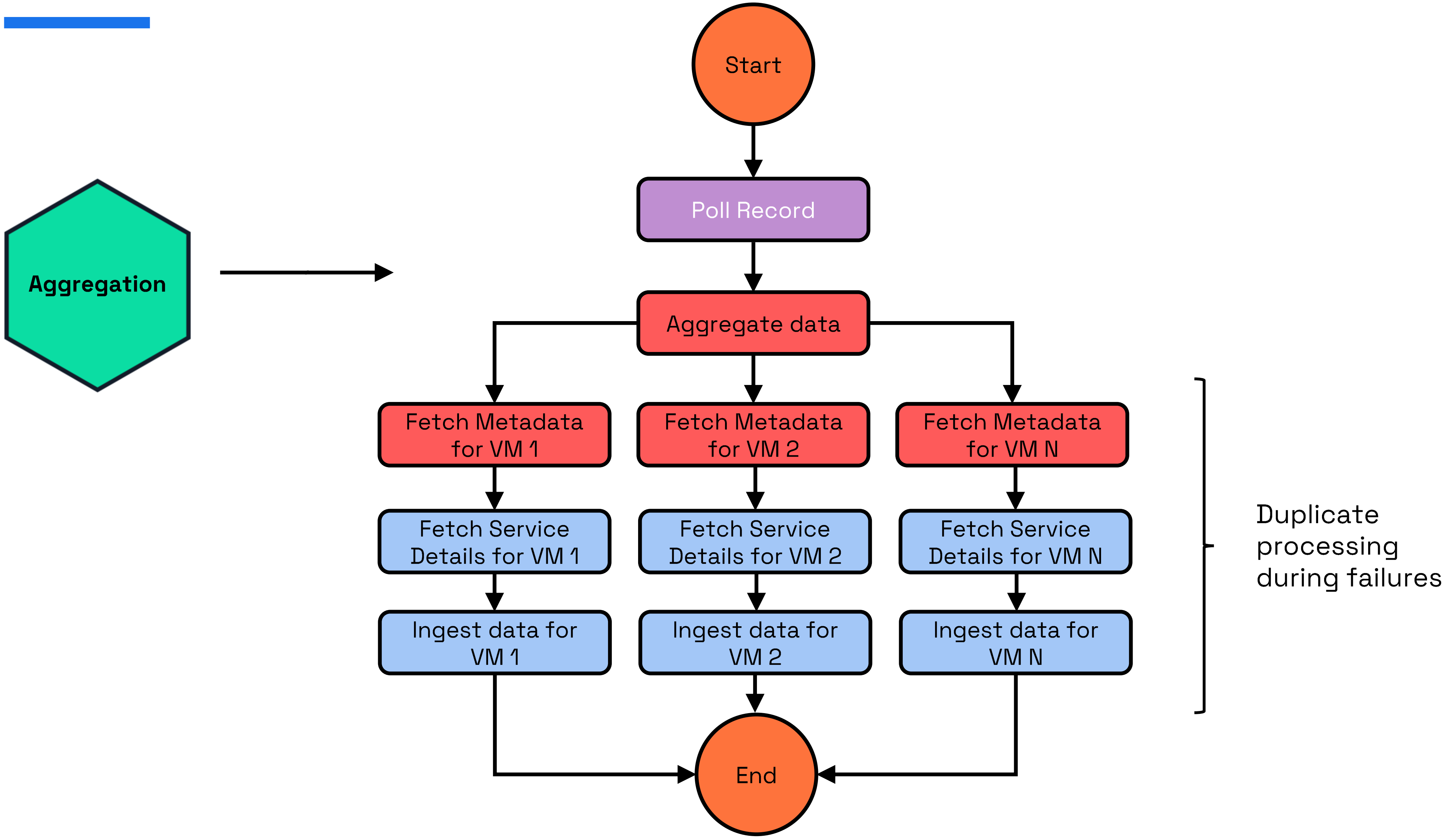
Example of a rule

```
{
    "Priority": 999,
    "Name": "AllowInternet",
    "NetworkGroupName": "Team",
    "Action": "Allow",
    "Direction": "Inbound",
    "Protocol": "Any",
    "SourceType": "IpAddress",
    "Source": "*",
    "SourcePorts": "1-65535",
    "DestType": "IpAddress",
    "Dest": "*",
    "DestPorts": "1-65535",
}
```

Before Dapr


Metadata
Source Datastore
Ingestion
Queue
Aggregation
Destination Datastore
Intermediate store

