.NET Conf
TAIWAN
2024

STUDY4

為 學 習 而 生

.NET Conf TAIWAN 2024

特別感謝

Microsoft® MVP Most Valuable Professional

tsmc

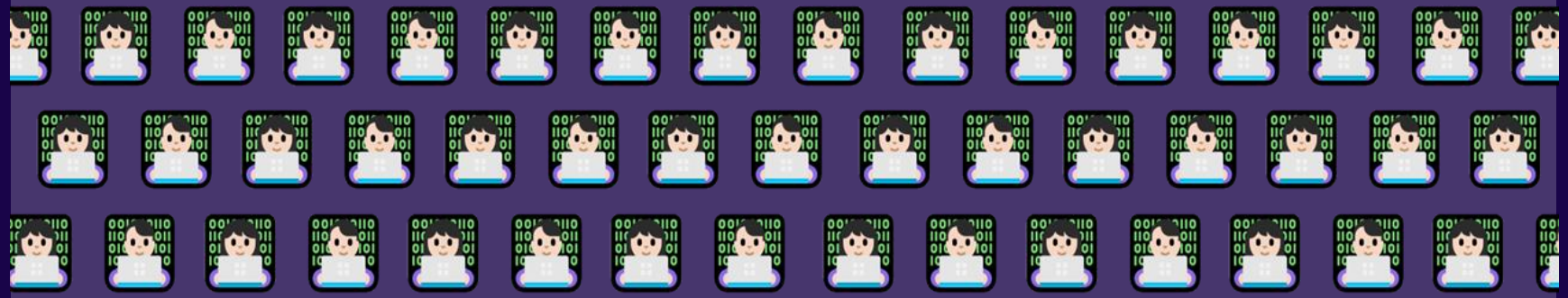
好房網 HouseFun

街口支付 JKOPAY

新加坡商 鈦坦科技 TITANSOFT

STUDY4 為學習而生

以及各位參與活動的各位



.NET Conf TAIWAN 2024

咱一路建置Observability的風風雨雨

Chuck
TSMC Senior Engineer

Observability是什麼？

.NET Conf TAIWAN 2024

The ability to measure the internal state of a system only by its external outputs

A Survey on Observability of Distributed Edge & Container-Based Microservices | IEEE Journals & Magazine | IEEE Xplore

