AZURE DAY

Azure Container App ♡ Dapr Workflow

Andrea Tosato @ Beta 80

Microsoft msc TECHNOLOGY NET CODE Azure Meetup TORINO TORINO.NET

AZURE DAY

Platinum Sponsor

Microsoft

msc TECHNOLOGY

Technical Sponsor

.NET CODE

Azure Meetup TORINO

TORINO .NET

stickermule

Packt

JET BRAINS

ATosato86

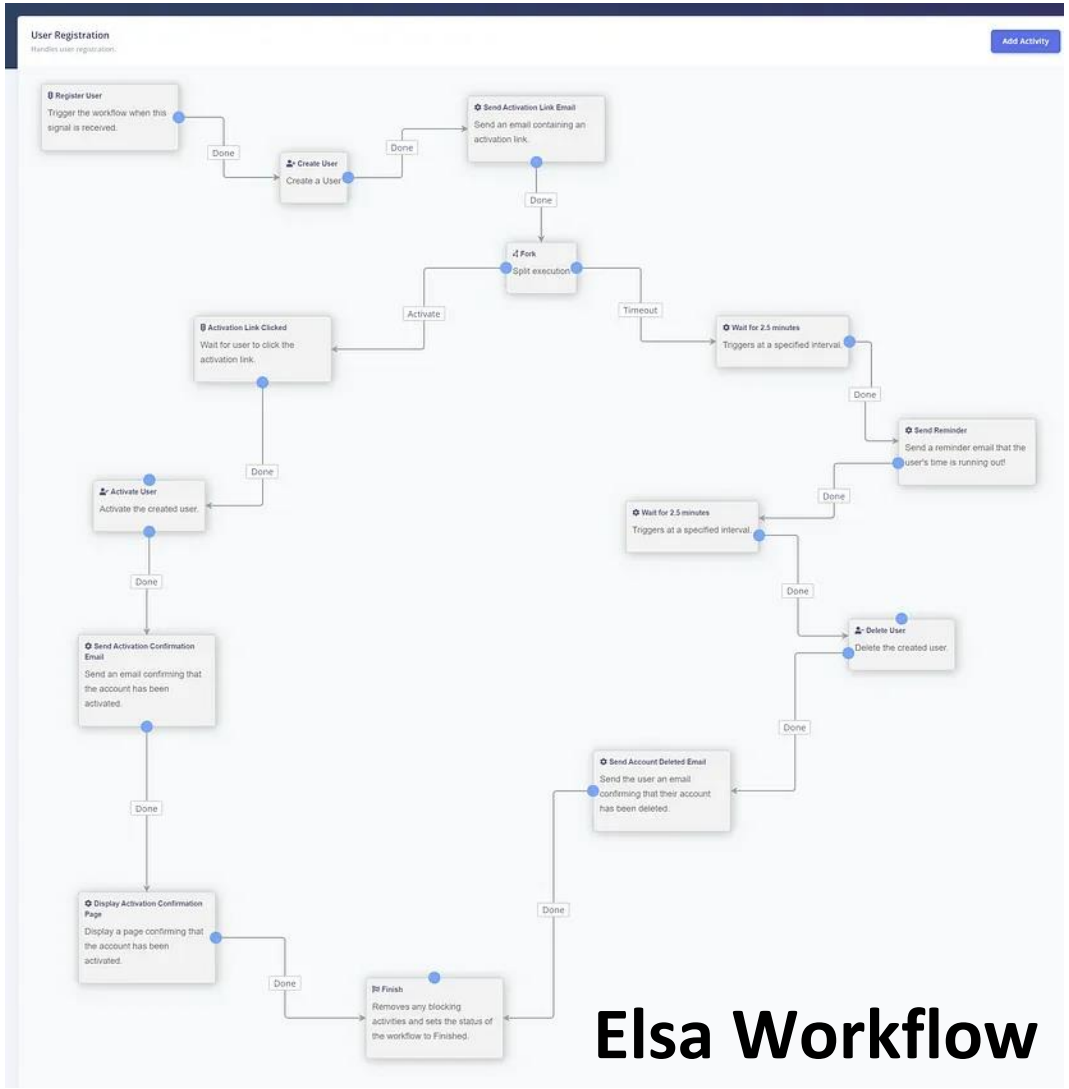
andreatosato

andreatosato

BETA 80 **GROUP**

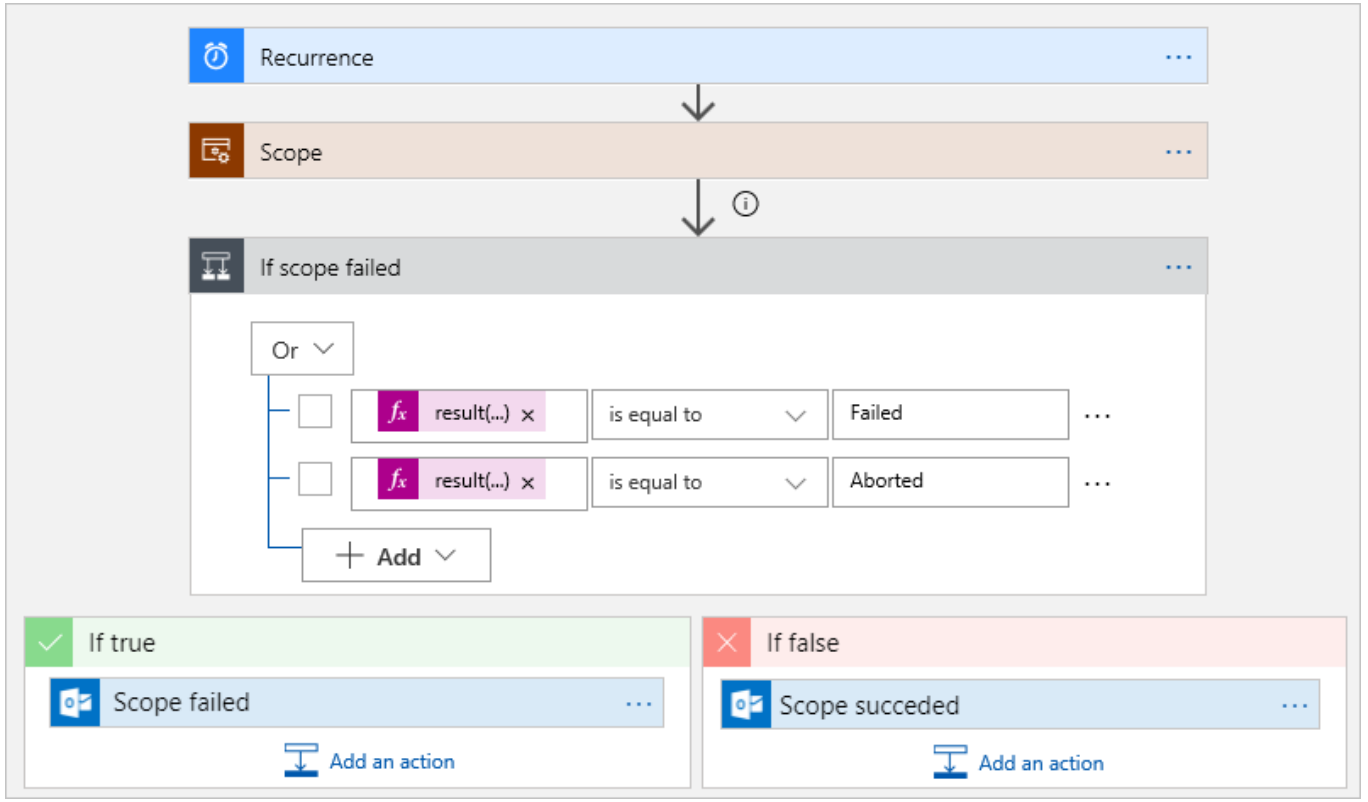
Microsoft® Most Valuable Professional

What kind of workflow?



