ARQUITECTURA DEL SOFTWARE

Escuela de Ingeniería Informática

Universidad de Oviedo

Laboratorio 12

Monitorización
Observabilidad

2025-26

Jose Emilio Labra Gayo
Pablo González
Diego Martín
Celia Melendi

Monitorización y Profiling

- **Atributo de Calidad: Observabilidad**
- **Monitorizar:** Observar comportamiento de un software
  - Cuadros de mando
  - Habitualmente en producción (después del despliegue)
- **Profiling (caracterizar)**: Medir rendimiento de un software mientras se ejecuta
  - Identificar partes que contribuyen a un problema
  - Encontrar dónde centrar esfuerzos para mejorar rendimiento
  - Suele hacerse en Desarrollo/pruebas, antes del despliegue

Profiling

Monitorizar una aplicación mientras se ejecuta

Registrar uso de CPU, memoria, hilos, etc.

JavaScript:

- Chrome (Timeline), Firefox Developer Edition (Performance tool)

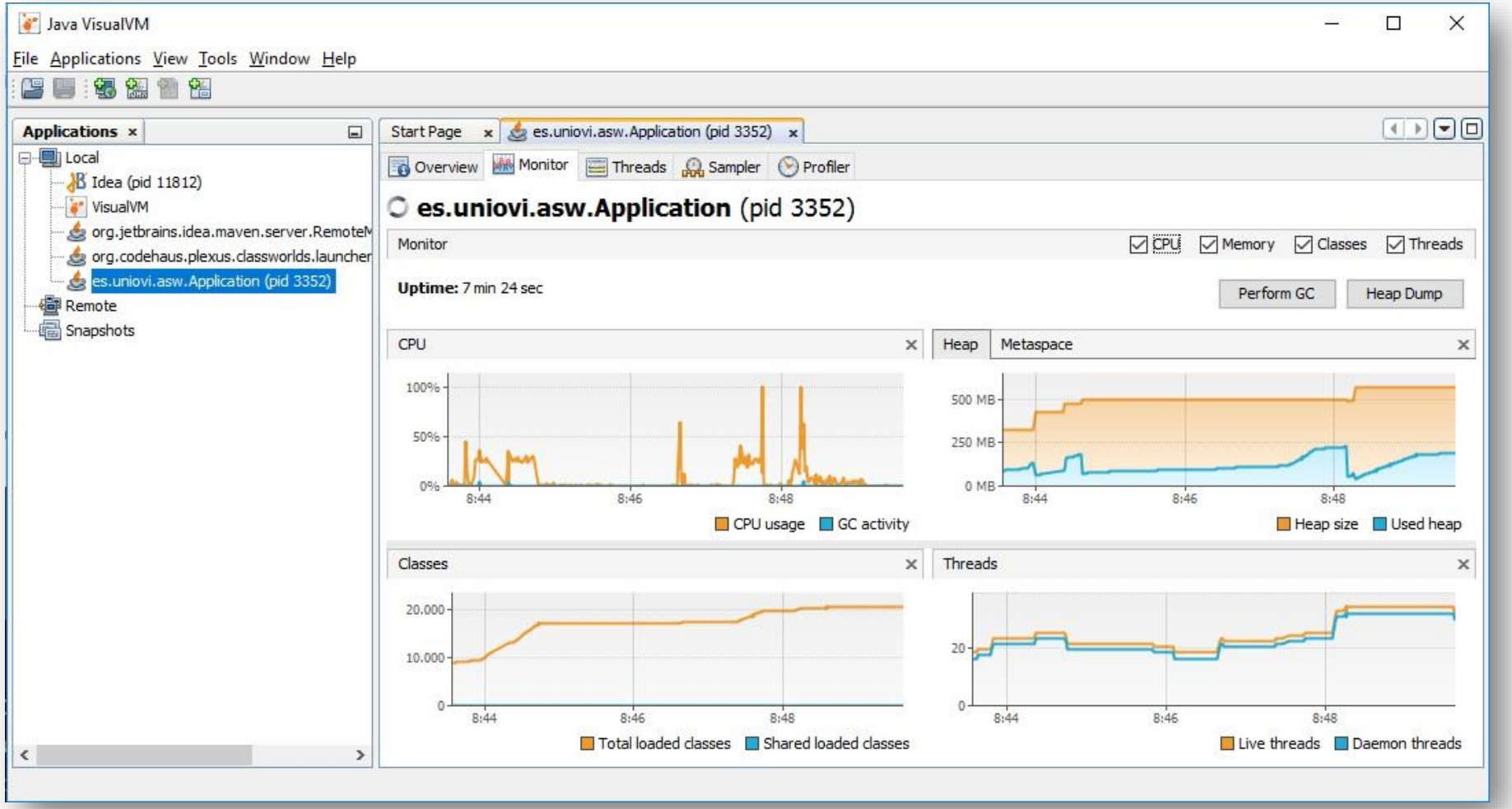
Herramientas de servidor:

- JVisualVM, JProfiler, YourKit, Jconsole, etc.
- Graphite, Datdog, Prometheus, Graphana

VisualVM

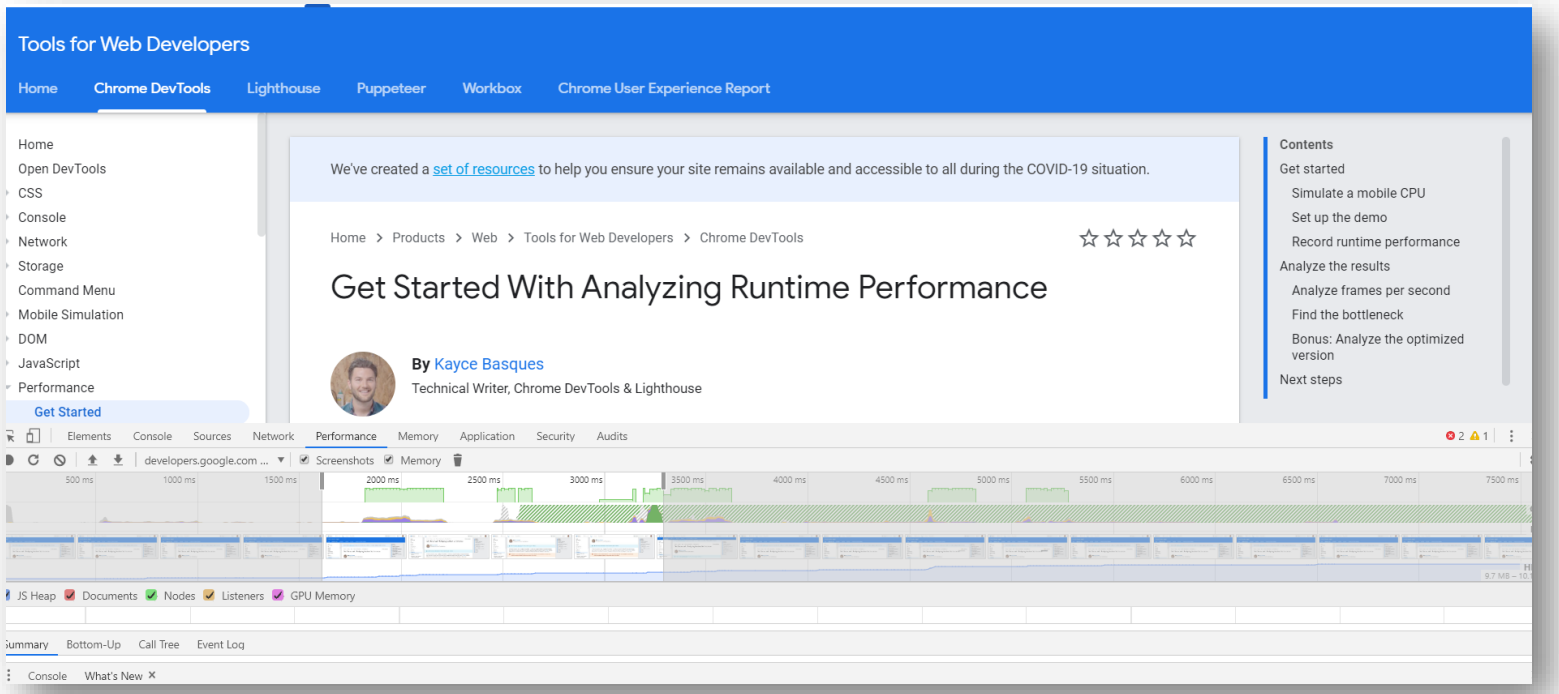
- https://visualvm.github.io/
- Ya está instalada con el JDK: `jvisualvm`