Observability是什麼？

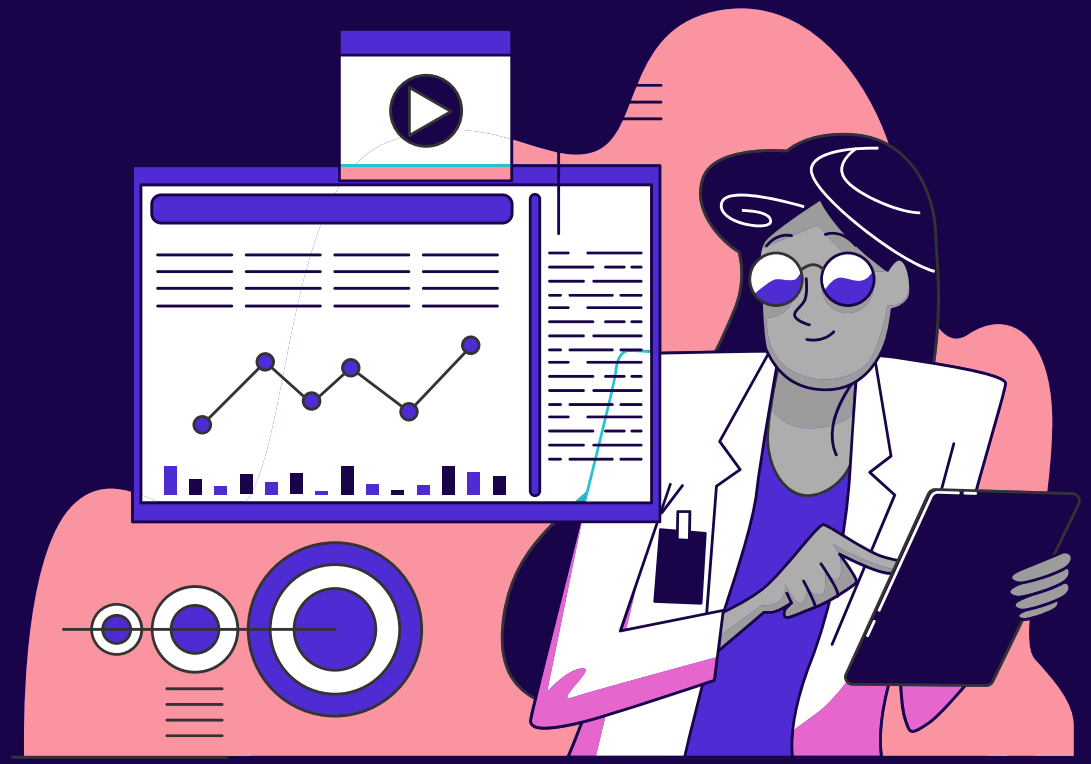
Observability是什麼？

Observability（可觀察性）是一個源自控制理論的概念，指的是一個系統的內部狀態能夠透過觀察外部輸出來進行推斷的能力。在軟體工程和IT運維領域，Observability 通常用來描述系統如何提供足夠的數據，讓工程師可以了解系統的運行狀況、性能問題以及故障原因。

Observability是什麼？

為什麼要建置Observability?

1. Avability
2. Performance
3. Cost management
4. Security



Observability能幫助我們什麼？

- Troubleshooting
- Optimization
- Finding Problems
- Predicting Problems
- Transparency