Elsa Workflow

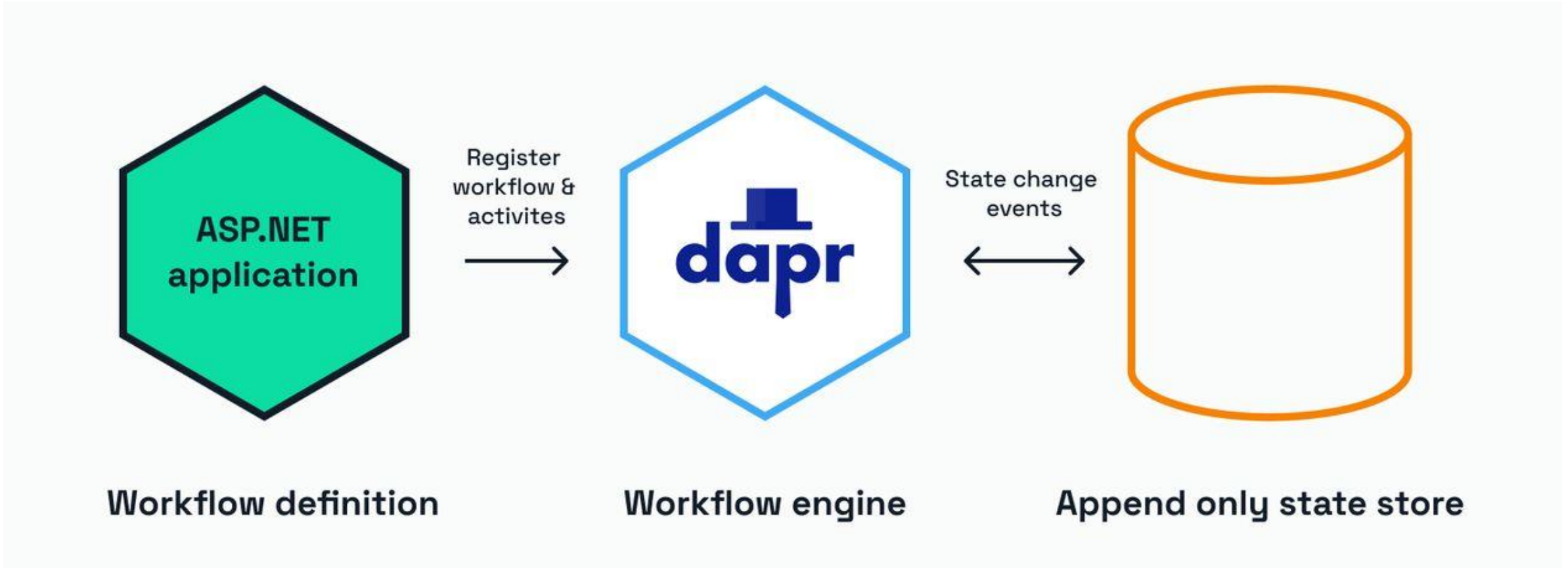
Azure Logic Apps



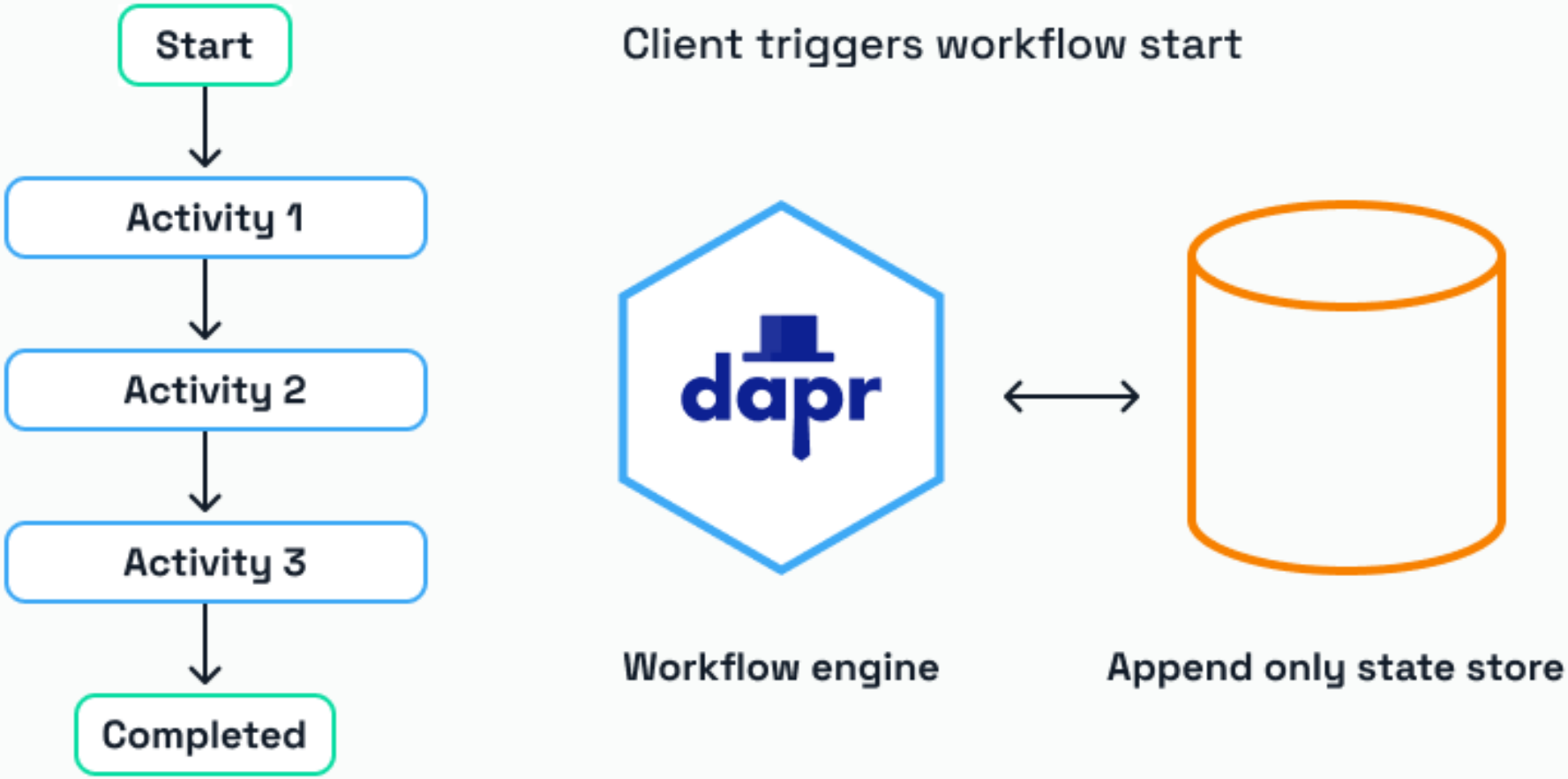
Have you lost interest?

```
public class ChainingWorkflow : Workflow<string, string>
{
    public override async Task<string> RunAsync(WorkflowContext context, string input)
    {
        var message1 =  await context.CallActivityAsync<string>(
            nameof(CreateGreetingActivity),
            input);

        var message2 = await context.CallActivityAsync<string>(
            nameof(CreateGreetingActivity),
            message1);

        var message3 = await context.CallActivityAsync<string>(
            nameof(CreateGreetingActivity),
            message2);

        return message3;
    }
}
```