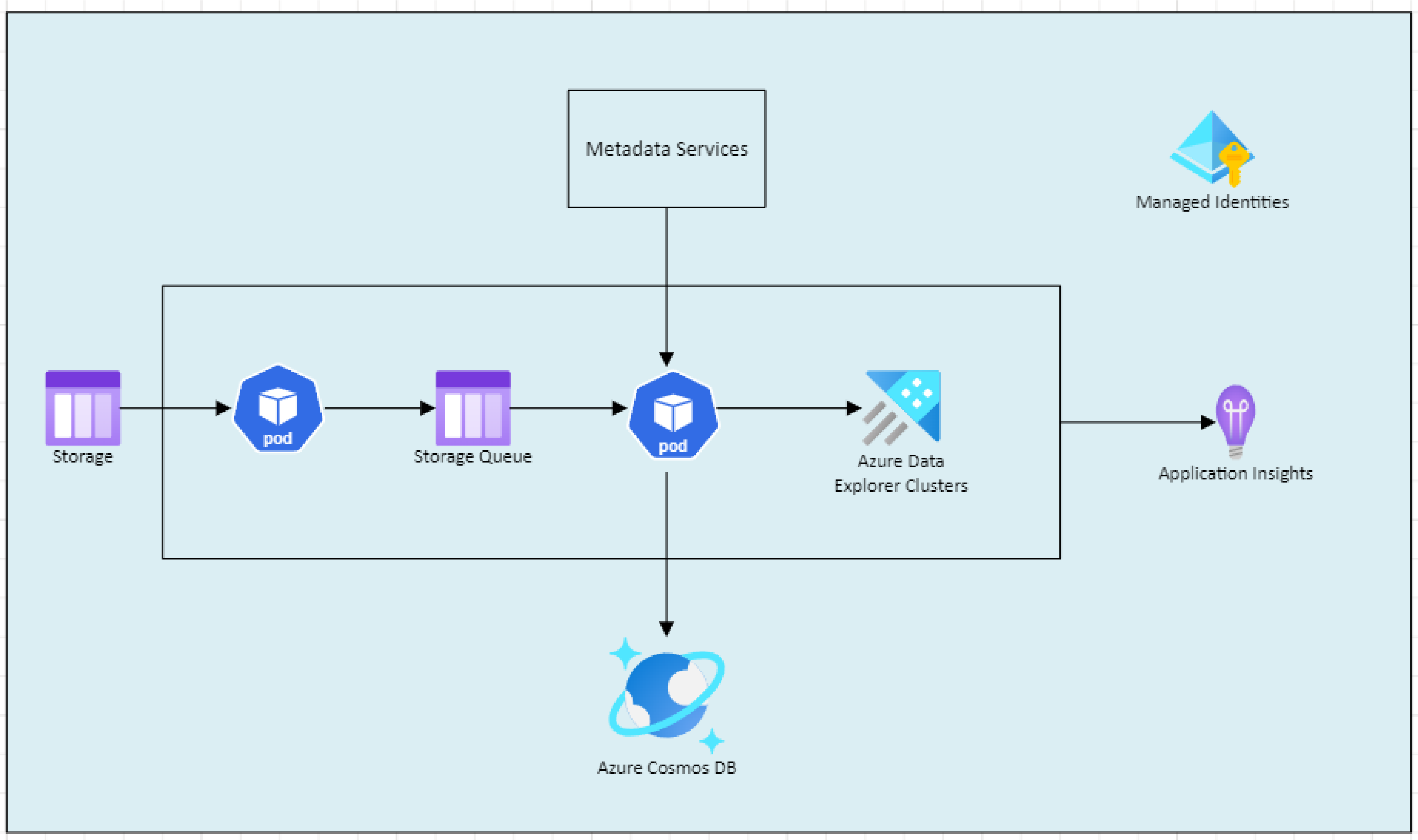
Aggregator Deep Dive



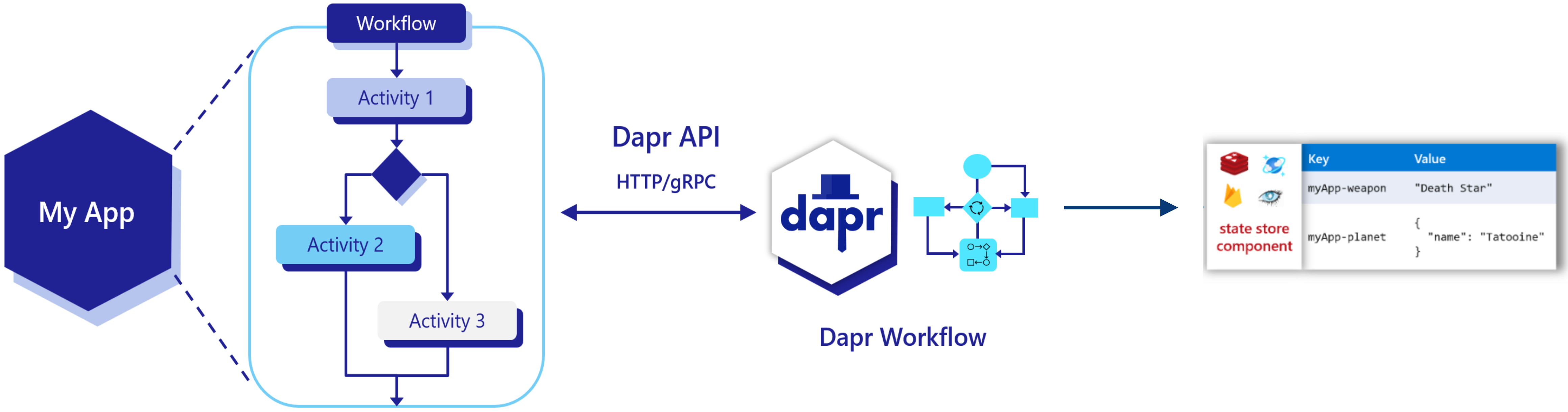
What we were looking for

- Out of box state management feature
