Making Microservices More Resilient

Gill Cleeren
CTO XPIRIT BELGIUM

@gillcleeren   www.snowball.be

Overview

Understanding the possible issues with microservice communication

Revisiting the IHttpClientFactory

Improving resiliency using Polly

Understanding the Possible Issues with Microservice Communication

We know one thing for sure…

Errors will happen in a distributed system.

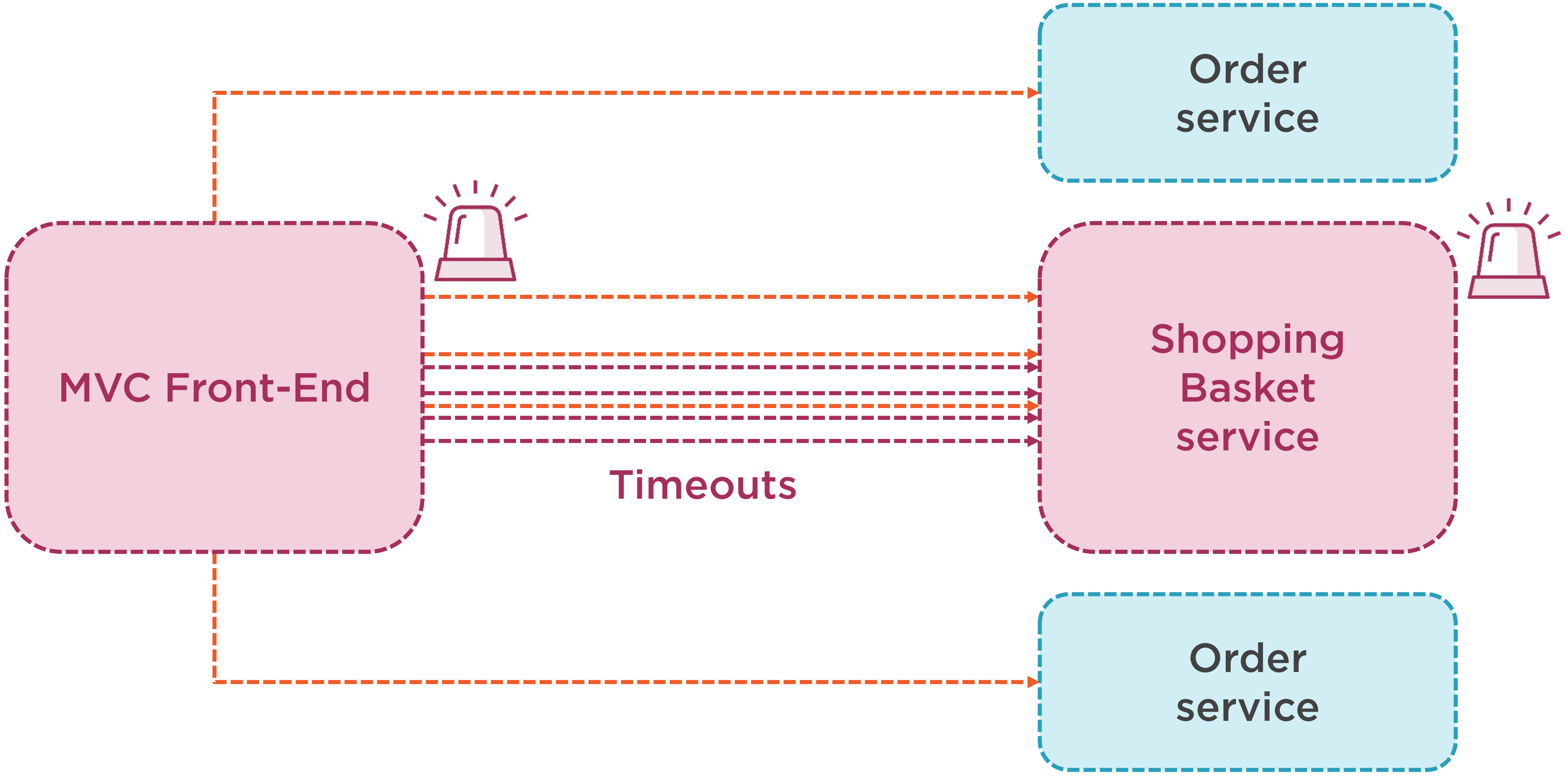
Errors that occur will be harder and can cause bigger problems.

