**Dapr Service Invocation: Ease of Communication between Microservices**

Microservice architectures are an increasingly popular approach to modern application development. This approach involves breaking applications into small, independent services and running each service as a separate process. However, this requires communication between microservices. Dapr (Distributed Application Runtime) is a framework developed to solve this problem and in this article, we will learn more about Dapr Service Invocation.

**What is it?**

Service Invocation is a feature offered by Dapr to facilitate communication between microservices. This allows one service to call another service or retrieve data. Service Invocation can be used client-side or server-side and they don't have to be written in the same language. This makes communication between services running in different languages and platforms fast and reliable.

**How does it work?**

Dapr Service Invocation enables communication between two services in a simple and secure way. The basic steps in this communication are as follows:

**Registering the Service:** Each service is registered with Dapr. After registration, the service receives an ID (for example, ServiceA).

**Request Sending**: When a service wants to call another service, it specifies the ID of the service it is calling (e.g. ServiceB) and the name of the destination method.

**Request Processing:** Dapr receives the request, determines the target service and method, and performs the operation by forwarding the request.

**Result Return:** The target service returns the result when it has completed the operation.

**Result Retrieval:** The calling service receives the result and completes the operation.

A diagram of a data processing process

Description automatically generated

**Advantages**

Dapr Service Invocation provides many advantages in microservice architectures:

**Language and Platform Independence:** Dapr facilitates communication between services written in different languages and platforms.

**Monitoring and Logging:** Dapr offers the ability to monitor communication traffic and log errors.

**Timing and Retry:** Manages timing and automatic retry during communication, making the application more reliable.

**Security:** Dapr offers various security measures to secure communication.

**Code Repetition Avoidance:** Reduces code repetition while facilitating communication between services.

**Example:**

A diagram of a company

Description automatically generated

1. The NodeJs application has an application ID called nodeapp. The Pyhton application finds this application when it requests a POST to the http://localhost:3500/v1.0/invoke/nodeapp/method/neworder method.

2. Dapr discovers the location of the nodejs application through its name resolver component.

3. Dapr forwards the request to this location.

4. The dapr sidecar of the node'js application receives and processes the request. In the meantime, it logs this request.

5. The NodeJs application forwards the response to its sidecar.

6. Sends it to the Python application's sidecar.

7. The Python application receives the response.

**Conclusion**

Dapr Service Invocation makes the communication of microservices simple and reliable. This is a great advantage in modern application development projects and makes it easier for us to communicate effectively between services running in different languages and platforms.

**Kod Örneği:**

Let's create 2 microservices as below.

A screenshot of a computer

Description automatically generated

We define an endpoint for our Product microservice and print to the console

A screenshot of a computer

Description automatically generated

We create our Simulator microservice that will call the products endpoint we defined in our Product microservice as follows.

A screenshot of a computer

Description automatically generated

As above, we make a call to the products endpoint of the product microservice and print it to the console.

Now let's see how to run our two microservices with Dapr.

Let's run the following command in developer powershell for Product microservice.

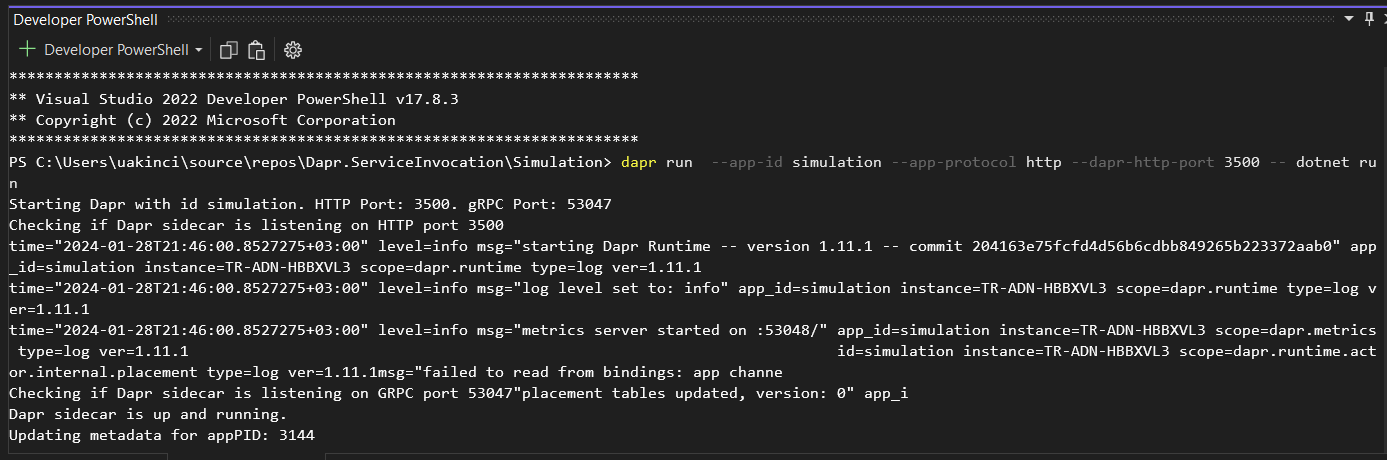
**dapr run --app-port 7001 --app-id product-service --app-protocol http --dapr-http-port 3501 -- dotnet run**

A screenshot of a computer program

Description automatically generated

Next, let's run our Simulator microservice, which will make requests to our product microservice.

**dapr run --app-id simulator --app-protocol http --dapr-http-port 3500 -- dotnet run**



The Simulator microservice is up and running and has started sending requests to the product microservice.

A computer screen shot of a black screen

Description automatically generated