- Support with Kubernetes.

Azure Architecture



Dapr Workflow



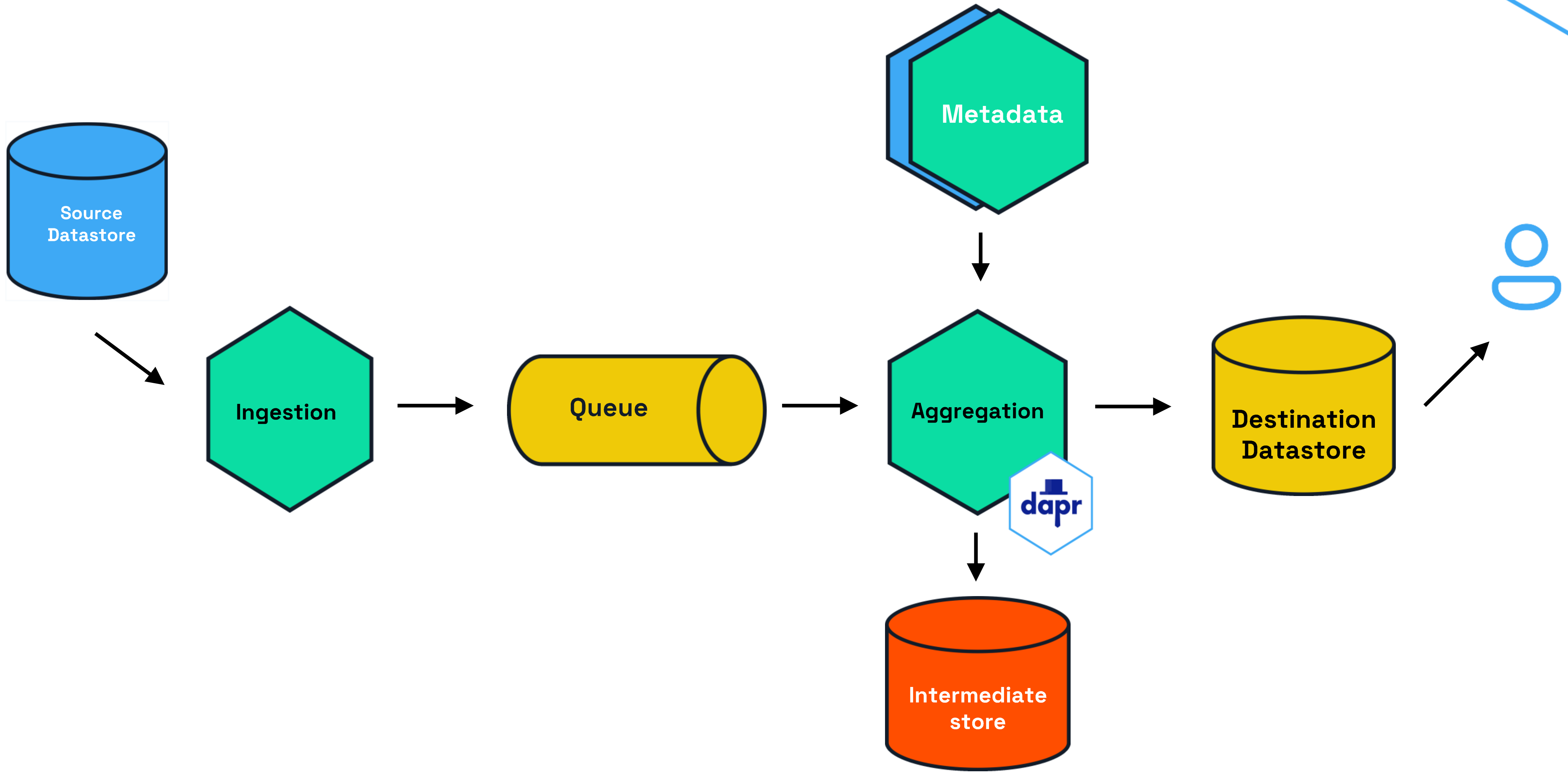
Key	Value
myApp-weapon	"Death Star"
myApp-planet	{ "name": "Tatooine" }

