ARQUITECTURA DEL SOFTWARE

Escuela de Ingeniería Informática

Universidad de Oviedo

Laboratorio 12

Monitorización
Observabilidad

2024-25

Jose Emilio Labra Gayo
Pablo González
Irene Cid Rico
Diego Martín Fernández

Monitorización y Profiling

- **Monitorizar:** Observar comportamiento de un software
  - Cuadros de mando
  - Habitualmente, después del despliegue
- **Profiling (caracterizar)**: Medir rendimiento de un software mientras se ejecuta
  - Identificar partes que contribuyen a un problema
  - Mostrar dónde centrar los esfuerzos para mejorar rendimiento
  - Suele hacerse antes del despliegue

Monitorización y Profiling

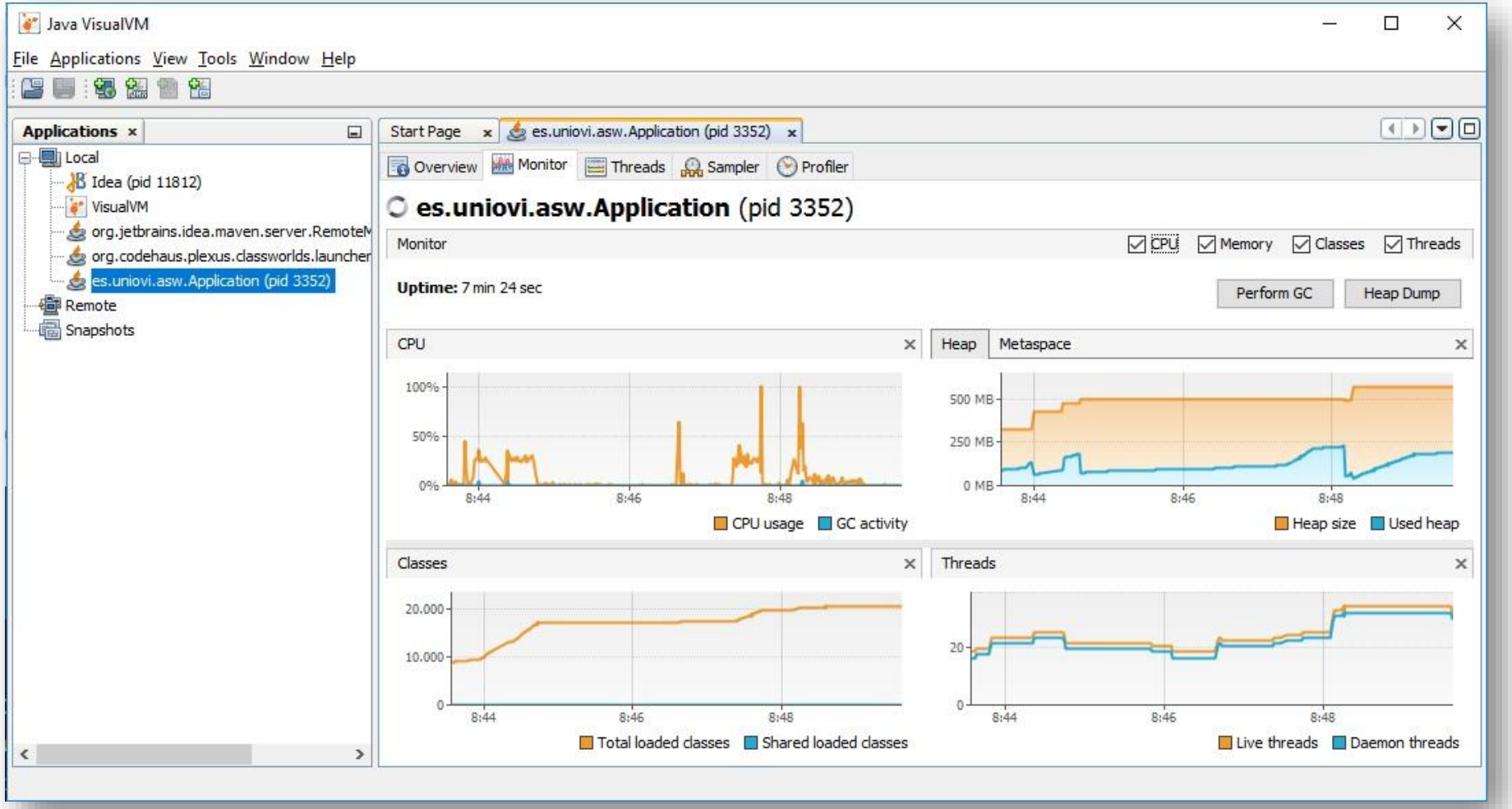
Monitorizar una aplicación mientras se ejecuta
Registrar uso de CPU, memoria, hilos, etc.