Server / Java: JVisualVM



Navegador: developer tools

- Monitorizar/chequear rendimiento



https://developers.google.com/web/tools/chrome-devtools/evaluate-performance

Ejemplo: Google Chrome

Modo incognito

En la esquina superior derecha, click en los tres puntos y nueva ventana incógnito

Windows, Linux, or Chrome OS: Ctrl + Shift + n.

Mac: ⌘ + Shift + n.

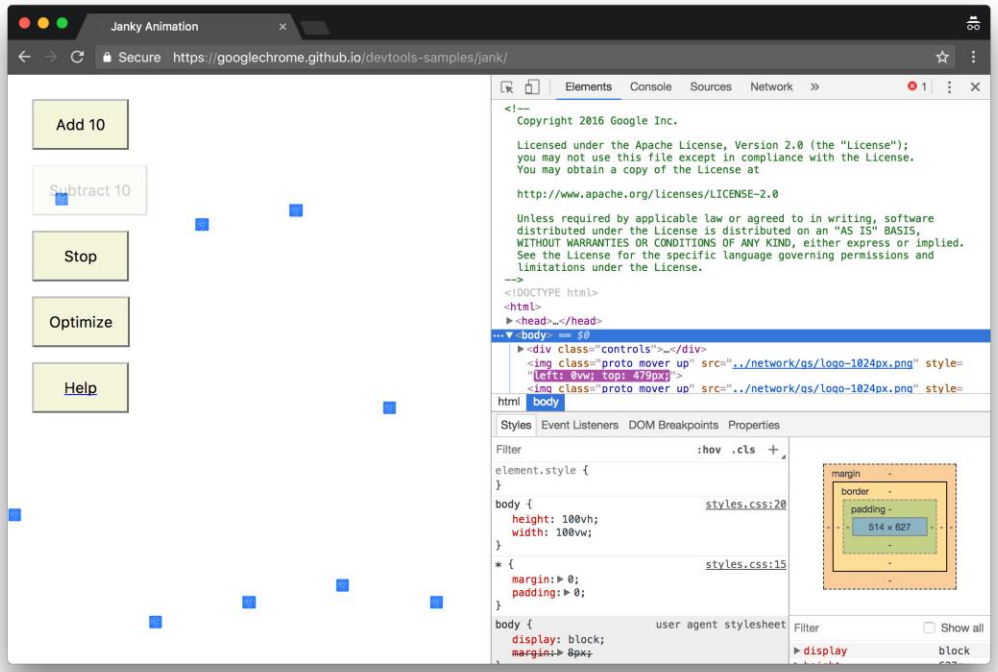
Chrome DevTools

Windows, Linux: Control+Shift+I

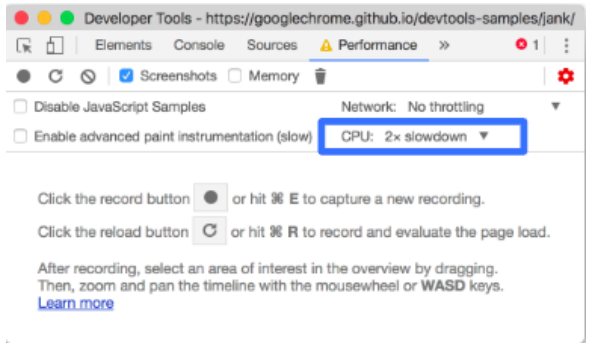
Mac: Command+Option+I

Ejemplo: Google Chrome

https://googlechrome.github.io/devtools-samples/jank/



Performance>CPU>2 x Slowdown



Performance>Record

click Add 10 (20 veces)

