STUDY4 為學習而生 Build School

使用 .NET 8 開發雲原生應用程式

Marcus @ .NET Conf

Hello!

Microsoft® Most Valuable Professional

I'm Marcus

- 後端打雜工
- #自我學習 #分享 #可觀測性

分享經驗

- COSCUP、MOPCON、DevopsDays
- .NET Conf、Will 保哥線上技術分享
- 微軟活動：打造化繁為簡的雲原生平台
- 2023 ITHome 鐵人賽 DevOps 組佳作

Fb : m@rcus 學習筆記

Agenda

- What is Cloud native Apps?
- .NET 8 in Cloud Native Apps
  - Observability , Resilience and Manageable
- Demo : .NET Cloud Native apps
- Takeaway

WARNING

本次分享旨在與聽眾分享本人對於 .NET 8 的個人觀點，本場不會談論任何技術細節，期待看到大量技術、工具實務細節的朋友們，可能會大失所望。為不耽誤您的青春，請趁其他教室關門前前往。

#新人（Newcomer） #開發者（Developers） #維運者（Operators）

.NET 8

- Performance
- .NET MAUI
- Cloud Native
- Artificial Intelligence
- ASP.NET Blazor Full Stack

Cloud Native Apps

雲原生應用程式

What is cloud-native application ?

Cloud native technologies empower organizations to build and run scalable applications in modern, dynamic environments such as public, private, and hybrid clouds. Containers, service meshes, microservices, immutable infrastructure, and declarative APIs exemplify this approach.

These techniques enable loosely coupled systems that are resilient, manageable, and observable. Combined with robust automation, they allow engineers to make high-impact changes frequently and predictably with minimal toil.

by CNCF

What is cloud-native application ?

雲端原生技術賦予組織在如公共雲、私有雲和混合雲等現代化、動態的環境中，建立和運行可擴展應用程式的能力。**容器(Containers)**、**服務網格(, service meshes)**、**微服務(microservices)**、**不可變基礎設施(immutable infrastructure)** 和 **宣告式 API(declarative APIs)** 是此方法的範例。

雲端原生技術使系統能夠鬆散耦合，從而實現具有 **彈性(resilient)**、**可管理(manageable)** 和 **可觀察(observable)** 的特點。結合強健的自動化，它們使工程師能夠頻繁且可預測地進行重大變更，同時將勞碌工作降至最低。

by CNCF + ChatGPT

.NET 8 如何協助 Developer

Observability

Resilience

.NET Aspire

Observability

可觀測性

The 3 pillars of Observability

Structured Log

Metrics

Traces

Detect

Seconds/SLI/KPIs

Metrics

Do I have a problem ?

Troubleshoot

Service dependencies

Traces

Where is the problem ?

Root Cause

Unlimited detail

Structured Log

What is causing the problem ?

OpenTelemetry：蒐集遙測數據的新標準

Open Source via CNCF

- 2nd Popular Project
- OpenTelemetry = OpenCensus + OpenTracing
- Support Logs, Metrics, Distributed tracing

Mission

- 無所不在的高品質、可攜式遙測
- 願景：有效的可觀測性世界

OpenTelemetry

Specifications

- OTel Specification
- OTel Protocol
- Open Agent Management Protocol
- OTel Semantic Conventions .