Source: https://docs.dapr.io/developing-applications/building-blocks/state-management/state-management-overview/

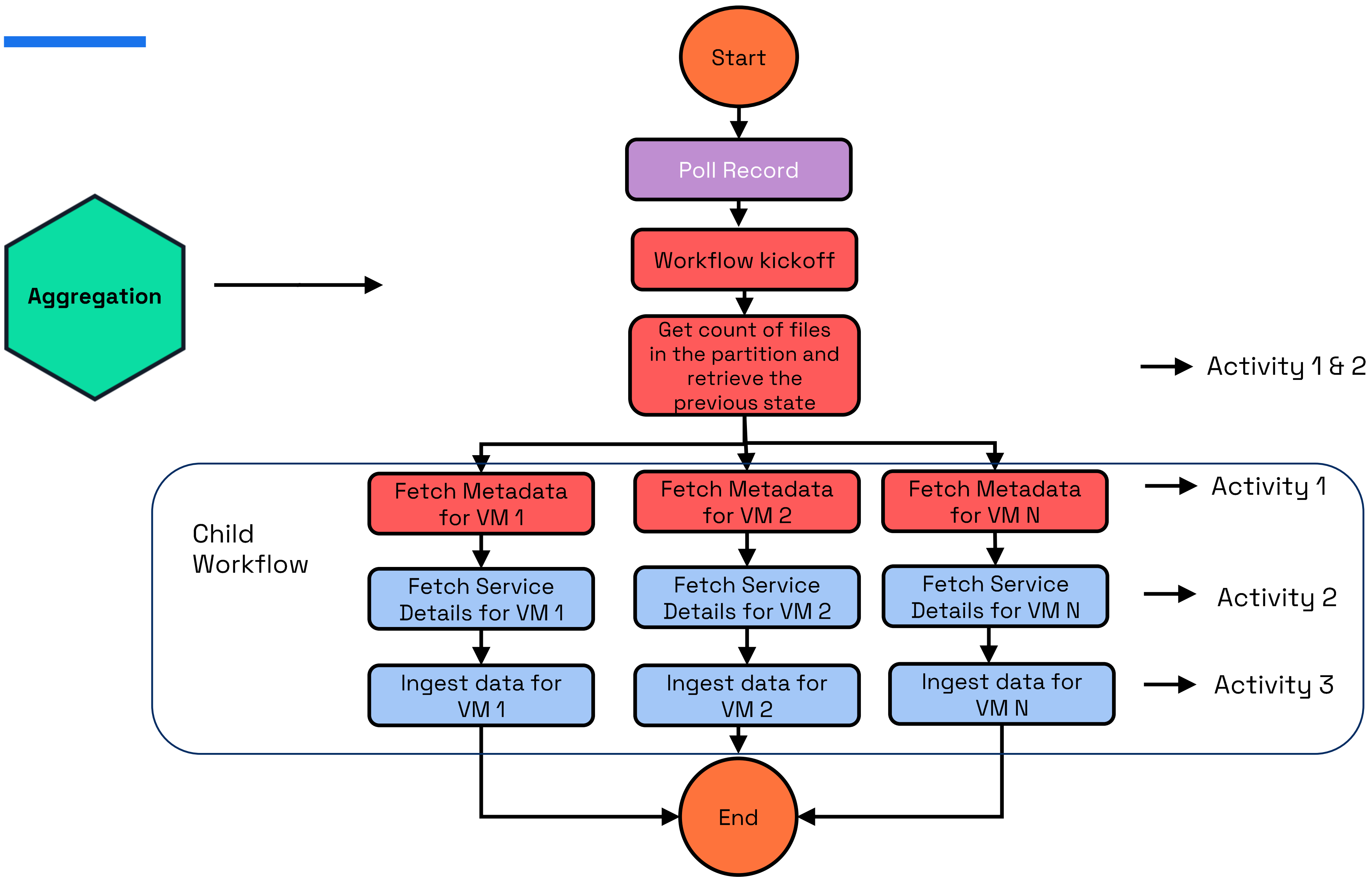
Constructs

- Workflow and Activities
- Event Sourcing
- “await”
- Patterns
    - Fan out pattern
    - Task chaining