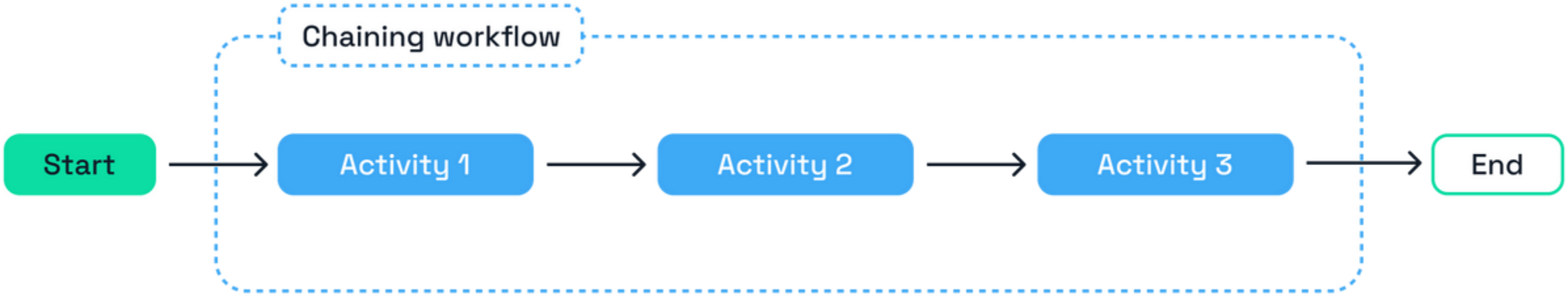
How does it works?