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

Herramientas de servidor:
- JVisualVM, JProfiler, YourKit, Jconsole, etc.
- Graphite, Datdog, Prometheus, Graphana

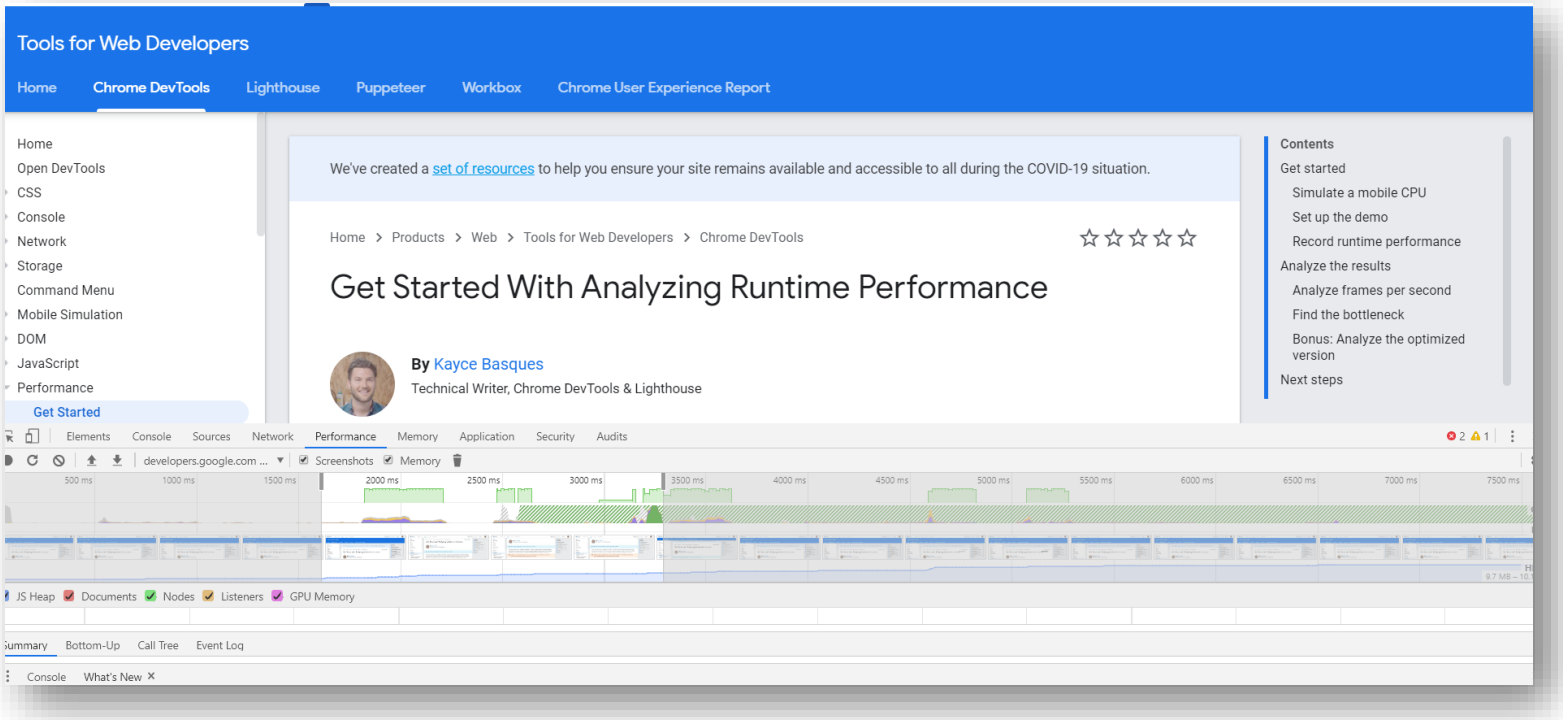
VisualVM
- https://visualvm.github.io/
- Ya está instalada con el JDK: `jvisualvm`