Optimize / Un-optimize

Stop

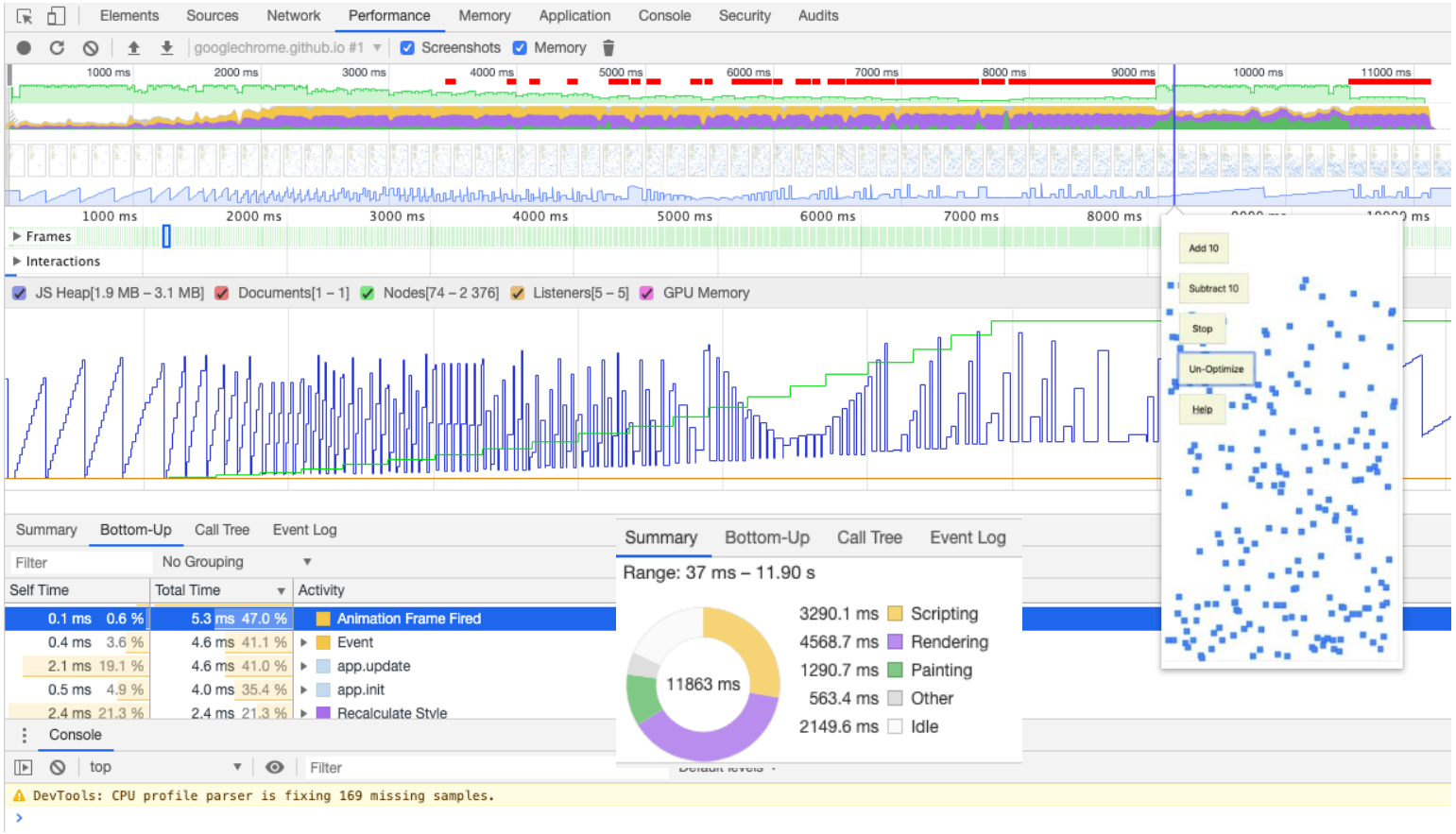
Ejemplo: Google Chrome

Profile result:

Frames por segundo

CPU

Cuellos de botella



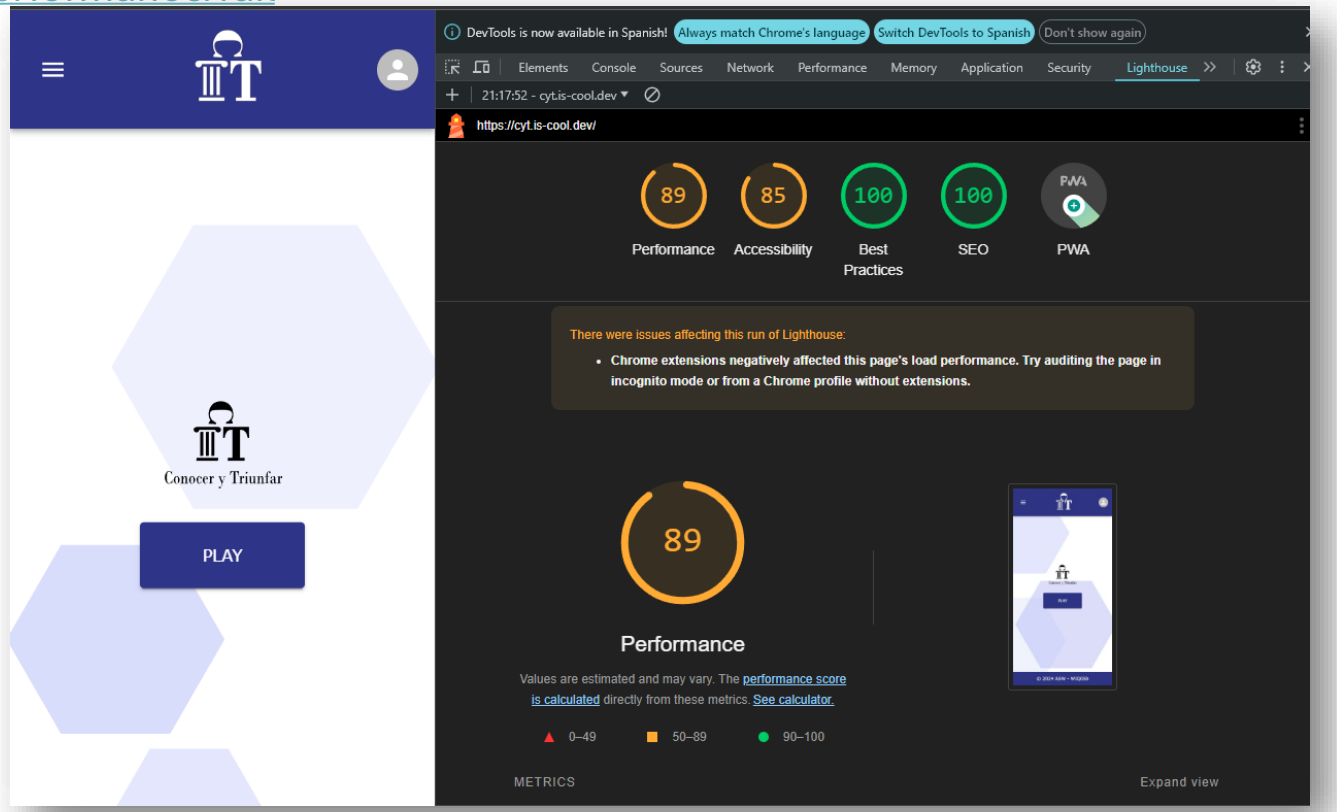
Otras herramientas de navegador

RAIL model (Response, Animation, Idle, Load)

https://developers.google.com/web/fundamentals/performance/rail

https://webpagetest.org/easy

Lighthouse (incluido en Chrome)