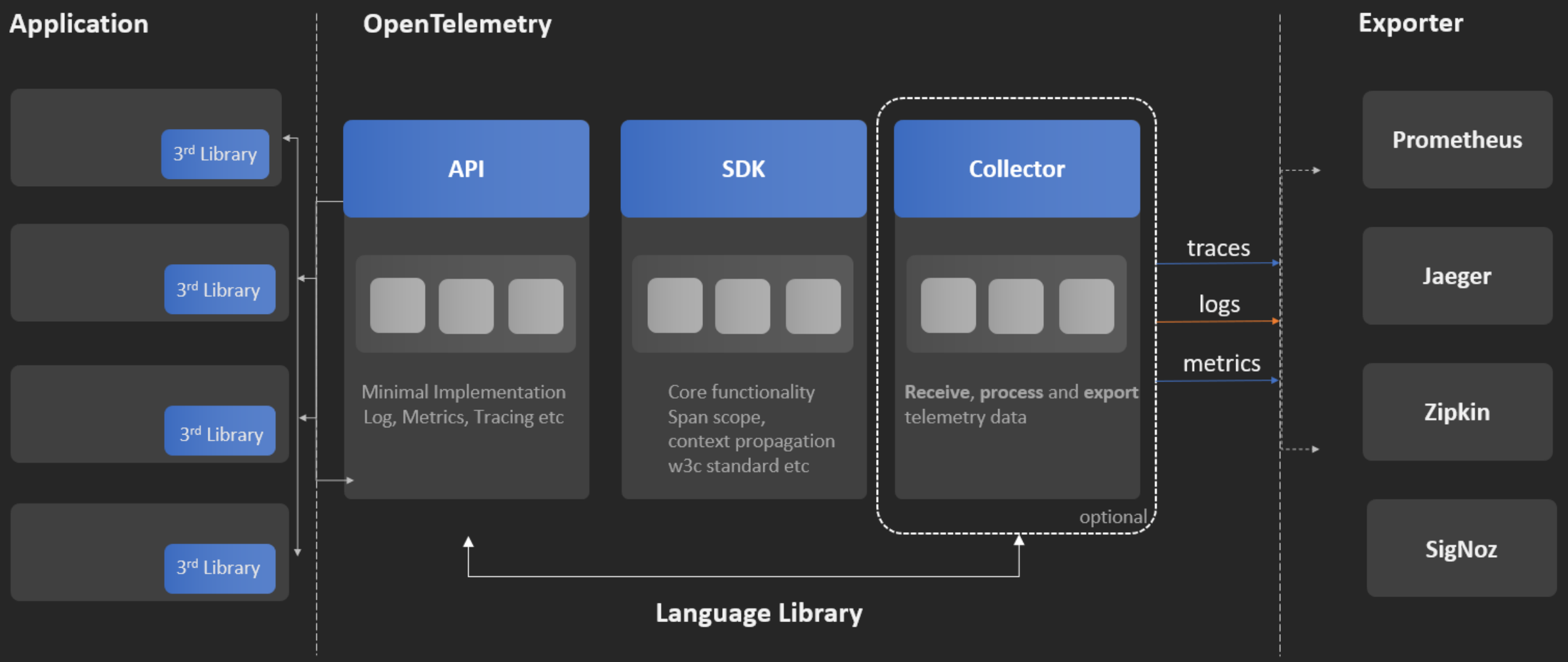
Implementations

- OTel SDKs
- OTel Collector
- OTel API

OpenTelemetry : Cloud Native Telemetry

OpenTelemetry	Tracing	Metrics	Logs
Instrumentation APIs	All languages		
Canonical implementations	All languages		
Data infrastructure	collectors		
Interop formats	w3c trace-context, write formats for observability		

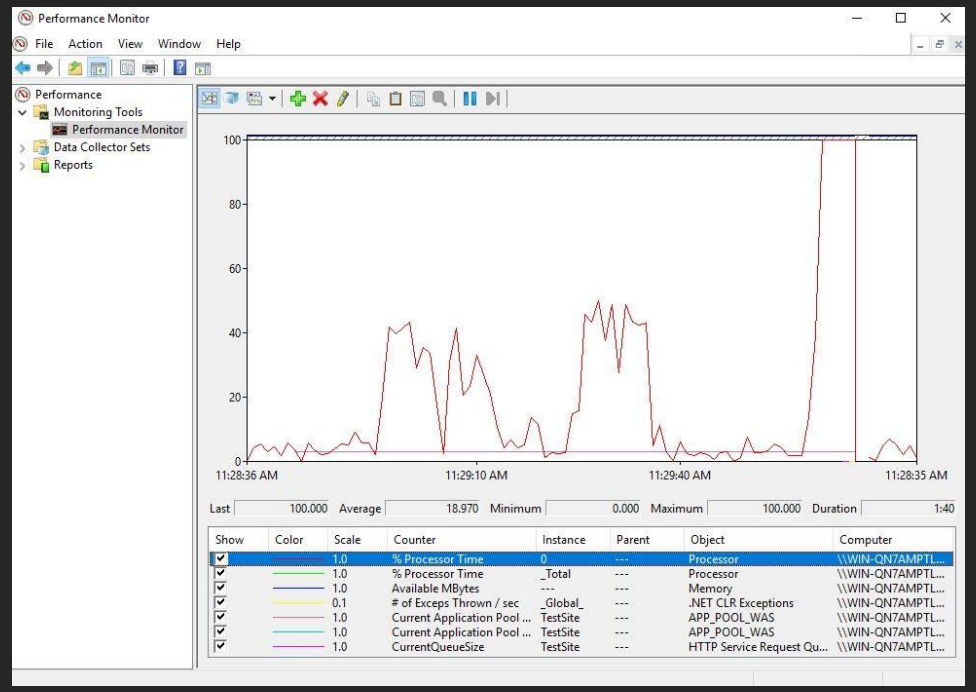
OpenTelemetry : OTel



Metrics

Do I have a problem ?

Metrics : API History



- PerformanceCounter
  - System.Diagnostics.PerformanceCounter
  - Windows only, Windosw OS 效能計數器
- EventCounters
  - .NET Core 3.1+
  - 支援變化率和平均值，不支援直方圖和百分位數、多維指標
- Third-party APIs
  - For APM vendors, AppDynamics, Application Insights, DataDog..etc

Metrics : API History

- .NET 6+
  - System.Diagnostics.Metrics
  - Instrument : counter、Gauge、Histograms
  - Key/value tags add dimensions to metrics
  - Observed through OpenTelemetry
- .NET 8
  - Build-in metrics for ASP.NET core & HttpClient
  - IMeterFactory
  - Testing Fake for Meter