Failures of services will happen

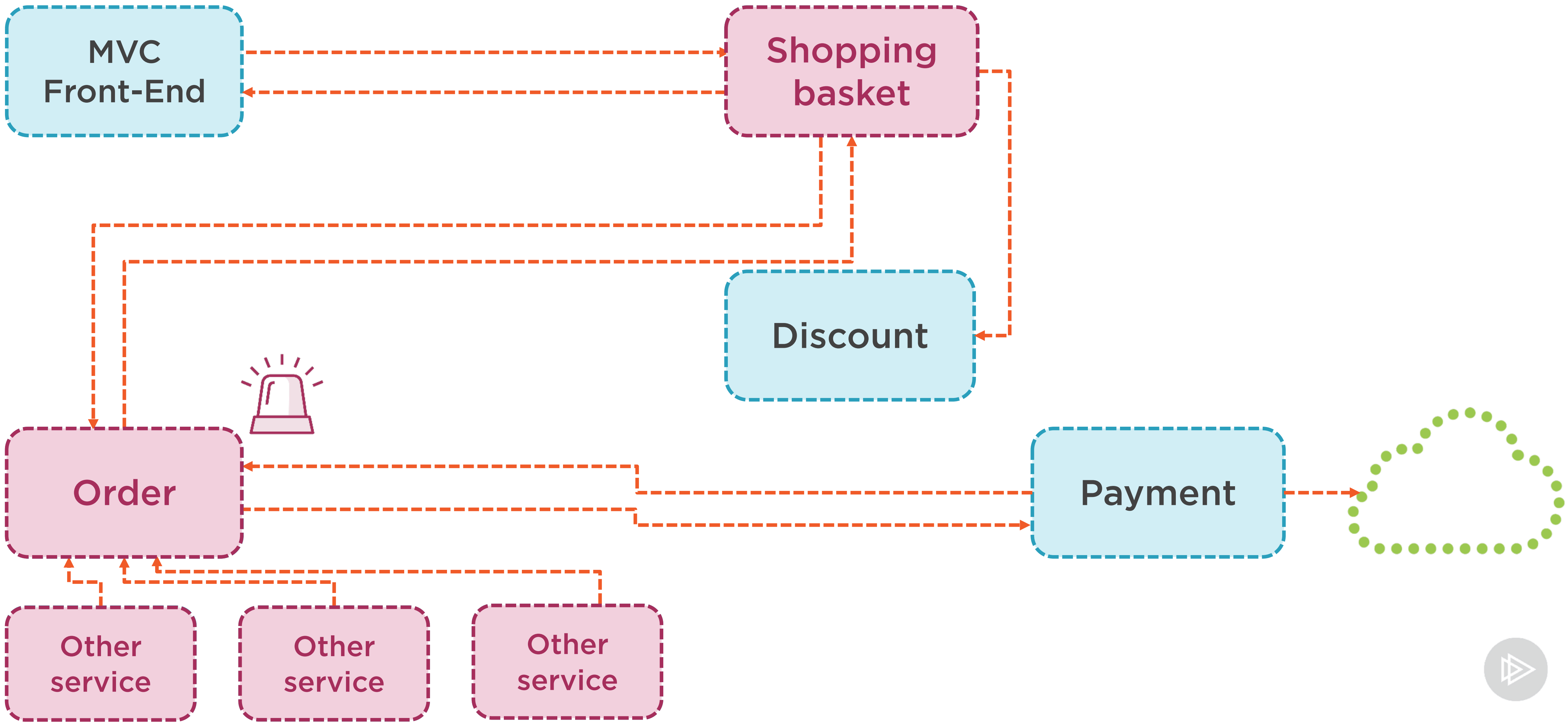
Services might not exist briefly

System must handle errors gracefully

The Effect of a Service Failure



The Effect of a Service Failure



Common Issues

Service not available

Service overload after unavailable