Observability能幫助我們什麼？

Observability 3 本柱

.NET Conf TAIWAN 2024

- **Metrics**
- **Logs**
- **Tracing**

Logs

- 最早衍生出的指標
- 透過「文字」將在某個時間下發生的的事件、資料、和有意義的資訊記錄起來。
- 結構化與非結構化格式
- 事件追蹤

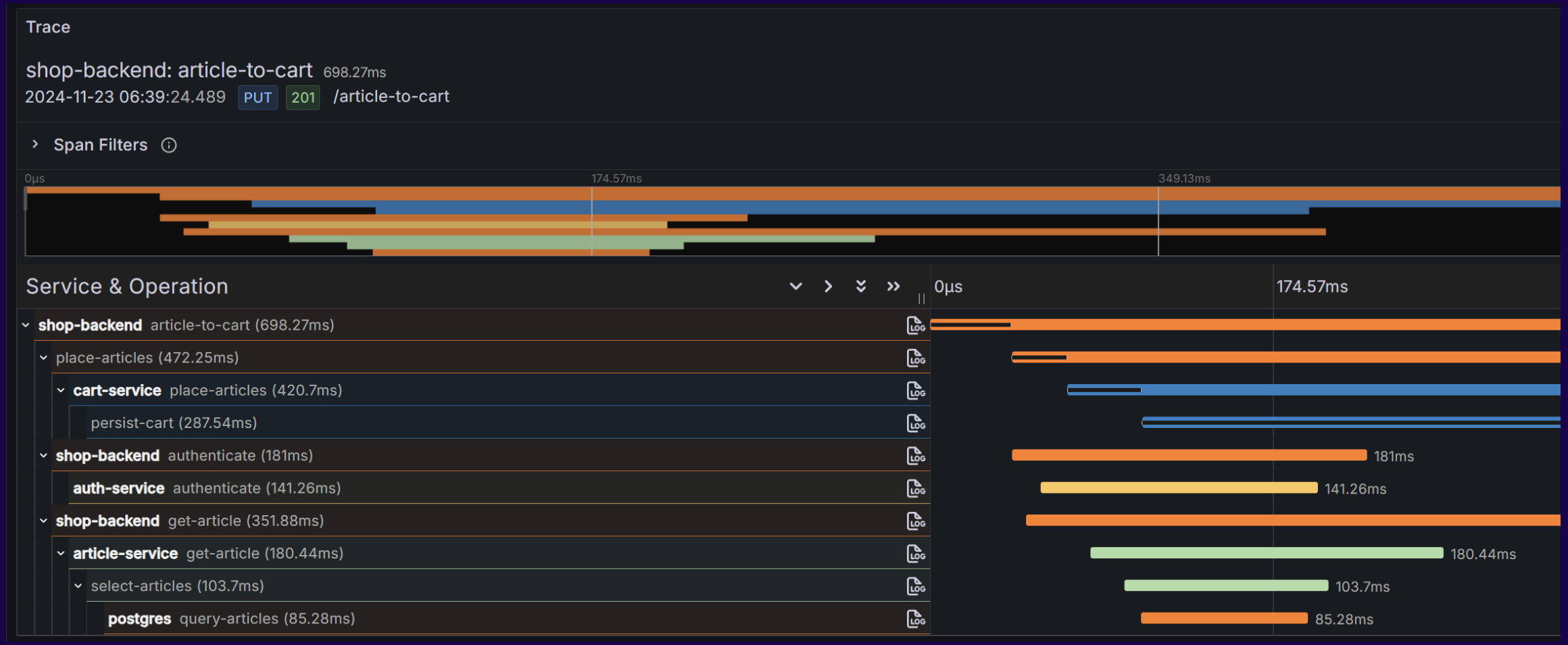
Metrics

- 透過觀察一定時間之內而量化後的數據。
- 性能監控:
  實時追蹤系統資源使用情況（如 CPU & Mem）。
  分析應用程序響應時間和吞吐量。
- Alarm設置:
  設定閾值以便在指標異常時發送告警。