Metrics : Collect

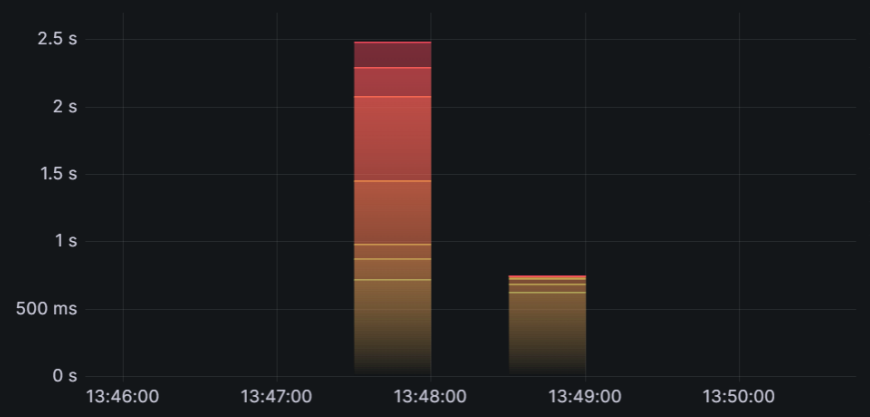
```
builder.Services.AddOpenTelemetry().WithMetrics(opts => opts
    .SetResourceBuilder(ResourceBuilder.CreateDefault().AddService("BookStore.WebApi"))
    .AddMeter(builder.Configuration.GetValue<string>("BookStoreMeterName"))
    .AddAspNetCoreInstrumentation()
    .AddRuntimeInstrumentation()
    .AddProcessInstrumentation()
    .AddOtlpExporter(opts =>
    {
        opts.Endpoint = new Uri(builder.Configuration["Otlp:Endpoint"]);
    }));
```

Metrics : Collect

- Microsoft.AspNetCore.Server.Kestrel
  - kestrel.active_connections
  - kestrel.connection.duration
  - kestrel.rejected_connections
  - kestrel.queued_connections
  - kestrel.queued_requests
  - kestrel.upgraded_connections
  - kestrel.tls_handshake.duration
  - kestrel.active_tls_handshakes
- Microsoft.AspNetCore.Http.Connections
  - signalr.server.connection.duration
  - signalr.server.active_connections
- Microsoft.AspNetCore.Hosting
  - http.server.request.duration
  - http.server.active_requests
- Microsoft.AspNetCore.Routing
  - aspnetcore.routing.match_attempts
- Microsoft.AspNetCore.Diagnostics
  - aspnetcore.diagnostics.exceptions
- Microsoft.AspNetCore.RateLimiting
  - aspnetcore.rate_limiting.active_request_leases
  - aspnetcore.rate_limiting.request_lease.duration
  - aspnetcore.rate_limiting.queued_requests
  - aspnetcore.rate_limiting.request.time_in_queue
  - aspnetcore.rate_limiting.requests
- System.Net.NameResolution
  - dns.lookup.duration
- System.Net.Http
  - http.client.open_connections
  - http.client.connection.duration
  - http.client.request.duration
  - http.client.request.time_in_queue
  - http.client.active_requests

Job otel-collector Instance otel-collector:8889

Requests Duration



Name	Last *	Min	Max
p50	625 ms	625 ms	719 ms
p75	0 ms	688 ms	875 ms
p90	0 ms	725 ms	980 ms
p95	0 ms	738 ms	1.45 s
p98	0 ms	745 ms	2.08 s
p99	0 ms	748 ms	2.29 s
p99.9	0 ms	750 ms	2.48 s

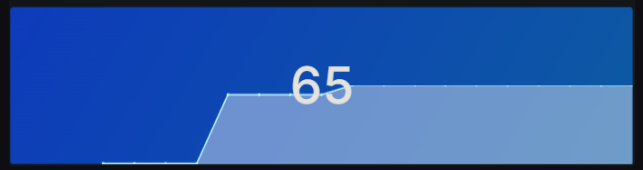
Errors Rate

Name	Last *	Min	Max
All	0%	0%	0%
4XX	0%	0%	0%
5XX	0%	0%	0%

Current Connections

Current Requests

Total Requests

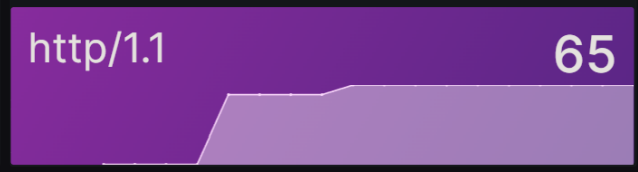


Total Unhandled Exceptions

0

Requests Secured

Requests HTTP Protocol



Top 10 Requested Endpoints

Endpoint	Requests
POST api/Books	17
POST api/Inventories	15
POST api/Orders	10
POST api/Categories	8
PUT api/Books	4
DELETE api/Orders/{id:int}	3