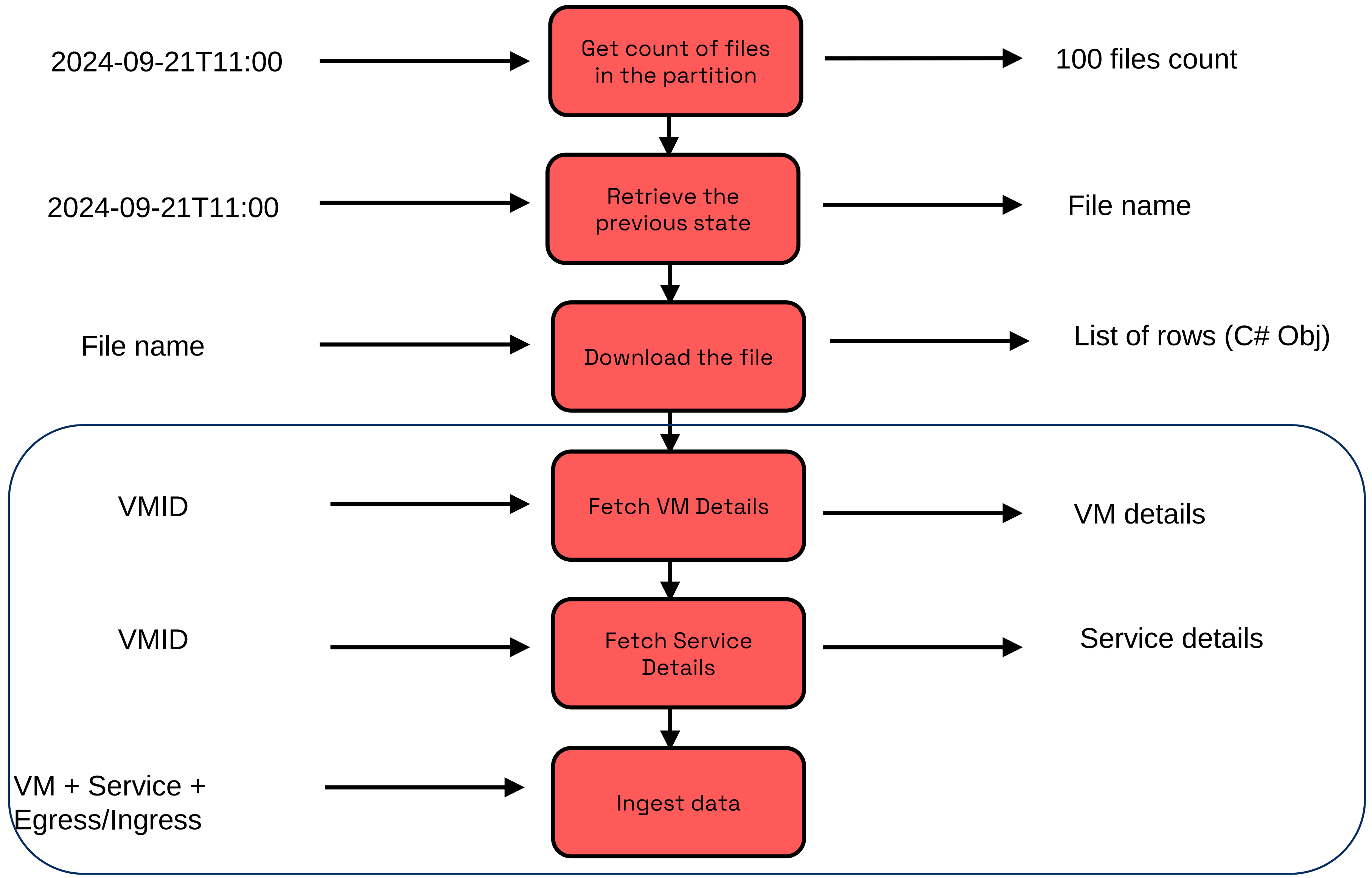
After Dapr

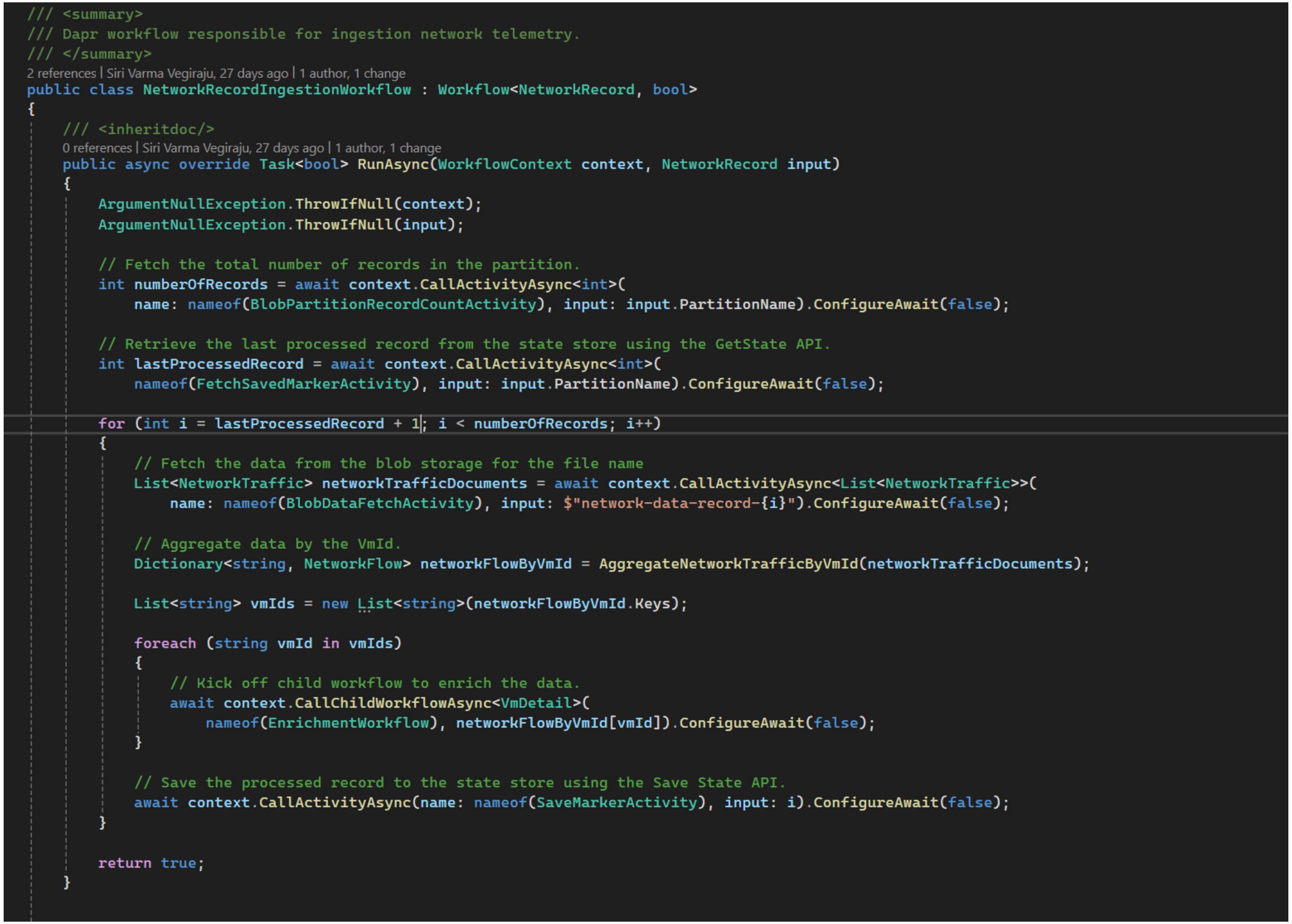

Metadata
Source Datastore
Ingestion
Queue
Aggregation
dapr
Destination Datastore
Intermediate store