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.

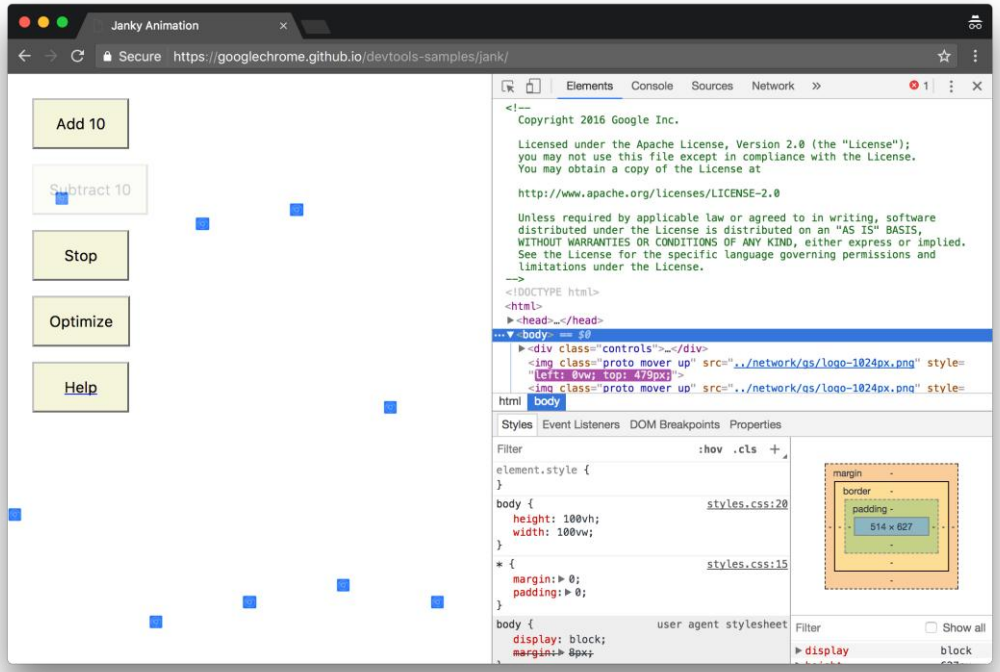
Chrome DevTools

Windows, Linux: Control+Shift+I

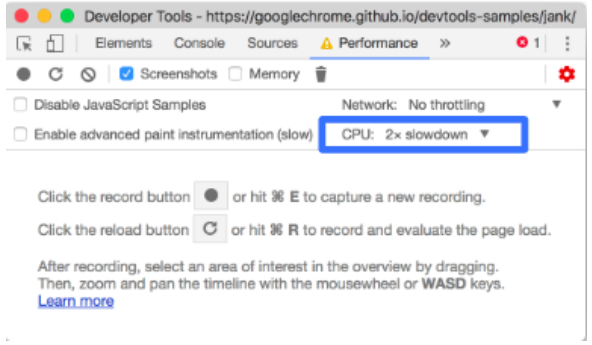