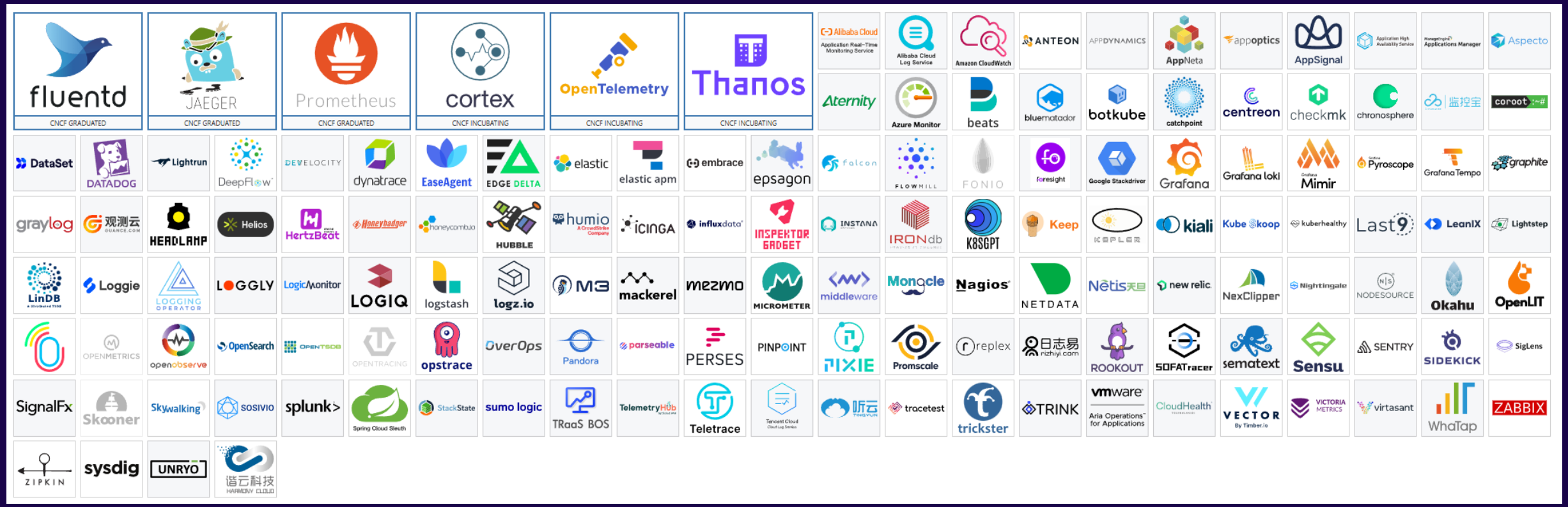
Traces

- 定義為「a tree of spans」。
- Trace記錄著每個功能其應用程式完整的執行過程。
- 執行過程會充滿著「上下游的呼叫關係」

- 跟蹤應用程式請求在系統中的流轉
- Root Cause分析



該選擇什麼工具？

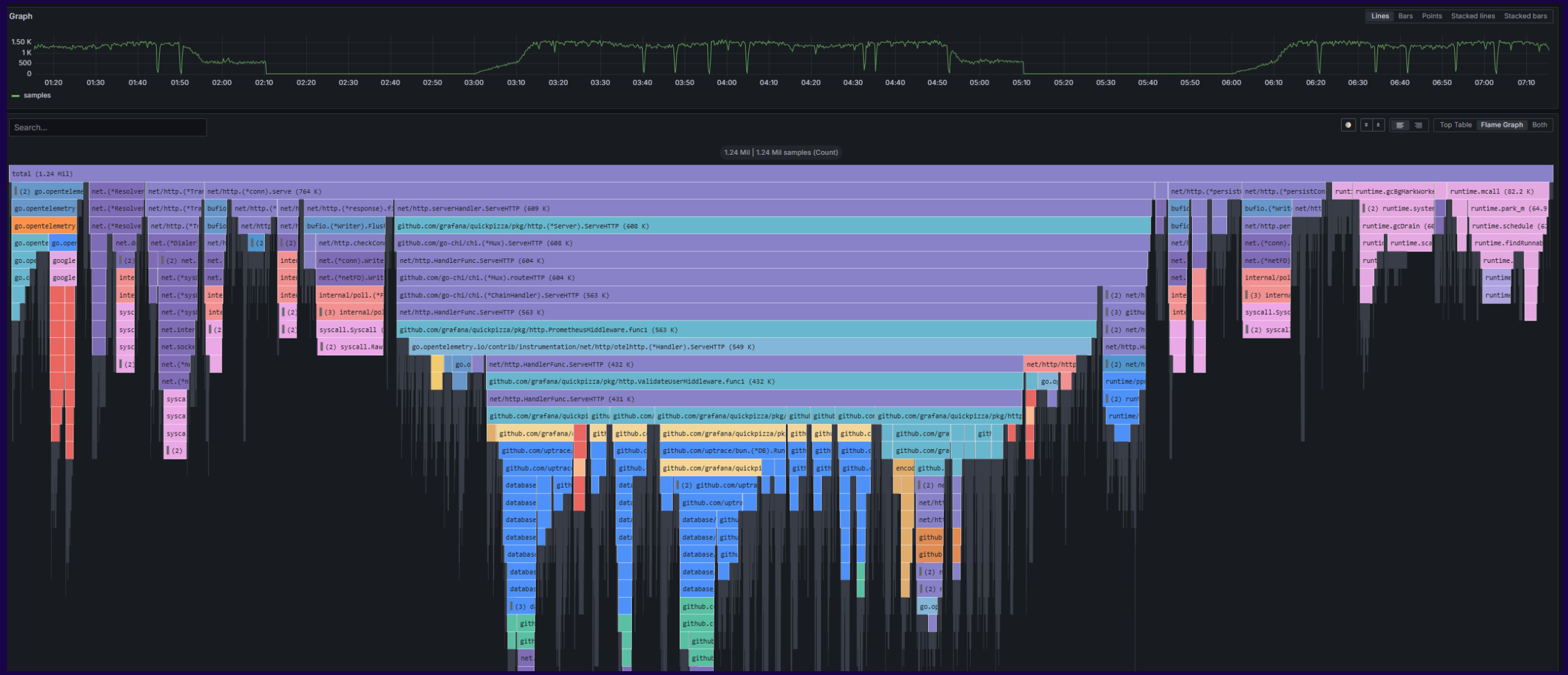

fluentd
CNCF GRADUATED
JAEGER
CNCF GRADUATED
Prometheus
CNCF GRADUATED
cortex
CNCF INCUBATING
OpenTelemetry
CNCF INCUBATING
Thanos
CNCF INCUBATING
Alibaba Cloud
Application Real-Time Monitoring Service
Alibaba Cloud Log Service
Amazon CloudWatch
ANTEON
APPDYNAMICS
AppNeta
appoptics
AppSignal
Aspecto
Aternity
Azure Monitor
beats
bluematador
botkube
catchpoint
centreon
checkmk
chronosphere
coroot
DataSet
DATADOG
Lightrun
DeepFlow
DEVELOCITY
dynatrace
EaseAgent
EDGE DELTA
elastic
elastic apm
embrace
epsagon
falcon
FLOWMILL
FONIO
foresight
Google Stackdriver
Grafana
Grafana loki
Mimir
Pyroscope
Grafana Tempo
graphite
graylog
观测云
HEADLAMP
Helios
HertzBeat
Honeybadger
honeycombio
HUBBLE
humio
ICINGA
influxdata
INSPEKTOR GADGET
INSTANA
IRONdb
K8SGPT
Keep
KEPLER
kiali
Kube koop
kuberhealthy
Last9
LeanIX
Lightstep
LinDB
Loggie
LOGGING OPERATOR
LOGGLY
LogicMonitor
LOGIQ
logstash
logz.io
M3
mackerel
mezmo
MICROMETER
middleware
Mongcle
Nagios
NETDATA
Nētis天旦
new relic
NexClipper
Nightingale
NODESOURCE
Okahu
OpenLIT
OPENMETRICS
openobserve
OpenSearch
OPENTSDB
OPENTRACING
opstrace
OverOps
Pandora
parseable
PERSES
PINPOINT
PIXIE
Promscale
replex
日志易
ROOKOUT
SOFATracer
sematext
Sensu
SENTRY
SIDEKICK
SigLens
SignalFx
Skooner
Skywalking
SOSIVIO
splunk>
Spring Cloud Sleuth
StackState
sumo logic
TRaaS BOS
TelemetryHub
Teletrace
Tencent Cloud
tracetest
trickster
TRINK
vmware