Aggregator Deep Dive



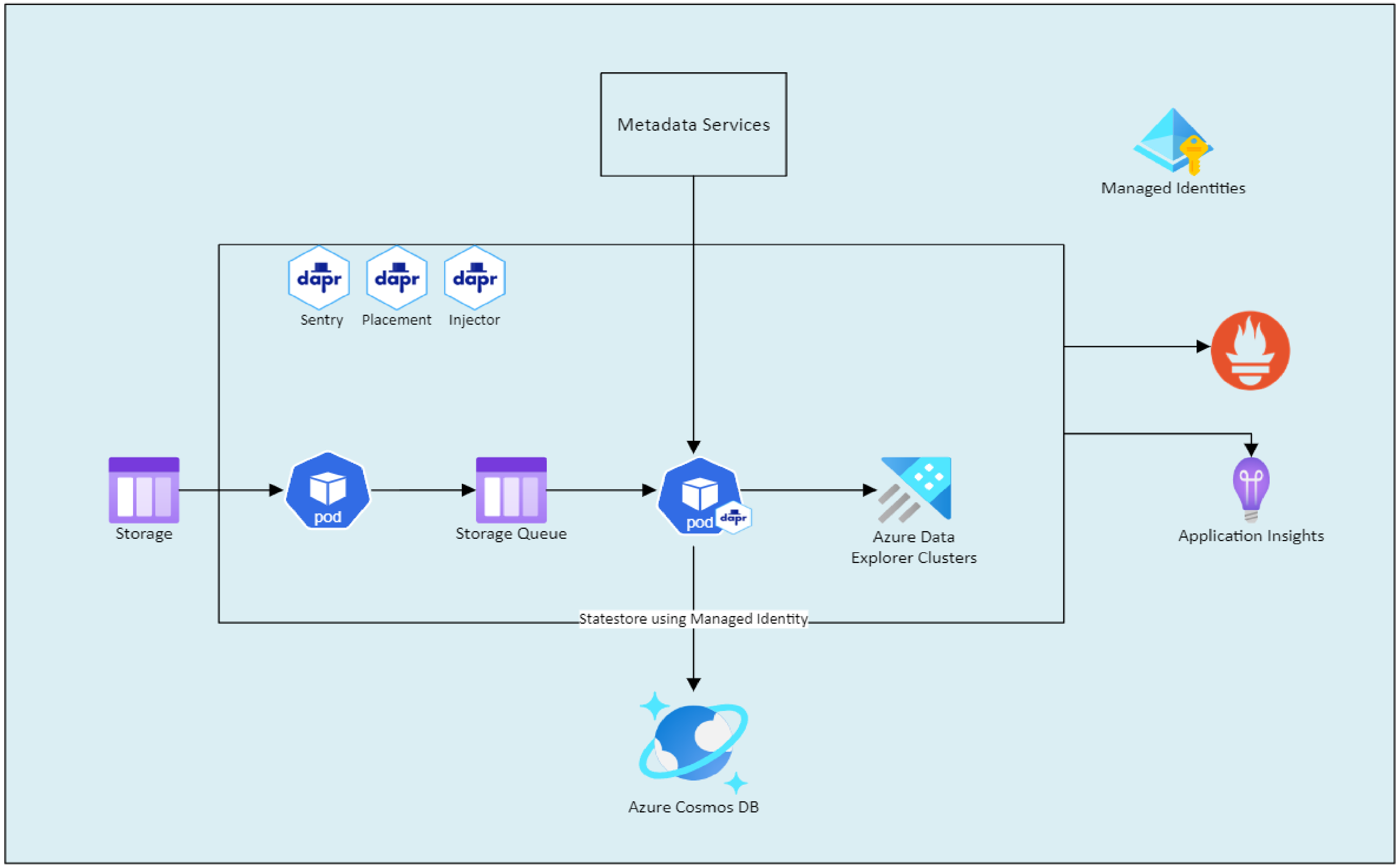
I/P and O/P to Activity





```
/// <summary>
/// Dapr workflow responsible for ingestion network telemetry.
/// </summary>
2 references | Siri Varma Vegiraju, 27 days ago | 1 author, 1 change
public class NetworkRecordIngestionWorkflow : Workflow<NetworkRecord, bool>
{
    /// <inheritdoc/>
    0 references | Siri Varma Vegiraju, 27 days ago | 1 author, 1 change
    public async override Task<bool> RunAsync(WorkflowContext context, NetworkRecord input)
    {
        ArgumentNullException.ThrowIfNull(context);
        ArgumentNullException.ThrowIfNull(input);

        // Fetch the total number of records in the partition.
        int numberOfRecords = await context.CallActivityAsync<int>(
            name: nameof(BlobPartitionRecordCountActivity), input: input.PartitionName).ConfigureAwait(false);

        // Retrieve the last processed record from the state store using the GetState API.
        int lastProcessedRecord = await context.CallActivityAsync<int>(
            nameof(FetchSavedMarkerActivity), input: input.PartitionName).ConfigureAwait(false);

        for (int i = lastProcessedRecord + 1; i < numberOfRecords; i++)
        {
            // Fetch the data from the blob storage for the file name
            List<NetworkTraffic> networkTrafficDocuments = await context.CallActivityAsync<List<NetworkTraffic>>(
                name: nameof(BlobDataFetchActivity), input: $"network-data-record-{i}").ConfigureAwait(false);

            // Aggregate data by the VmId.
            Dictionary<string, NetworkFlow> networkFlowByVmId = AggregateNetworkTrafficByVmId(networkTrafficDocuments);

            List<string> vmIds = new List<string>(networkFlowByVmId.Keys);

            foreach (string vmId in vmIds)
            {
                // Kick off child workflow to enrich the data.
                await context.CallChildWorkflowAsync<VmDetail>(
                    nameof(EnrichmentWorkflow), networkFlowByVmId[vmId]).ConfigureAwait(false);
            }

            // Save the processed record to the state store using the Save State API.
            await context.CallActivityAsync(name: nameof(SaveMarkerActivity), input: i).ConfigureAwait(false);
        }

        return true;
    }
```