Improving Service Resiliency

- Asynchronous communication
- Retry with exponential backoff
- Circuit breaker

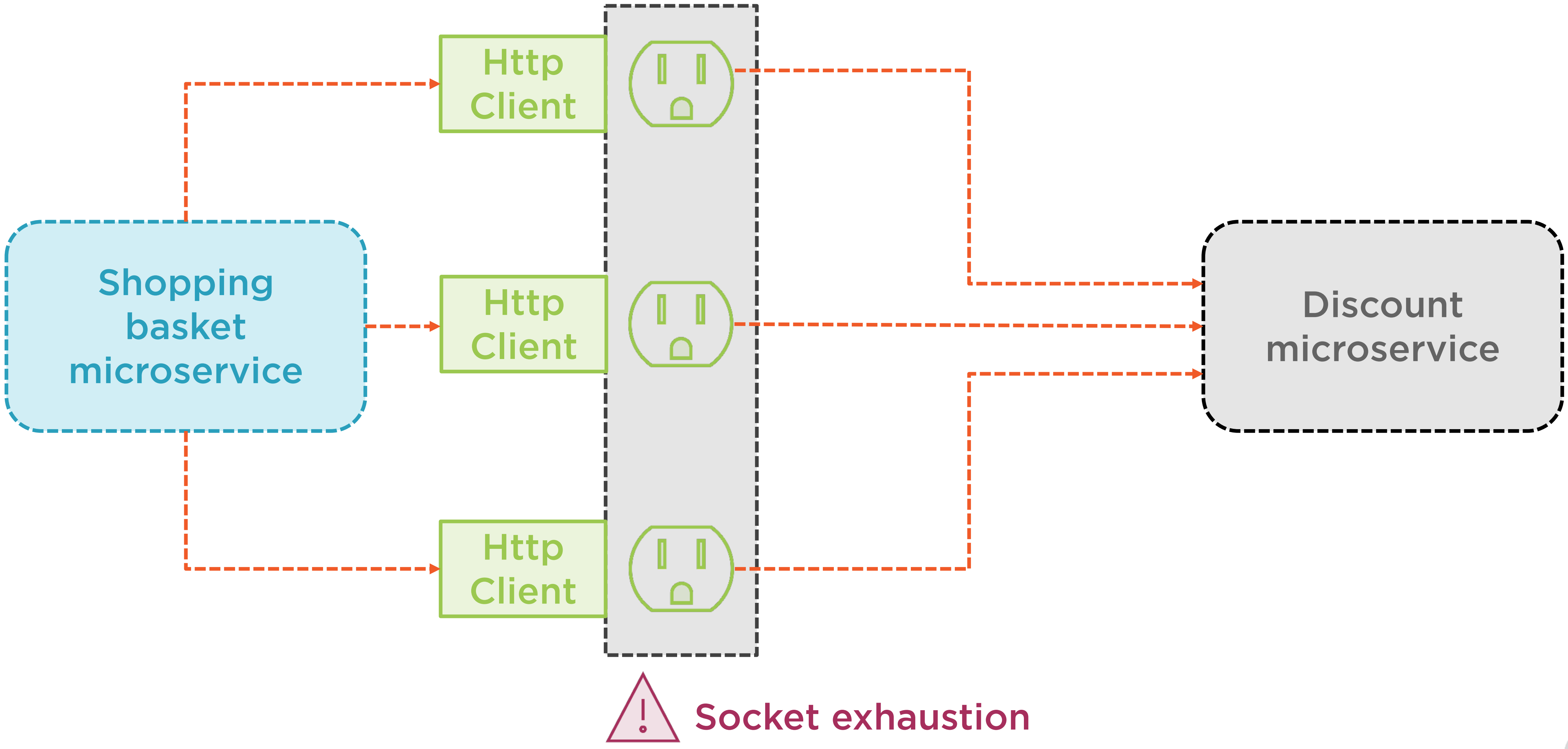
Revisiting the IHttpClientFactory

Using HttpClient

Simple. Perhaps too simple.

Can lead to socket exhaustion, blowing up your app or service.