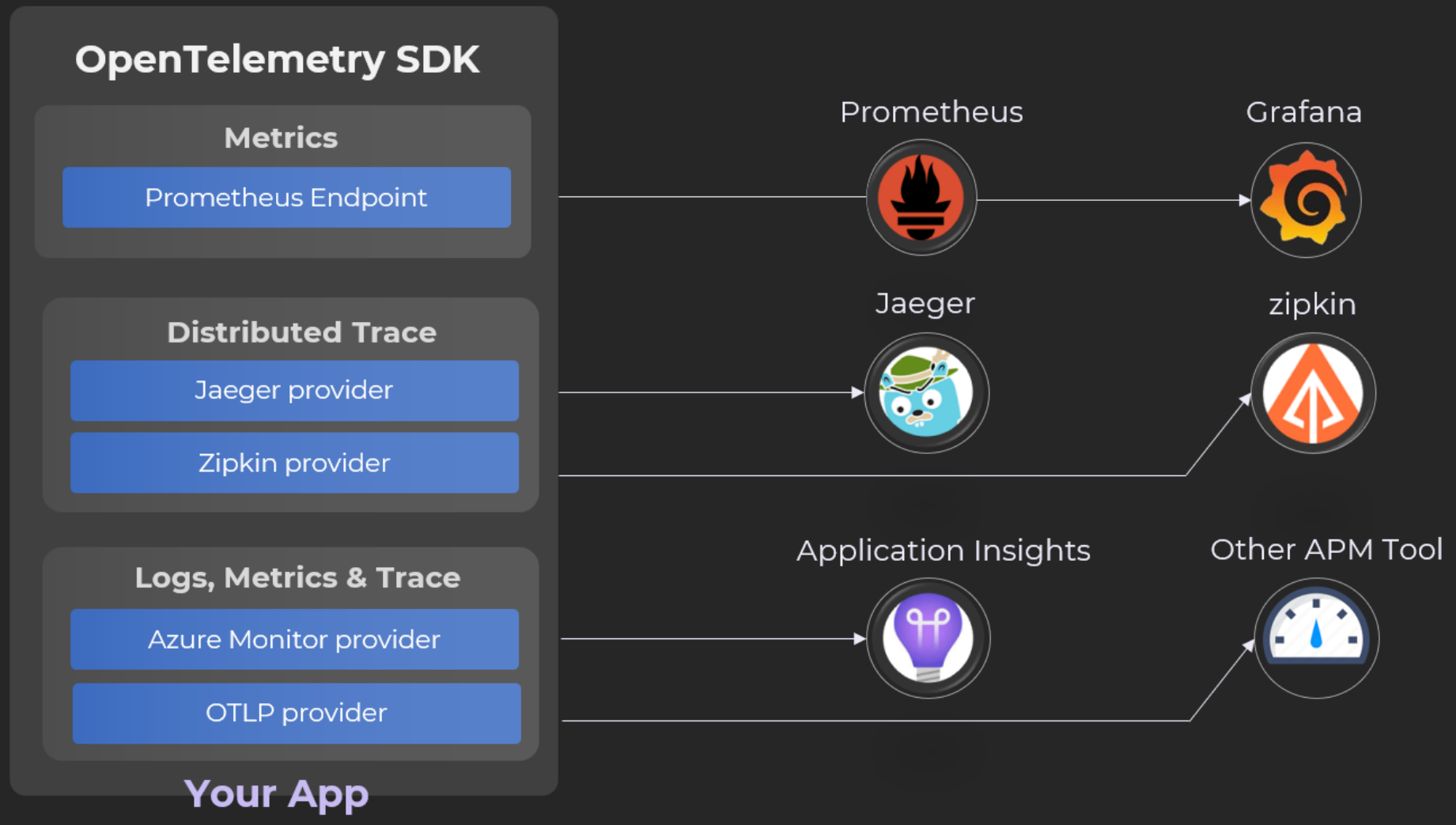
Top 10 Unhandled Exception Endpoints

No data

Traces

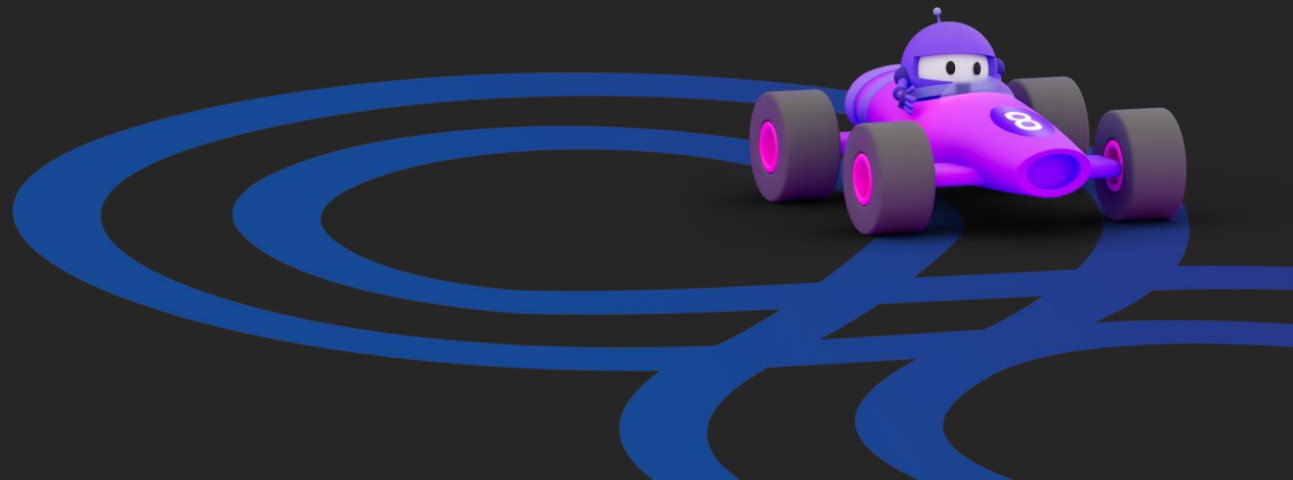
Where is the problem ?

Observing .NET Apps with OpenTelemetry



Demo

.NET Cloud Native apps



Structured Log

What is causing the problem ?

Logging API : ILogger

- ILogger is an abstraction over multiple log sources
  - Usually featched via dependency injection (DI)
  - Usually using Ilogger<T> where T provides the name of the log source
- Source are configured in code & configuration
  - Automatic in ASP.NET applications
  - Namespace : Microsoft.Extensions.Logging
- LogLevel
  - Information、Warning、Error、Trace、Debug

Benchmarking the different approaches.