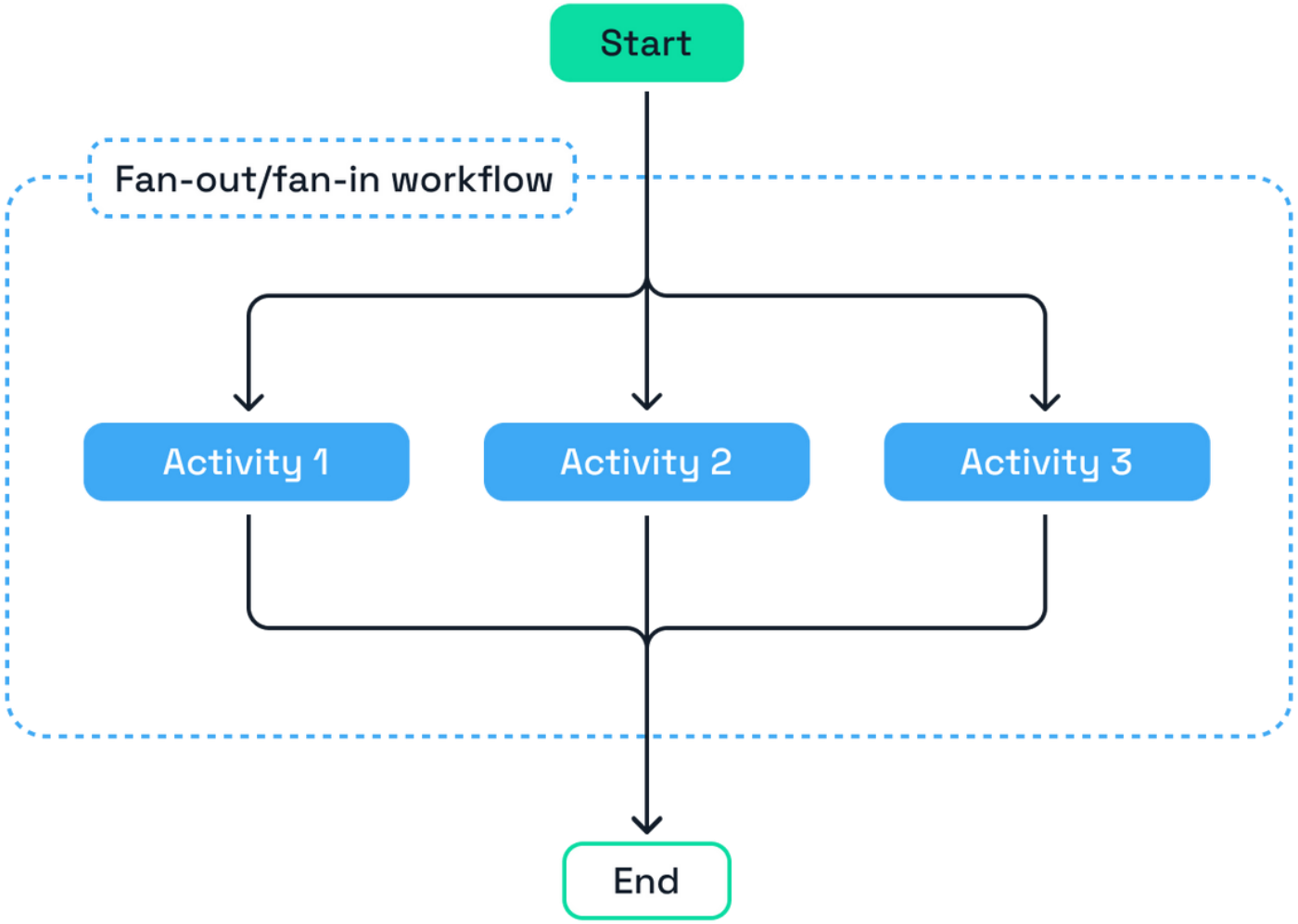
How does it works?