Aria Operations for Applications
CloudHealth
VECTOR
VICTORIA METRICS
virtasant
WhaTap
ZABBIX
ZIPKIN
sysdig
UNRYO
谐云科技

- 分布式架構
- 強大的搜索和查詢語言
- 集中式日誌管理
- 日誌結構化與非結構化支持
- 標準化與統一性
- 分布式追蹤
- 語言支持廣泛
- 多種協議支持

- Open Metrics
- 可以用簡單的方式將應用程式的指標暴露出來
- 提供多種程式語言的Libraries
- Alarm管理

Profiling

- 用於監控應用程序的性能行為，幫助開發和運維團隊深入了解系統資源使用情況，解決性能瓶頸問題。
- 深入了解資源的使用情況，確保應用程序有效利用系統資源。
- 快速定位異常問題的根本原因，減少 MTTR
- 歷史分析與趨勢追蹤


Graph
Lines
Bars
Points
Stacked lines
Stacked bars
1.50 K
1 K
500
0
01:20
01:30
01:40
01:50
02:00
02:10
02:20
02:30
02:40
02:50
03:00
03:10
03:20
03:30
03:40
03:50
04:00
04:10
04:20
04:30
04:40
04:50
05:00
05:10
05:20
05:30
05:40
05:50
06:00
06:10
06:20
06:30
06:40
06:50
07:00
07:10
samples
Search...
Top Table
Flame Graph
Both
1.24 Mil | 1.24 Mil samples (Count)
total (1.24 Mil)
net/http.(*conn).serve (764 K)
net/http.serverHandler.ServeHTTP (609 K)
github.com/grafana/quickpizza/pkg/http.(*Server).ServeHTTP (608 K)
github.com/go-chi/chi.(*Mux).ServeHTTP (608 K)
net/http.HandlerFunc.ServeHTTP (604 K)
github.com/go-chi/chi.(*Mux).routeHTTP (604 K)
github.com/go-chi/chi.(*ChainHandler).ServeHTTP (563 K)
net/http.HandlerFunc.ServeHTTP (563 K)
github.com/grafana/quickpizza/pkg/http.PrometheusMiddleware.func1 (563 K)
go.opentelemetry.io/contrib/instrumentation/net/http/otelhttp.(*Handler).ServeHTTP (549 K)
net/http.HandlerFunc.ServeHTTP (432 K)
github.com/grafana/quickpizza/pkg/http.ValidateUserMiddleware.func1 (432 K)
net/http.HandlerFunc.ServeHTTP (431 K)
runtime.gcBgMarkWorke
runtime.mcall (82.2 K)
runtime.park_m (64.9
runtime.schedule (6
runtime.findRunnab

小明家的架構?