- Simple to use, but sub-optimal in high frequency applications
- Parameter expressions have to be calculated, ref values are boxed
- Don't use string interpolation, You lose the structured logs

```
// Using interpolation instead of structured logging
_logger.LogInformation($"Writing hello world response to {person}");
```

- Correct number of parameters

```
// missing Reason
_logger.LogInformation("hello world  to {Person} because {Reason}", person);
```

Logging API : LoggerMessage

- .NET 6 use Action delegate parameters are strongly-type (no boxing)
- provider LoggerMessageAttribute to create logging delegate

```
[LoggerMessage(Level = LogLevel.Information,
Message = "Hello World! Logging is {Description}.")]
    static partial void LogStartupMessage(ILogger logger, string description);
```

- Generates similar code to LoggerMessage.Define
- Automatically handles exception parameters

Logging API : LogProperties

- .NET 8 , Personal Identifiable Information (PII)
- Logging whole objects with the LogProperties Attribute
- Namespace
  - Microsoft.Extensions.Telemetry.
  - Microsoft.Extensions.Telemetry.Abstractions

```
[LoggerMessage(Level = LogLevel.Debug,Message = "Generated Forecast")]
private static partial void GeneratedForecast(
    ILogger logger,
    [LogProperties] WeatherForecast forecast); // [LogProperties]
```

Logging API : LogProperties

- Properties all written out

```
{
  "EventId": 0,
  "LogLevel": "Debug",
  "Category": "Handler",
  "Message": "Generated Forecast",
  "State": {
    "Message": "{OriginalFormat}=Generated Forecast,forecast.TemperatureF=125,forecast.
    "{OriginalFormat}": "Generated Forecast",
    "forecast.TemperatureF": 125,
    "forecast.TemperatureC": 52,
    "forecast.Date": "11/27/2023"
  }
```