Mac: Command+Option+I

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

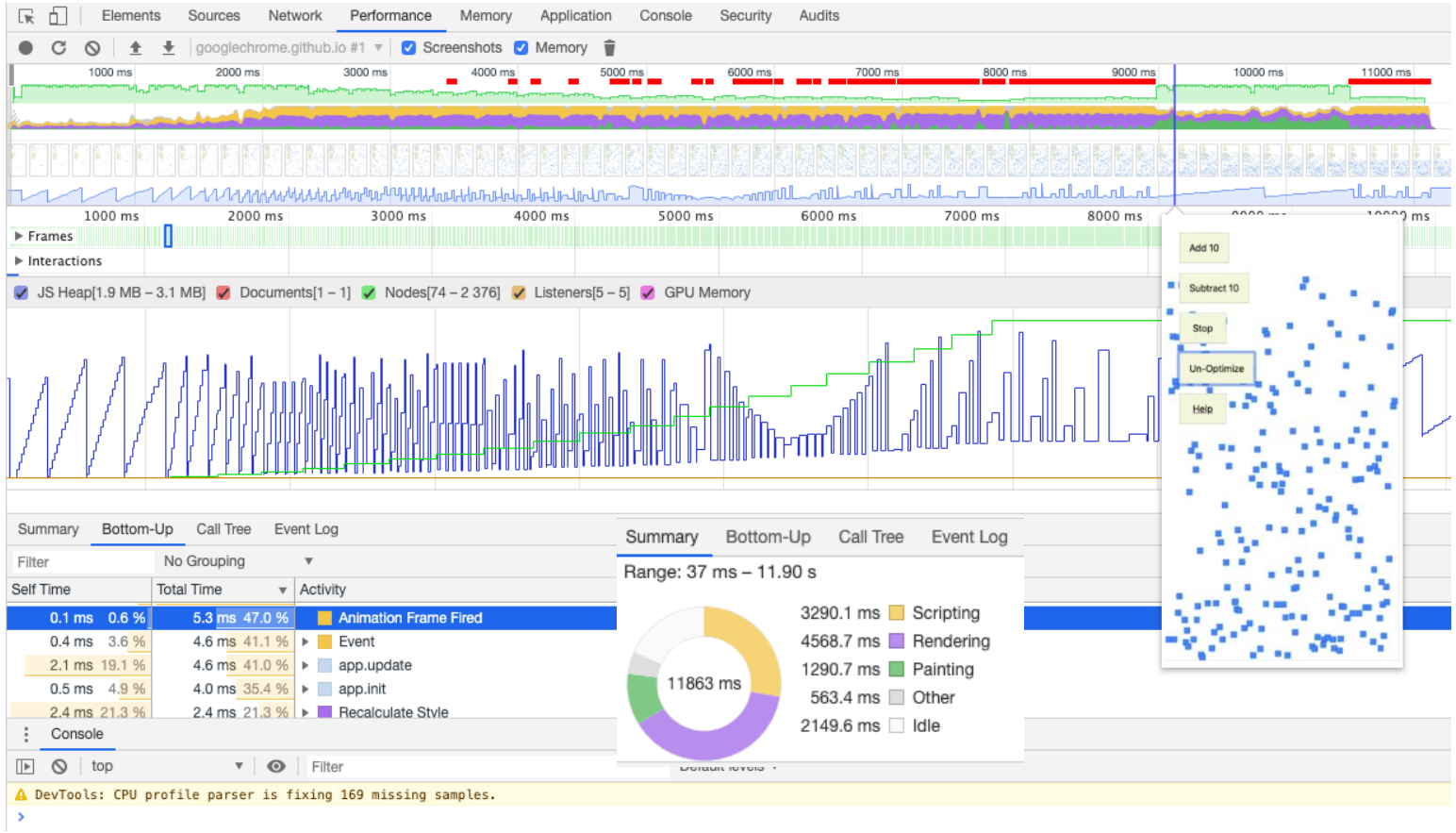
Ejemplo: Google Chrome

Profile result:

Frames por segundo ➡️

CPU ➡️

Cuellos de botella ➡️



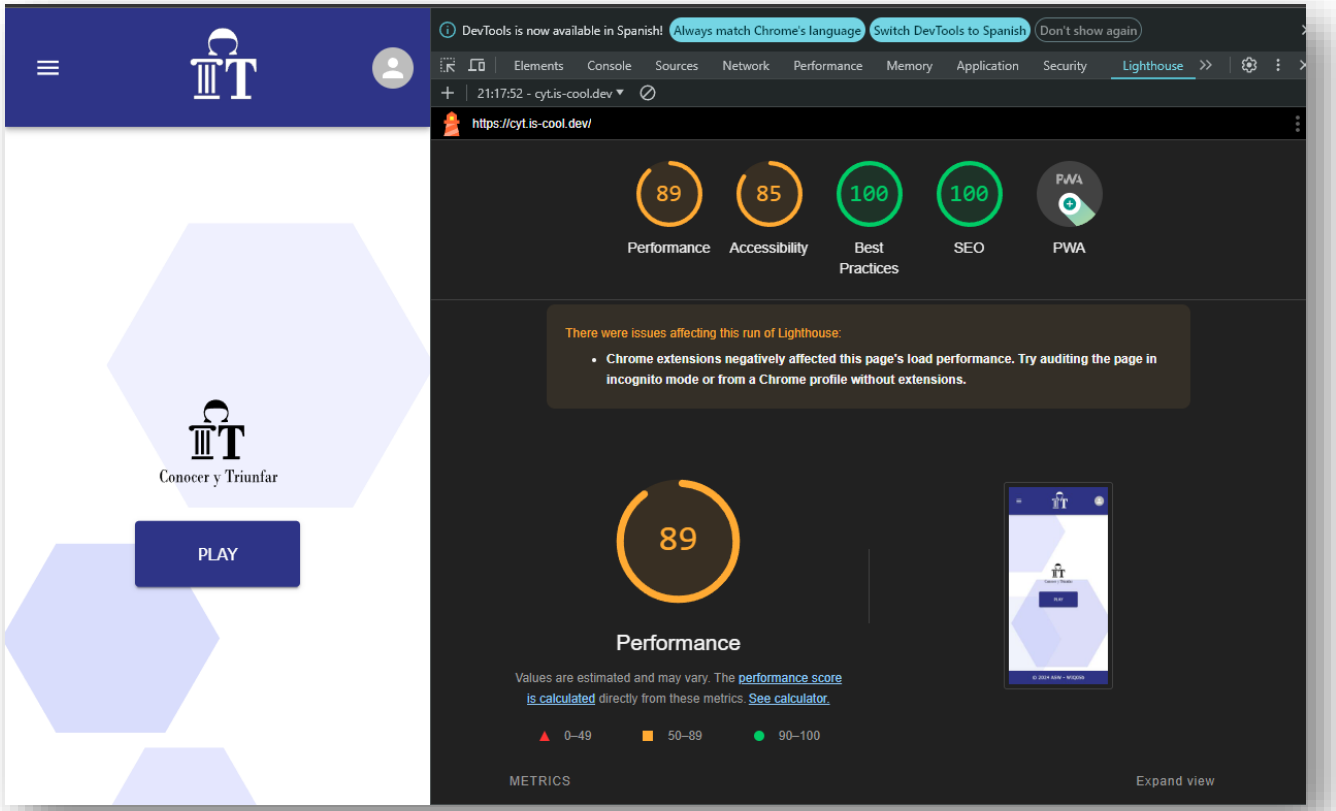
Otras herramientas de navegador

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 **Graphana**

- wichat_0:
  https://github.com/Arquisoft/wichat_0/blob/master/gatewayservice/README.md