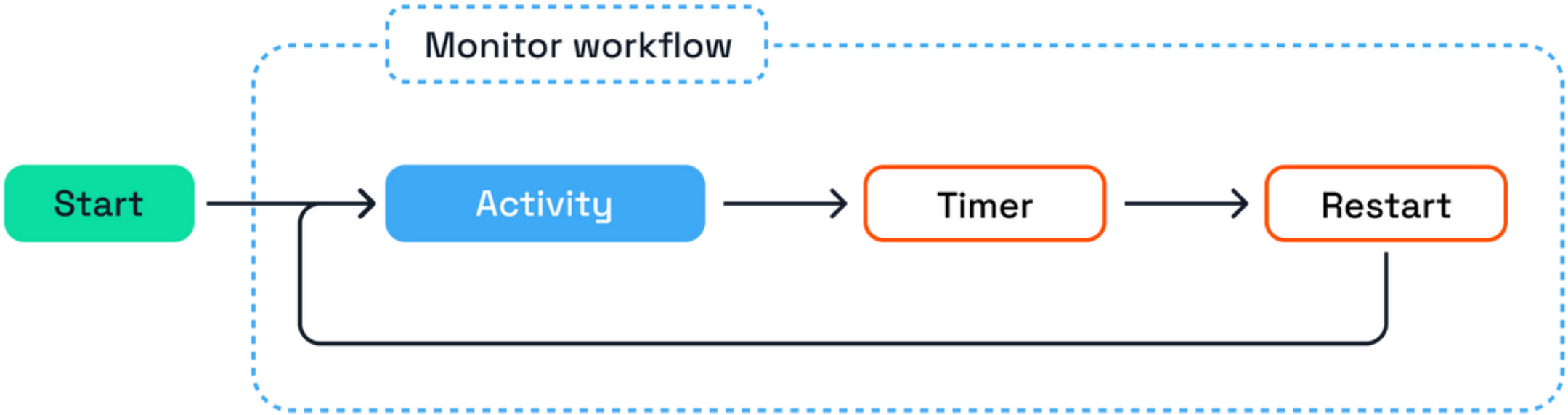
Chaining pattern



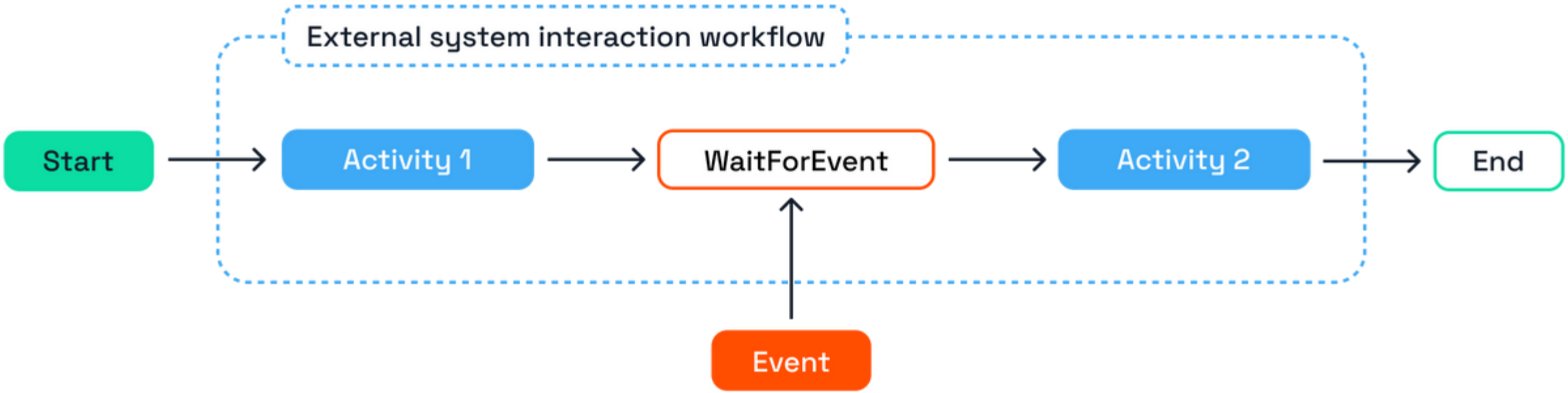
Fan-out/Fan-in



Monitor



External system interaction

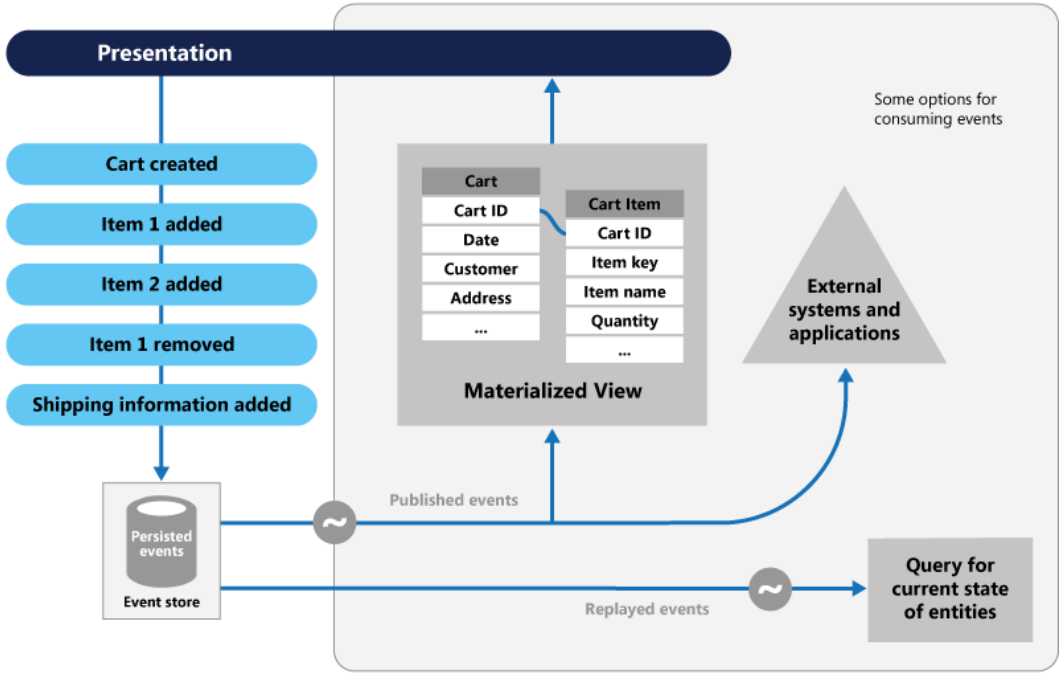


HTTP calls to manage a workflow

- **Started** or terminated through a POST request
- **Triggered** to deliver a named event through a POST request
- **Paused** and then resumed through a POST request
- **Purged** from your state store through a POST request
- **Queried** for workflow **status** through a GET request

Workflow replay - Workflow determinism

Instead of storing the current state of a workflow as a snapshot, the workflow engine manages an **append-only log of history events** that describe the various steps that a workflow has taken. When using the workflow SDK, these history events are stored automatically whenever the workflow “awaits” for the result of a scheduled task.