Grafana
Kibana
Prometheus
Tempo
Loki
Elasticsearch
Fluentd + OtelCollector
Kubernetes
Thanos

.NET Conf TAIWAN 2024

風風雨雨的部分

小明他　公司開到全世界

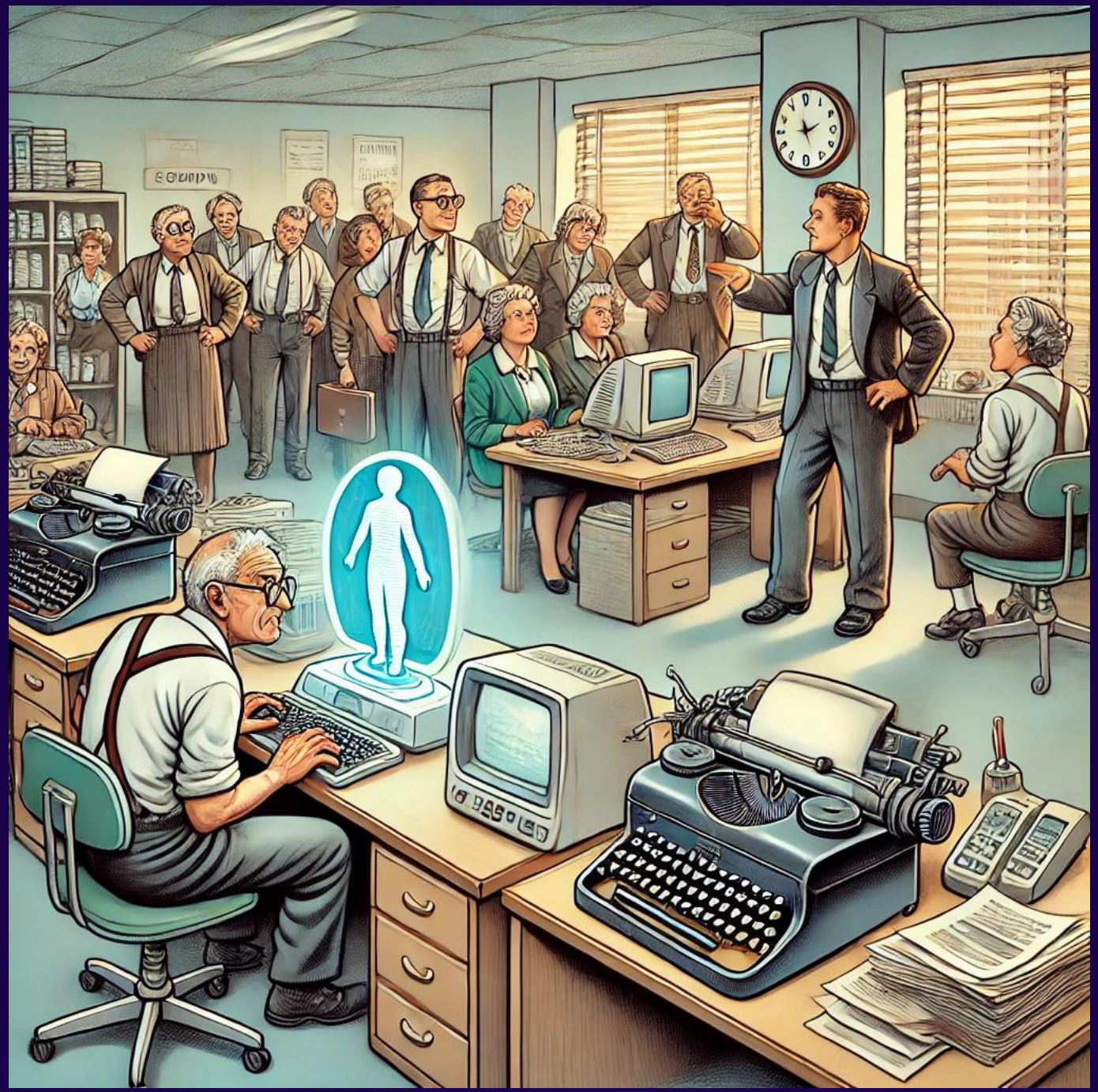
Azure/GCP/AWS

From Hero

To

Zero

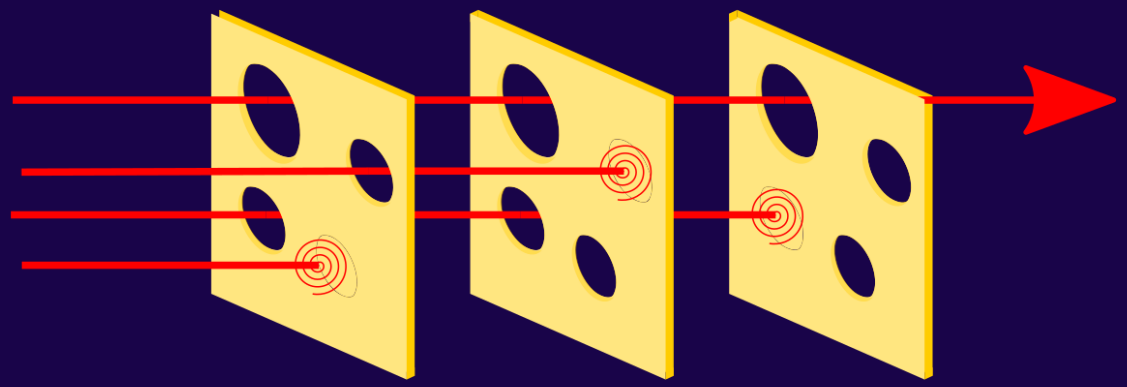
.NET Conf TAIWAN 2024



.NET Conf TAIWAN 2024

挑戰

- 公司規模龐大，團隊產品需支援全世界的公司員工，需要大量的維運成本
- 且基於某些原因，公司規定，地端Only
- 老舊系統過多且架構複雜還無法汰換還要跟新產品整合監控
- 團隊技術落差龐大，不是所有人都知道Observability的必要性及價值



先聊一下導入DevOps時我們悟了什麼

開方式編碼	頻率
文化和心態轉變	138
持續學習	111
領導承諾缺乏	90
團隊適應能力	84
不確定性恐懼	77
抵抗心理	72
領導行為不一	69

- 組織文化和領導支持
- 持續發展和學習
- 技術和流程整合

Solution?

- ~~離職~~
- 先瞄重點 ：針對最需要改進的關鍵系統（如果出事，會有數萬人憤怒且可能影響到你職業生涯的系統）優先導入 Observability，再逐漸拓展到各個據點。
- Config統一管理：盡量透過helm chart分開打包佈署並將各據點環境參數化
- 自動化：As a Engineer 某方面已經成為default該知道該會該利用的一點
- 老王賣瓜： 妥善教育團隊，告訴他們這個應用實例有多棒，破除萬難也要導入
- 真的還有問題，找老闆協調
- 還是有問題 → ~~離職~~

- 歡迎來小明的公司一起解決~~面對~~問題


.NET Conf
TAIWAN
2024
Observability
2.0

Three Pillars
Metrics,Logs,Traces

Observability

Single source of truth:
wide structured logs

Monitor Everything!

Bad “Best Practice”:
Monitor only what you need and understand.
Why this is a myth:

- **Holistic View**
  To understand the dynamics of modern infrastructure we need comprehensive and holistic views of the technologies involved. Cherry picking data usually means blind spots.
- **Easier and Faster Root Cause Analysis**
  When issues arise, having access to a broad set of data helps in pinpointing the source of the problem.
- **Proactive Issue Detection**
  With a full spectrum of monitoring, patterns and anomalies can be identified early. Such issues may not be evident when only a subset of the data is considered.
- **Adaptability to Changing Environments**
  Modern IT environments are dynamic and constantly evolving. What is considered important today may change tomorrow due to shifts in application architecture, user behavior, or infrastructure changes.