- **Prometheus**: servidor de almacenamiento de datos en series de tiempo
  - **Modelo de datos multidimensional**
  - **Lenguaje flexible de consultas**
  - **Nodos autónomos de servidor único**
  - **Configuración estática**
- **Grafana**: Visualización de datos. Permite crear, explorar y compartir tableros

Monitorización en servidor

- Necesitamos una biblioteca que pueda extraer algunas métricas de nuestro servicio (e.g. Gatewayservice)
  1. Instalar el cliente

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

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

  3. Si lanzamos el gatewayservice, en */metrics* podremos ver algunos datos de fila que Graphana usaría para trazar los gráficos.
     Podemos elegir qué métrica medir [doc]

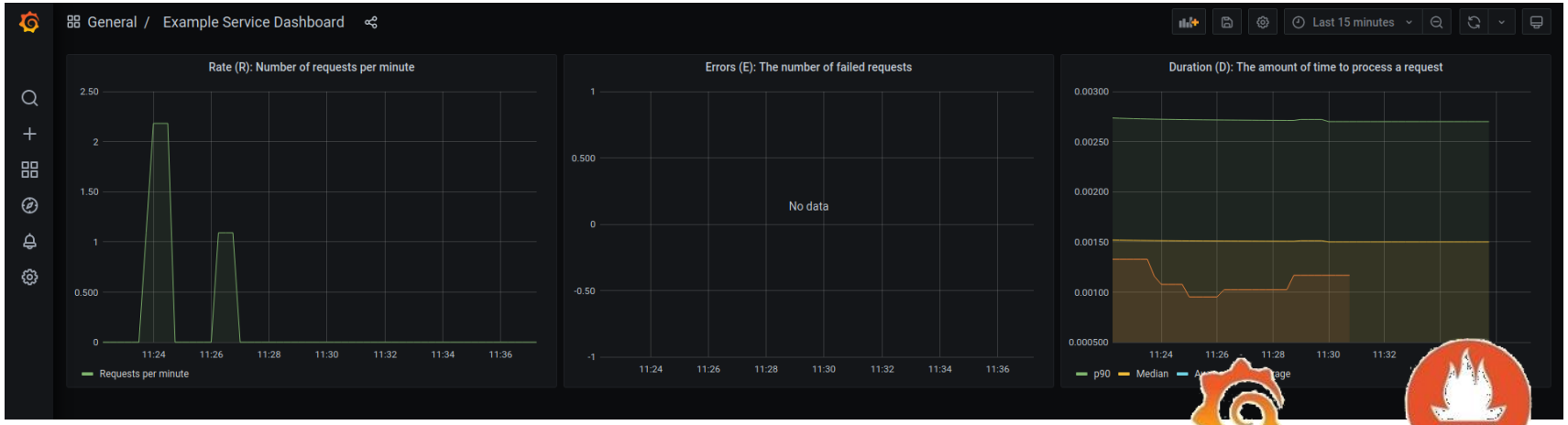
Monitorización en servidor

- Graphana no puede usar esta información directamente, necesita Prometheus
  - Prometheus recuperará los datos expuestos por el servicio (e.g. gatewayservice) y los almacenará para que Grafana pueda consumirlos.
  - Trabajaremos con una docker image [prom/prometheus] que se puede configurar a través de un solo archivo

```yaml
global:
  scrape_interval: 5s
scrape_configs:
  - job_name: "example-nodejs-app"
    static_configs:
      - targets: ["gatewayservice:8000"]
```

Monitorización en servidor

- Como configurar Graphana
  - Graphana usará Prometheus como fuente de datos
  - Tenemos una docker image para ejecutarlo [grafana/grafana]
  - Nosotros necesitamos configurar datasource y el dashboard (gráficos a visualizar)



Grafana Prometheus

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