Monitorización en servidor

Las plataformas en la nube brindan soluciones de monitoreo

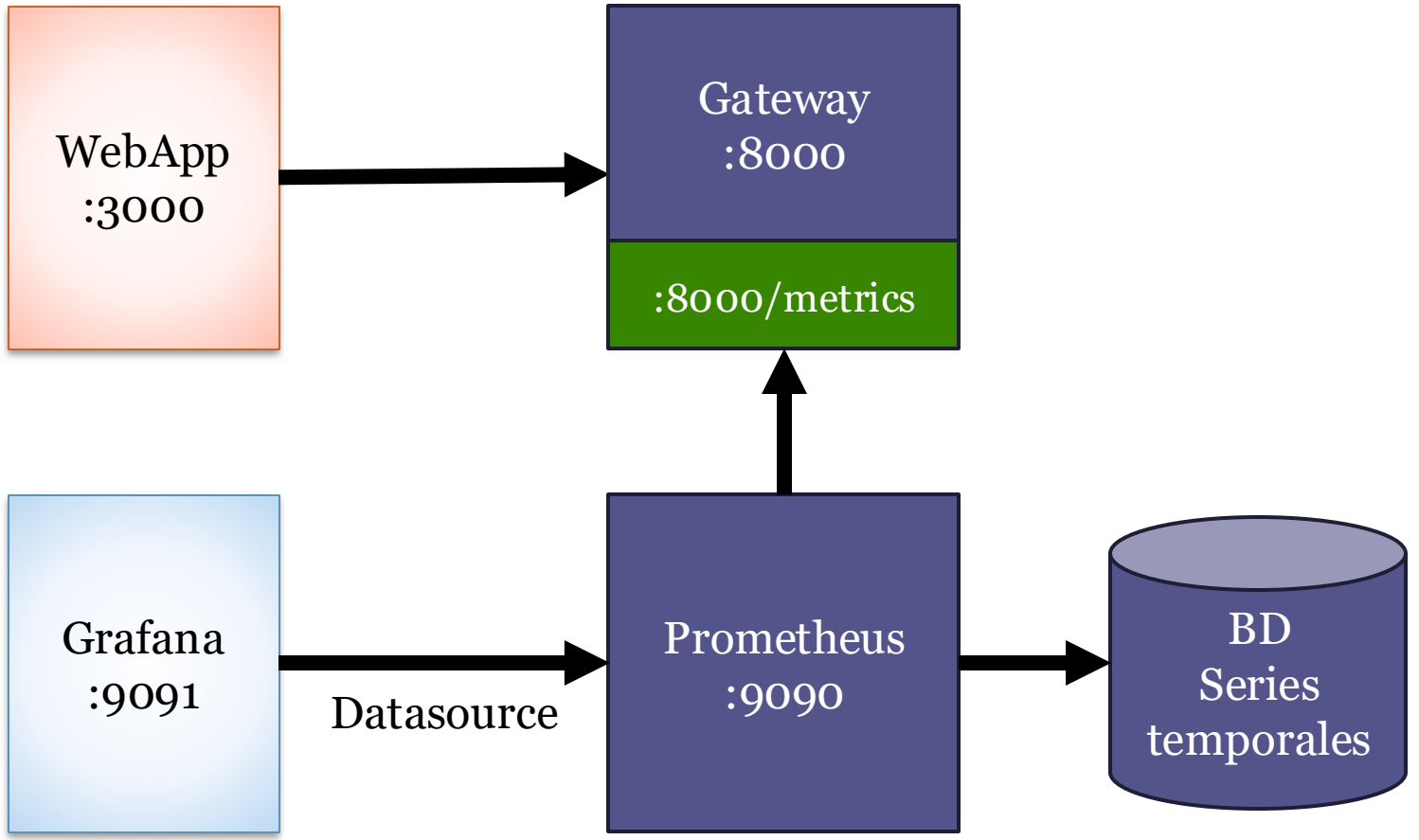
- También disponible en Google Cloud, Amazon AWS, Alibaba Cloud…
- En el caso de Azure: Azure Monitor

Aunque también existen soluciones de terceros

- Prometheus, Graphite, Grafana, Datadog, Nagios, Sensu, …

Usaremos: **Prometheus** y **Grafana**

- **Grafana**: Visualizar datos. Permite crear, explorar y compartir tableros y cuadrod de mandos
  - Suele utilizarse en combinación con Prometheus
- **Prometheus**: servidor de almacenamiento de bases de datos de series temporales
  - **Modelo de datos multidimensional**
  - **Lenguaje flexible de consultas**
  - **Nodos autónomos de servidor único**
  - **Configuración estática**