Using HttpClient



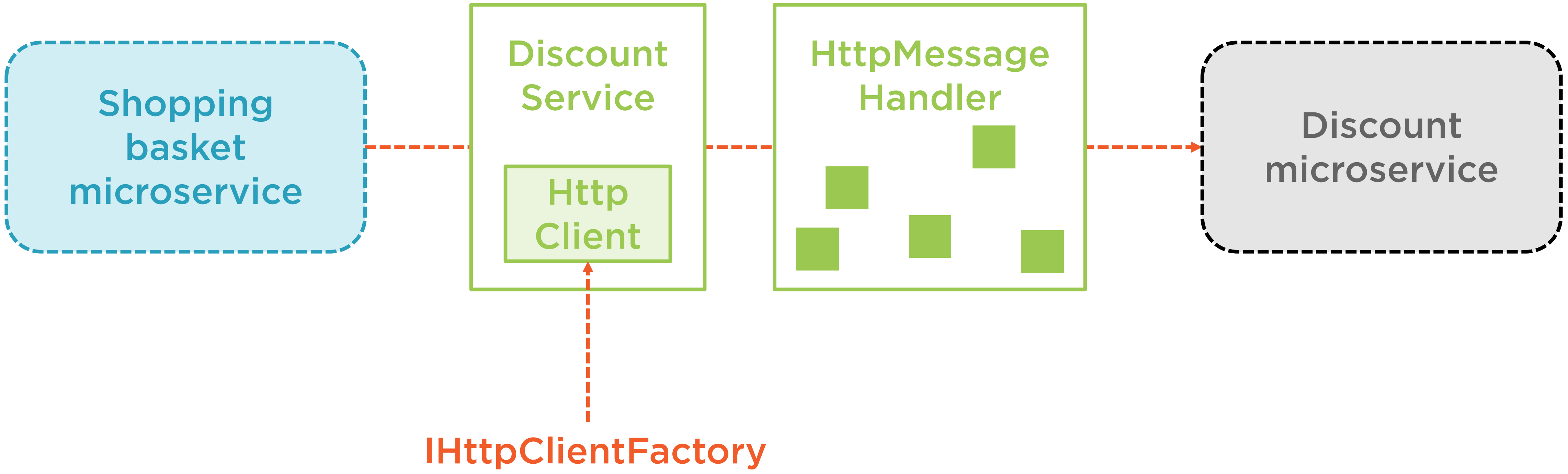
Using IHttpClientFactory

Managed HttpClient instances

Dependency injection

Pooling of HttpMessageHandler instances

Using IHttpClientFactory



```
services.AddHttpClient<IDiscountService, DiscountService>
    (c => c.BaseAddress = new
        Uri(Configuration["ApiConfigs:Discount:Uri"]));
```

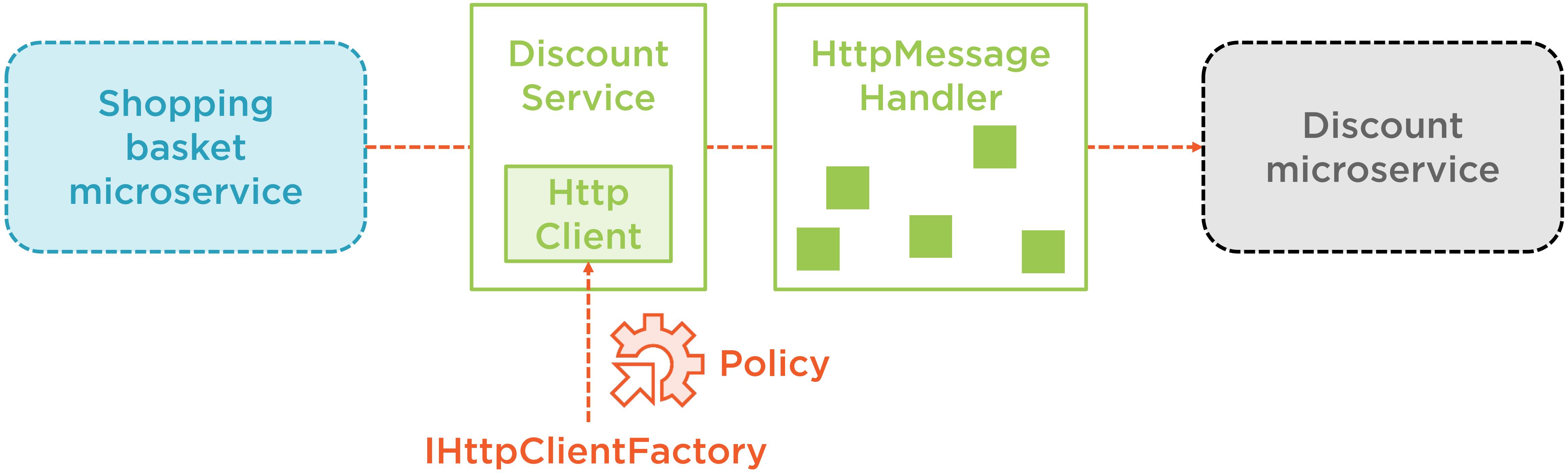
Registering with the DI Container

Using IHttpClientFactory

```
public class DiscountService: IDiscountService
{
    private readonly HttpClient client;

    public DiscountService(HttpClient client)
    {
        this.client = client;
    }
}
```

Adding Policies



Demo

Using the IHttpClientFactory for increased service resiliency

Improving Resiliency Using Polly

Resilient Services

Microservices will encounter errors, even planned ones.

Transient failures will happen.

Adding Resiliency via Code

Polly

- Open source library

Define policies

- Retry, Timeout, Circuit breaker…

Integrated with IHttpClientFactory

The Retry Policy

Retry a certain request to another service

Wait times

User doesn't notice typically

The Circuit Breaker Policy

Service can be under stress or even unable to respond

Avoid causing more requests through pause

“Break” the Circuit before trying again

Demo

Adding a Retry Policy

Demo

Adding a Circuit Breaker Policy

Summary

Microservices will fail

Communication will suffer

Our code needs to cater for this

Improving resiliency through the IHttpClientFactory and Polly

Up next:

Accessing the microservices from the outside