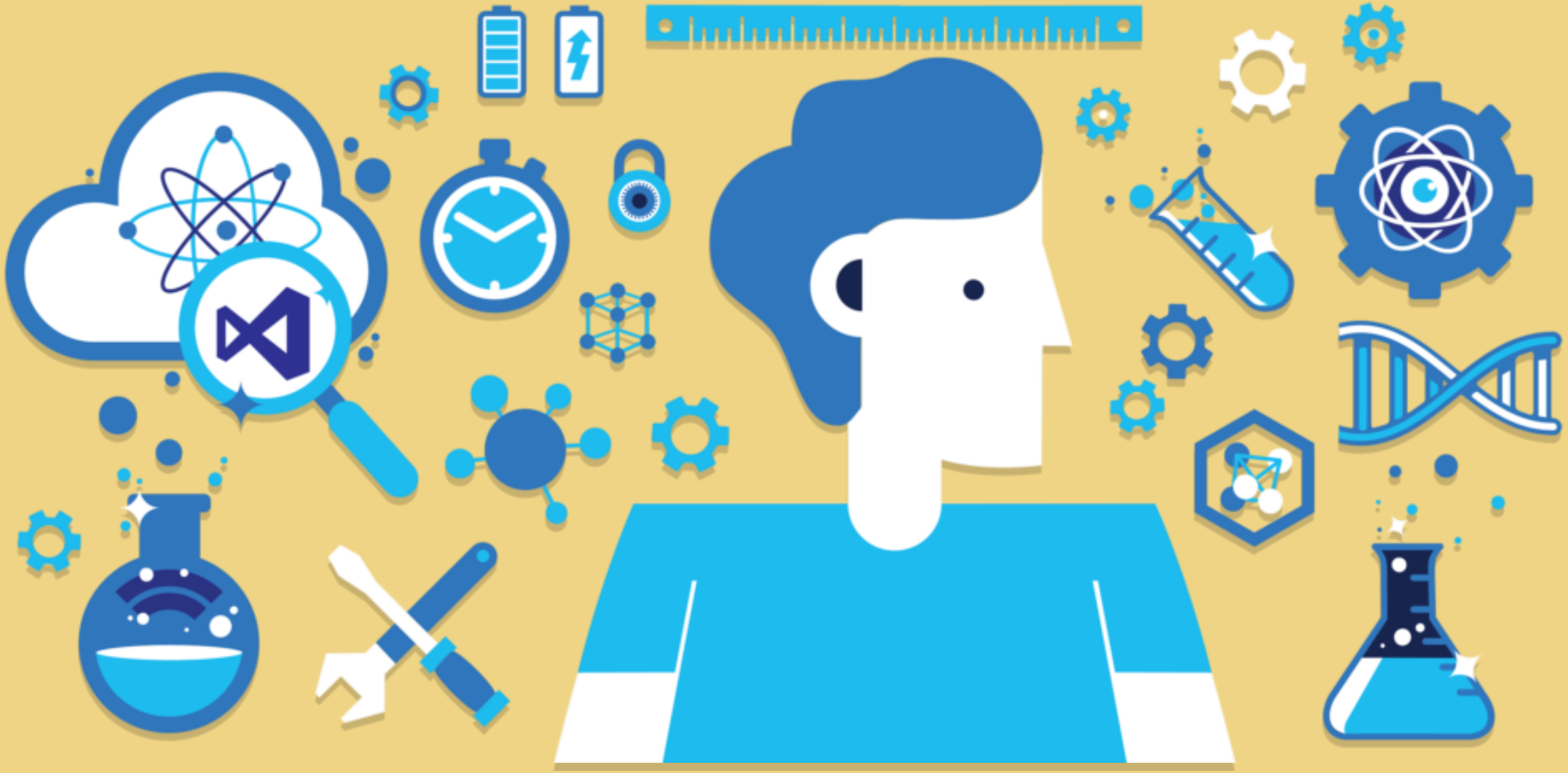

```
DateTime currentTime = context.CurrentUtcDateTime;
Guid newIdentifier = context.NewGuid();
string randomString = await context.CallActivityAsync<string>("GetRandomString");
```