WebApp
:3000
Gateway
:8000
:8000/metrics
Grafana
:9091
Datasource
Prometheus
:9090
BD
Series
temporales

Monitorización en servidor

- Se necesita librería que pueda extraer métricas del servicio (e.g. Gatewayservice)
  1. Instalar el cliente

     npm install prom-client express-prom-bundle
  2. Modificar *gatewayservice/gateway-service.js*

```
const metricsMiddleware = promBundle({includeMethod: true});
app.use(metricsMiddleware);
```

  3. Si se lanza usersservice, en */metrics* se pueden ver datos de fila que Prometheus puede almacenar y Grafana puede usar para visualizar
     Se puede elegir qué métrica medir [doc]

Monitorización en servidor

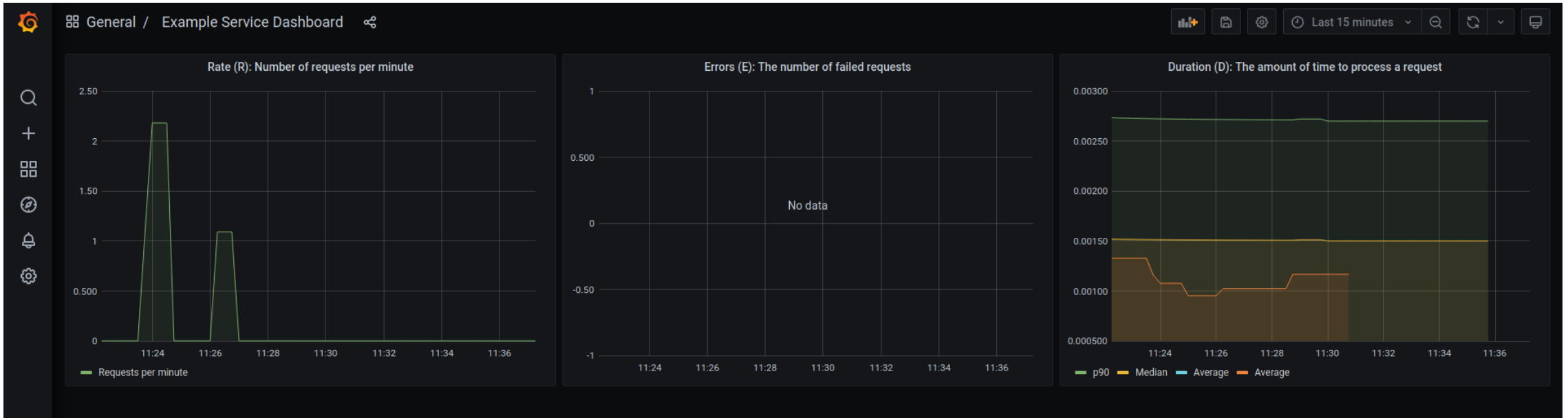
- Prometheus
  - Prometheus recupera datos expuestos por el servicio (e.g. userservice) y los almacena en BD de series temporales para que Grafana pueda visualizar.
  - Se utiliza una imagen docker [prom/prometheus] que se puede configurar a través de archivo

```
global:
  scrape_interval: 5s

scrape_configs:
  - job_name: 'users-service'
    static_configs:
      - targets: ['users:3000']
```

Monitorización en servidor

- Grafana
  - Grafana puede usar Prometheus como fuente de datos
  - Se usa una imagen docker para su ejecución [grafana/grafana]
  - Se puede configurar datasource y dashboard (gráficos a visualizar)



Ejemplos de cuadros de mandos reales

- https://grafana.wikimedia.org/

Referencias

- Monitorización y Profiling
  - Get Started With Analyzing Runtime Performance
    https://developers.google.com/web/tools/chrome-devtools/evaluate-performance/
  - How to Use the Timeline Tool
    https://developers.google.com/web/tools/chrome-devtools/evaluate-performance timeline-tool#profile-js
  - Otro Ejemplo
    https://github.com/coder-society/nodejs-application-monitoring-with-prometheus-and-grafana