Resilience

彈性

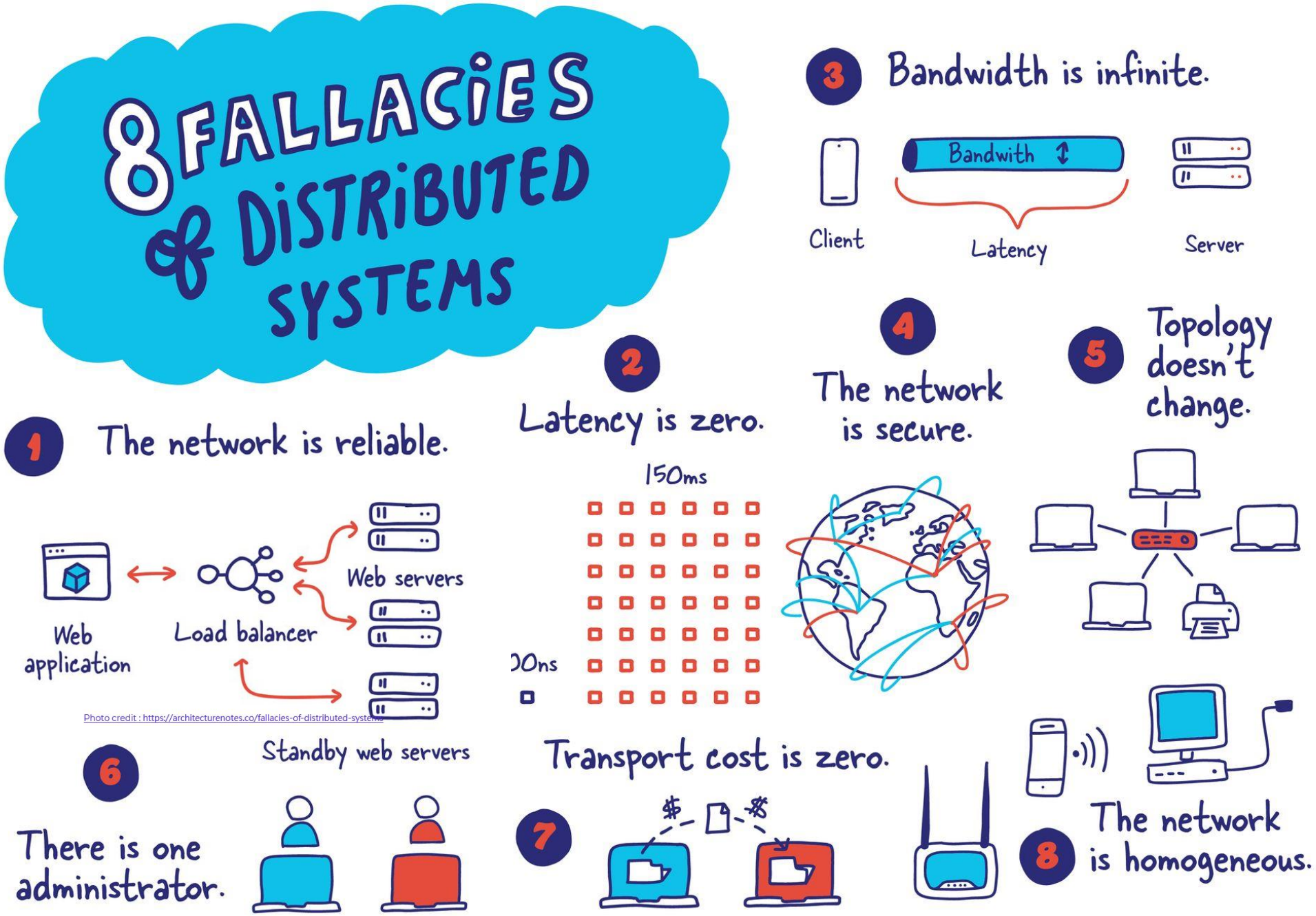


Photo credit : https://architecturenotes.co/fallacies-of-distributed-systems

Resilience : Polly

- Handling transient failures and improve the resilience of your applications.
- Most popular .NET resilient library (almost 500 million)
- Microsoft & Polly community to develop the **v8** version
  - Unified sync/async flows、Built-in telemetry、fluent syntax
- Namespace
  - Microsoft.Extensions.Resilience.
  - Microsoft.Extensions.Http.Resilience

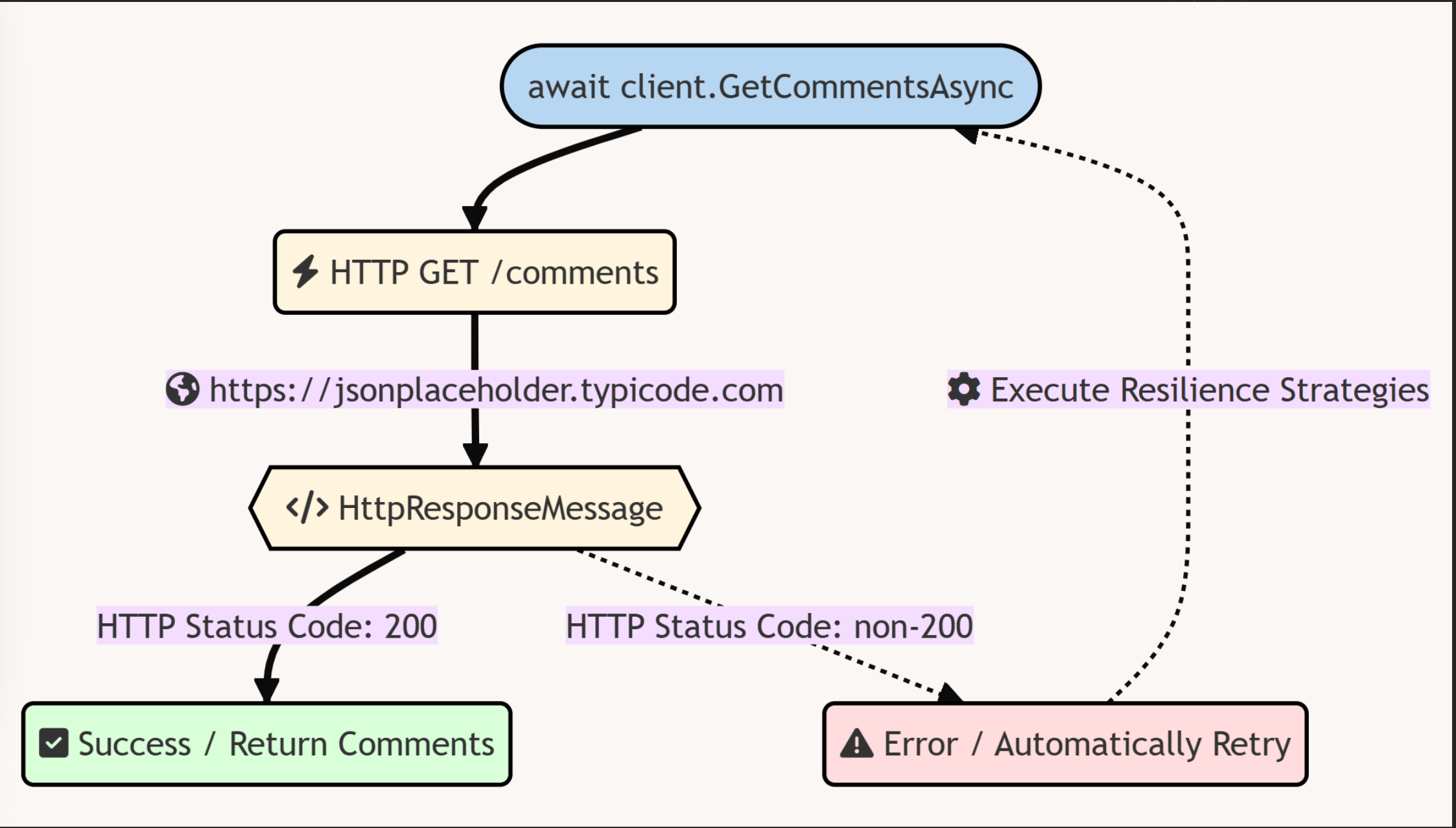


Photo credit : https://learn.microsoft.com/zh-tw/dotnet/core/resilience/http-resilience?tabs=dotnet-cli

Resilience : Resilience pipeline

- API for build and using HTTP resilience pipelines
  - Customer pipeline
  - Standard / standard hedging pipeline

```
var pipeline = new ResiliencePipelineBuilder<HttpResponseMessage>()
    .AddRetry(new RetryStrategyOptions<HttpResponseMessage>
    {
        MaxRetryAttempts = 4,
        Delay = TimeSpan.FromSeconds(2),
        BackoffType = DelayBackoffType.Exponential
    })
    .AddTimeout(TimeSpan.FromSeconds(5))
    .Build();
```