Limitations

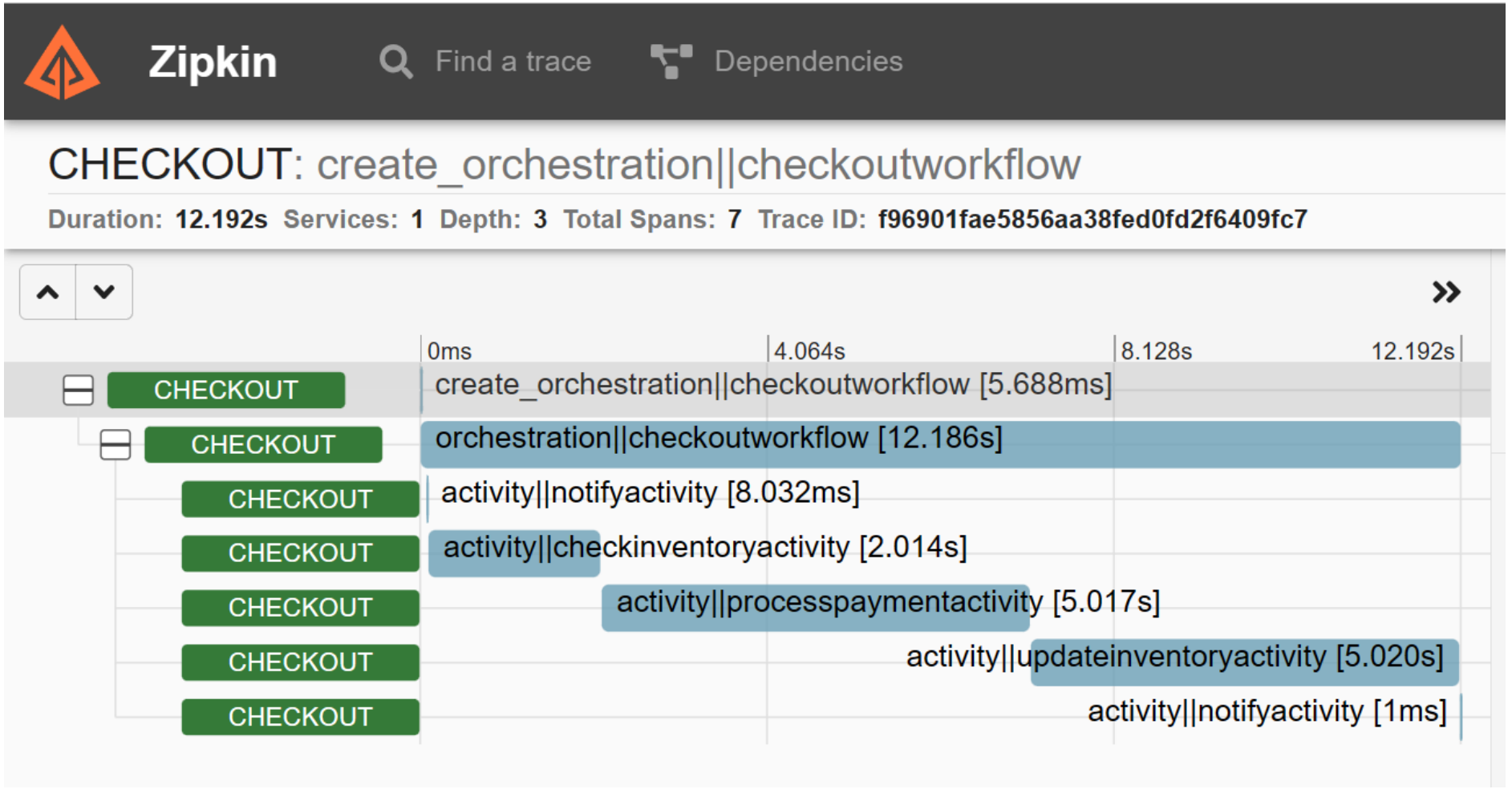
- **State stores:** For the 1.12.0 beta release of Dapr Workflow, you're not able to use NoSQL databases. Only SQL databases are supported in the latest release.

- **Application instances:** For the 1.12.0 beta release of Dapr Workflow, only a maximum of 2 application instances is supported.

AZURE DAY



Telemetry


Zipkin
Find a trace
Dependencies
CHECKOUT: create_orchestration||checkoutworkflow
Duration: 12.192s Services: 1 Depth: 3 Total Spans: 7 Trace ID: f96901fae5856aa38fed0fd2f6409fc7
0ms
4.064s
8.128s
12.192s
CHECKOUT
create_orchestration||checkoutworkflow [5.688ms]
CHECKOUT
orchestration||checkoutworkflow [12.186s]
CHECKOUT
activity||notifyactivity [8.032ms]
CHECKOUT
activity||checkinventoryactivity [2.014s]
CHECKOUT
activity||processpaymentactivity [5.017s]
CHECKOUT
activity||updateinventoryactivity [5.020s]
CHECKOUT
activity||notifyactivity [1ms]

- https://www.diagrid.io/blog/in-depth-guide-to-dapr-workflow-patterns
- https://www.diagrid.io/blog/authoring-dapr-workflows-in-dotnet
- https://docs.dapr.io/operations/configuration/configuration-overview/
- https://docs.dapr.io/reference/api/workflow_api/

Thank You!!!

AZURE DAY

Platinum Sponsor

Microsoft

msc TECHNOLOGY

Technical Sponsor

.NET CODE

Azure Meetup TORINO

TORINO .NET

stickermule

Packt

JET BRAINS