Real-Time (per-second) & Low-Latency!

Bad “Best Practice”:
Monitoring every 10, 15 or 60 seconds.
Why this is a myth:

- **True Performance Monitoring**
  Virtualized environments are highly dynamic, non-linear, and unpredictable. Monitoring per second reveals the true performance characteristics of the infrastructure and applications by capturing the fine-grained details of their behavior
- **Improved Correlation and Context**

Having data points every second enhances the ability to correlate events and metrics across different parts of the system. This improved correlation helps in understanding the context of issues, such as determining the sequence of events leading up to a failure or performance degradation.

- **Faster Response to Issues**
  The effects of your corrective actions, or even configuration changes, are immediately reflected on the monitoring data, enabling quicker identification and mitigation of problems.

Structured Data


```
{
    "timestamp": "2024-11-17T12:34:56Z",
    "level": "ERROR",
    "service": "auth-service",
    "user_id": "12345",
    "request_id": "abcde-67890",
    "error_code": "401",
    "ip_address": "192.168.1.1",
    "session_duration": 3600,
    "custom_field_1": "extra_value_1",
    "custom_field_2": "extra_value_2"
}
```

Logs

```
{
    "trace_id": "xyz-123",
    "span_id": "span-456",
    "operation": "database_query",
    "start_time": "2024-11-17T12:35:00Z",
    "end_time": "2024-11-17T12:35:01Z",
    "duration_ms": 1000,
    "tags": {
        "db.statement": "SELECT * FROM users WHERE id = ?",
        "db.type": "PostgreSQL",
        "host": "db-server-1",
        "region": "us-east-1"
    },
    "annotations": [
        {"timestamp": "2024-11-17T12:35:00.500Z", "event": "query_sent"},
        {"timestamp": "2024-11-17T12:35:00.800Z", "event": "response_received"}
    ]
}
```

Tracing

Structured Data


```
{
    "host": "server-1",
    "timestamp": "2024-11-17T12:37:00Z",
    "cpu_usage": 85.5,
    "memory_usage_mb": 2048,
    "disk_io": {
        "read_bytes": 1048576,
        "write_bytes": 524288
    },
    "network_io": {
        "inbound_kbps": 500,
        "outbound_kbps": 450
    },
    "custom_metrics": {
        "gpu_temperature": 75,
        "process_count": 120
    }
}
```

Metrics

```
{
    "event_type": "button_click",
    "user_id": "user-789",
    "timestamp": "2024-11-17T12:36:00Z",
    "session_id": "sess-9876",
    "device": {
        "type": "mobile",
        "os": "iOS",
        "version": "16.2"
    },
    "location": {
        "latitude": 37.7749,
        "longitude": -122.4194
    },
    "custom_properties": {
        "button_id": "btn-signup",
        "page": "landing_page",
        "referrer": "google_ads"
    }
}
```

User Behavior

What you can do NOW to start moving towards observability 2.0

- Instrument your Code!
- 添加（Trace IDs）和（Span IDs），這樣你可以通過相同的事件直接追蹤Code，而無需在不同的工具之間切換。
- 將你的data store到Columnar Store中
- 使用支持高基數（High Cardinality）的Storage (例如:ClickHouse)
- 使用可探索界面的Tool，或至少使用動態的Dashboard

.NET Conf TAIWAN 2024

Thank you

Do you have any questions?

www.linkedin.com/in/chuck-chen-36156a174

Ref:

1. What_Developers_Should_Know_About_Observability – Peter Zaitsev
2. O11y 1.0 vs O11y2.0 – Charity Majors
3.A Survey on Observability of Distributed Edge & Container-Based Microservices
4. Grafana Alloy Best Practice – Eric Huang
5.應該是 Profilling 吧？-雷N
6. Netdata: Open Source, Distributed Observability Pipeline – Journey and Challenges