Service Discovery

- Service Discovery
  - A service registry, which is a database of services and their locations
  - A client, which queries the registry to find out where a service is
  - Health check
- Microsoft.Extensions.ServiceDiscovery
  - .NET 8
  - builder.Services.AddServiceDiscovery()

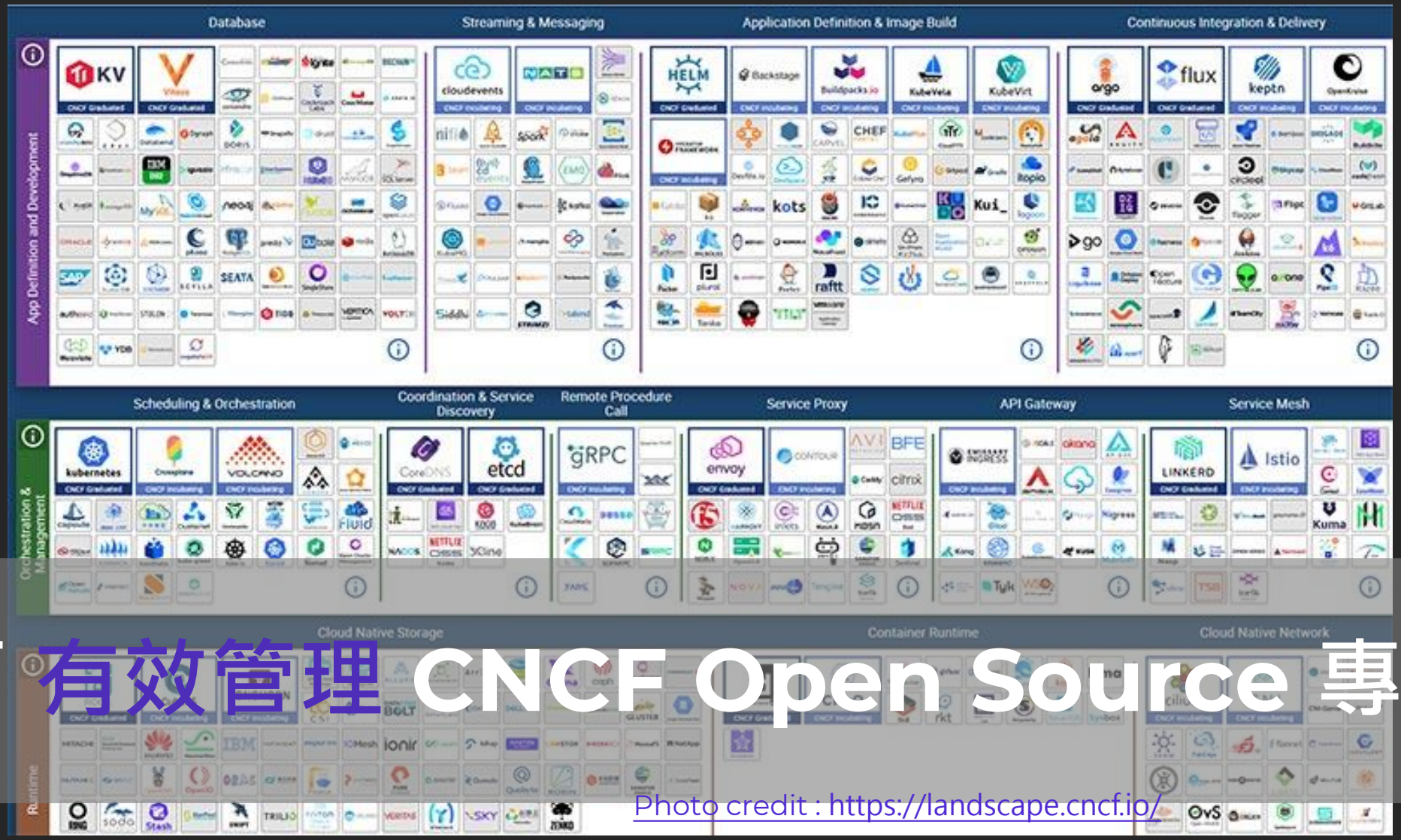
Cloud Native Apps 特性

- Observable
  - 讓你更了解系統的狀況，實作 OpenTelemetry
  - Signals : Metrics、Trace、Logging and Profiling
- Resilient
  - 彈性 / 靈活度
  - 實作 Polly V8
- Manageable ?

.NET Aspire

雲原生應用開發框架

CNCF Landscape



如何有效管理 CNCF Open Source 專案

Photo credit : https://landscape.cncf.io/

Why .NET Aspire

The feedback from Developers

- How to get my web App to talk to API app ?
- What should I do with dev/Production config ?
- Should/shouldn’t I use containers ?
- What should I do with E2E logging/monitor ?
- How to put them together property ?
- How to put useful package together properly ?