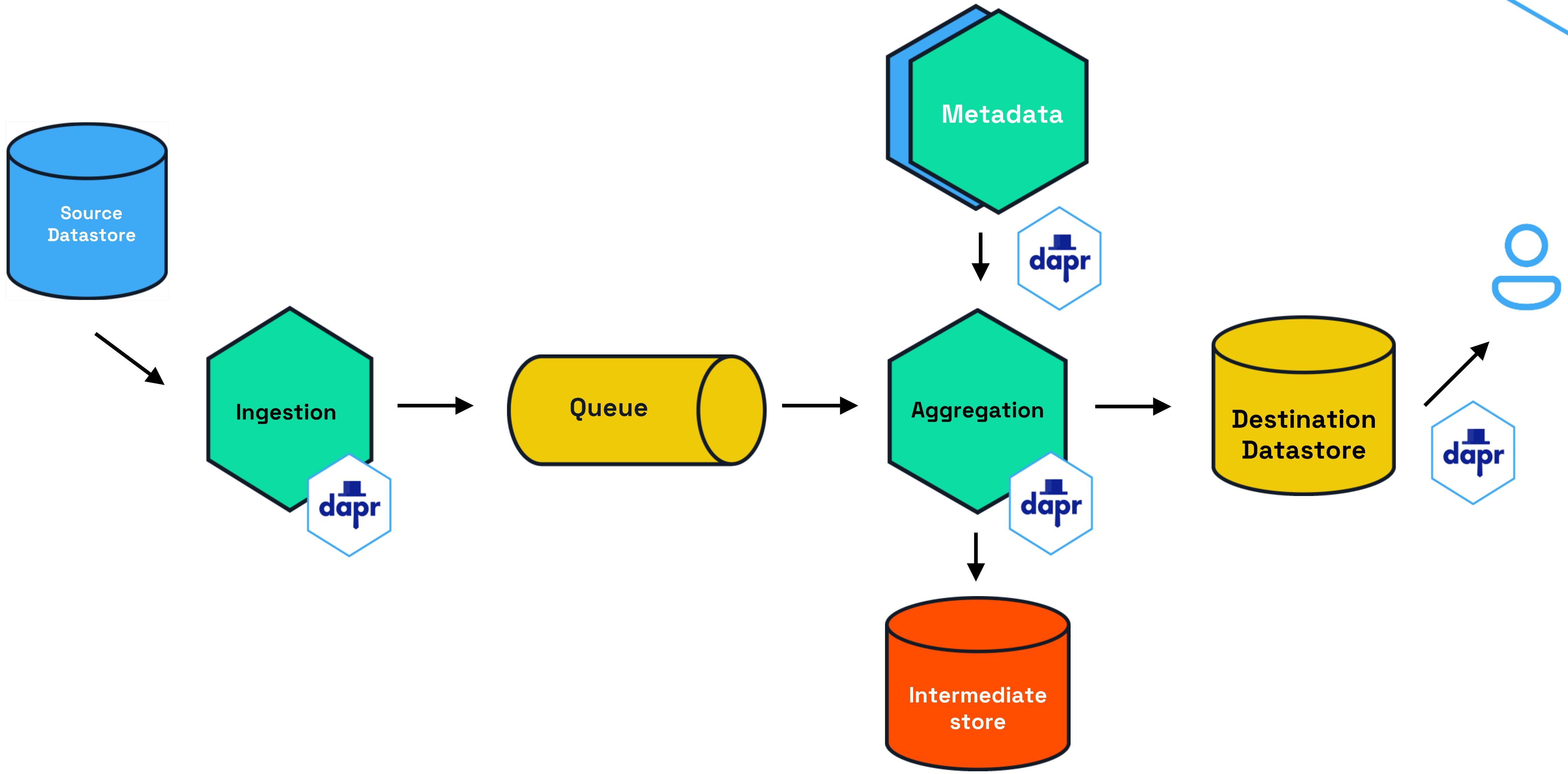
Azure Architecture



Learnings

- Smaller Workflows are better
- Understand the State store limitations
- Breakdown Workflows and Activities effectively.

North Star


Source Datastore
Ingestion
dapr
Queue
Metadata
dapr
Aggregation
dapr
Intermediate store
Destination Datastore
dapr

Thank you