Example : How to add Redis Cache to Project

- Add StackExchange.Redis package
- Add AspNetCore.HealthChecks.Redis package
- Add Redis in DI and configure from appsettings.json
- Add Redis health check for availability
- Add Redis Client to output logging to ILogger
- Do plumbing Redis client events/profilers to metrics
- Do wrapping Redis client with resiliency policy & logic

Now

- Add Aspire.StackExchange.Redis.OutputCaching package
- Add builder.AddRedisOutputCache("cache");
- Override default configuration through appsetting.json

.NET Aspire

.NET Aspire is

.NET Aspire is a stack for build **resilient**, **observable**, and **configurable** cloud-native application with .NET

.NET Aspire includes a curated set of components enhanced for cloud-native fundamentals including **Telemetry**, **Resilience**, **Configuration**, **Health Checks** and **composition**

.NET Aspire makes it easy to **discover**, **acquire** and **configure** essential dependencies for cloud-native application on day 1 to 100

.NET Aspire Components

NuGet: A...ge.Worker

Browse | Installed | Updates

owner:Aspire tags:component

Include prerelease

NuGet Package Manager: AspireStorage.Worker

Package source: nuget.org

Aspire.StackExchange.Redis by Microsoft, **5.63K** downloads — 8.0.0-preview.1.23557.2
A generic Redis® client that integrates with Aspire, including health checks, logging, and telemetry.

Aspire.RabbitMQ.Client by Microsoft, **3.47K** downloads — 8.0.0-preview.1.23557.2
A RabbitMQ client that integrates with Aspire, including health checks, logging, and telemetry.

Aspire.Npgsql by Microsoft, **3.28K** downloads — 8.0.0-preview.1.23557.2
A PostgreSQL® client that integrates with Aspire, including health checks, metrics, logging, and telemetry.

Aspire.StackExchange.Redis.OutputCaching by Microsoft, **2.66K** downloads — 8.0.0-preview.1.23557.2
A Redis® implementation for ASP.NET Core Output Caching that integrates with Aspire, including health...

Aspire.Npgsql.EntityFrameworkCore.PostgreSQL by Microsoft, **3.76K** downloads — 8.0.0-preview.1.23557.2
A PostgreSQL® provider for Entity Framework Core that integrates with Aspire, including connection poo...

Aspire.Azure.Storage.Blobs by Microsoft, **218** downloads — 8.0.0-preview.1.23557.2
A client for Azure Blob Storage that integrates with Aspire, including health checks, logging and telemetry.

Aspire.Microsoft.Data.SqlClient by Microsoft, **177** downloads — 8.0.0-preview.1.23557.2
A Microsoft SQL Server client that integrates with Aspire, including health checks, metrics and telemetry.

Aspire.Azure.Messaging.ServiceBus by Microsoft, **156** downloads — 8.0.0-preview.1.23557.2
A client for Azure Service Bus that integrates with Aspire, including health checks, logging and telemetry.

Aspire.Azure.Storage.Queues by Microsoft, **154** downloads — 8.0.0-preview.1.23557.2
A client for Azure Queue Storage that integrates with Aspire, including health checks, logging and telem...

Aspire.StackExchange.Redis — nuget.org

Version: Latest prerelease 8.0.0-preview.1.23557.2 — Install

Package source mapping is off. Configure

Options

Description

A generic Redis® client that integrates with Aspire, including health checks, logging, and telemetry.

Version:	8.0.0-preview.1.23557.2
Author(s):	Microsoft
License:	MIT
Readme:	View Readme
Downloads:	5,631
Date published:	Tuesday, November 14, 2023 (11/14/2023)
Project URL:	https://github.com/dotnet/aspire
Report Abuse:	https://www.nuget.org/packages/Aspire.StackExchange.Redis/8.0.0-preview.1.23557.2/ReportAbuse
Tags:	aspire, component, cloud, cache, caching, redis

Dependencies

net8.0

AspNetCore.HealthChecks.Redis (>= 7.0.1)
Microsoft.Extensions.Configuration.Binder (>= 8.0.0)
Microsoft.Extensions.Diagnostics.HealthChecks (>= 8.0.0)
Microsoft.Extensions.Hosting.Abstractions (>= 8.0.0)

Demo

.NET Aspire Dashboard

.NET Aspire

目的：

- 簡化開發分散式系統的複雜性
- Tye，實驗性質開發人員工具
- AppHost、ServiceDefault
- Dashboard
  - 作為應用程式監控和檢查
  - 追蹤應用程式日誌、追蹤和配置
- Preview 階段

Takeaway

總結

Observability

Resilience

Health Checks

Testing/Fakes

- Extensions.Resilience
- Extensions.Http.Resilience
- Extensions.Diagnostics.HealthChecks.Common
- Extensions.Diagnostics.Probes
- Extensions.Telemetry
- Extensions.Compliance.Redaction
- Extensions.Http.Telemetry
- AspNetCore.Testing
- Hosting.Testing
- Extensions.TimeProvider.Testing

Photo credit : https://devblogs.microsoft.com/dotnet/announcing-dotnet-8/

Reference

- Building Cloud Native apps with .NET 8 | .NET Conf 2023
- Hack Together .NET: The Great .NET 8 Hack - Your stack for building Cloud Native apps
- Building resilient cloud services with .NET 8 | .NET Conf 2023
- Cloud-native development with .NET 8
- Improving your application telemetry using .NET 8 and OpenTelemetry | .NET Conf 2023
- Observability 101 @ITHome 鐵人賽

特別感謝

.NET Conf
TAIWAN

Microsoft

tsmc

Build School

成功

HackMD

Microsoft® Most Valuable Professional

STUDY4
為學習而生

Azure Taiwan User Group

以及各位參與活動的你們

THANK YOU

{ .NET Conf ◦ Everyone }

Any question ?

Marcus 的學習筆記

marcus tung