Arquitectura del Tablero de Monitoreo de Apertura de Datos Abiertos

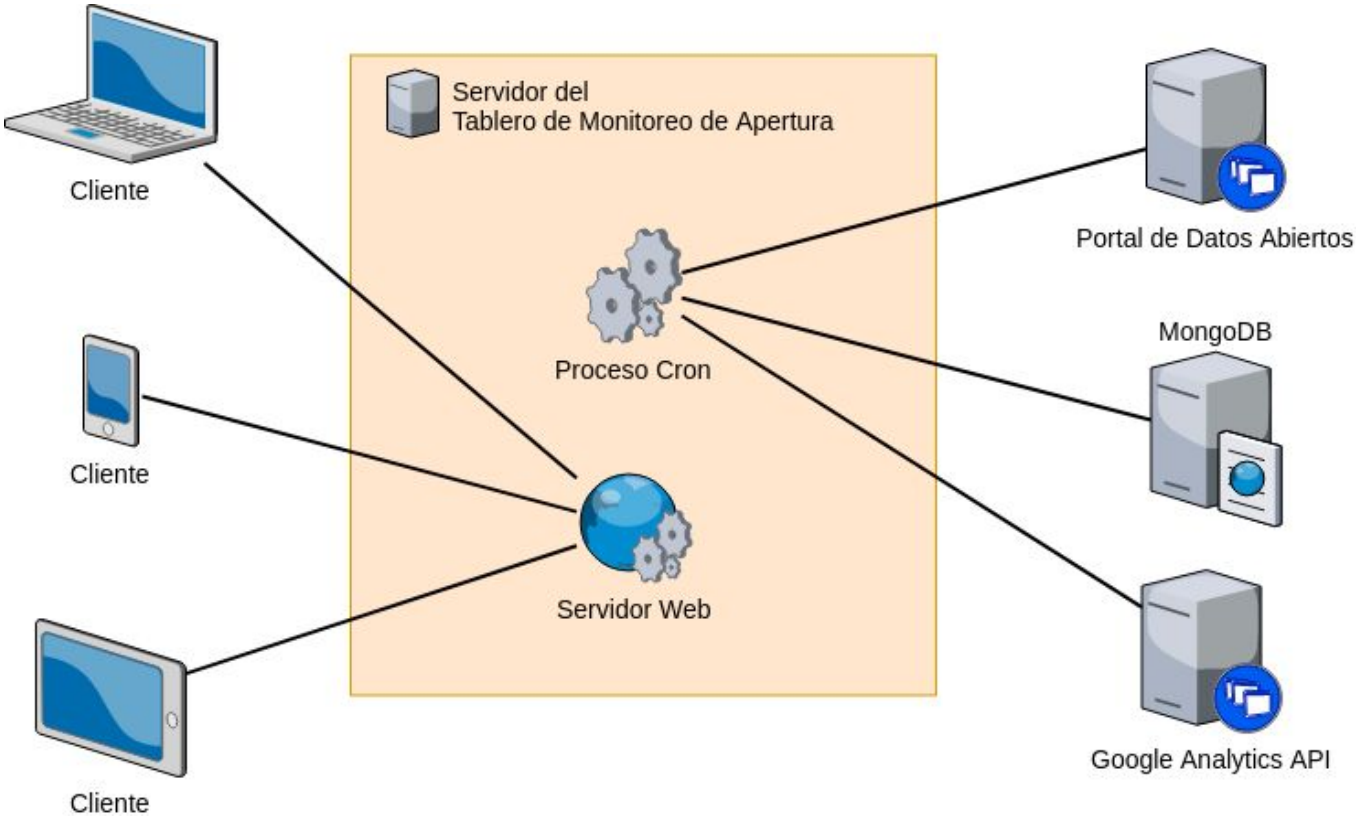
Casos de uso

El propósito del sistema es proveer una serie de indicadores de la actividad del portal de datos abiertos.
Esto permitirá tanto a el equipo de datos abiertos como a las áreas de gobierno monitorear la cantidad de datasets y de recursos subidos por organización y categoría así como también el tráfico en el portal mediante visualizaciones o acceso al archivo en formato csv que contiene el histórico de los indicadores.
Los indicadores son calculados a partir de los metadatos expuestos por el portal de datos abiertos y por la API de Google Analytics, el tablero automatiza el armado de los indicadores y facilita su lectura con visualizaciones.

Elementos de la aplicación

La aplicación consiste en un servidor web y un proceso cron y se sirve de servicios externos como una base de datos MongoDB, el Portal de Datos Abiertos y Google Analytics.

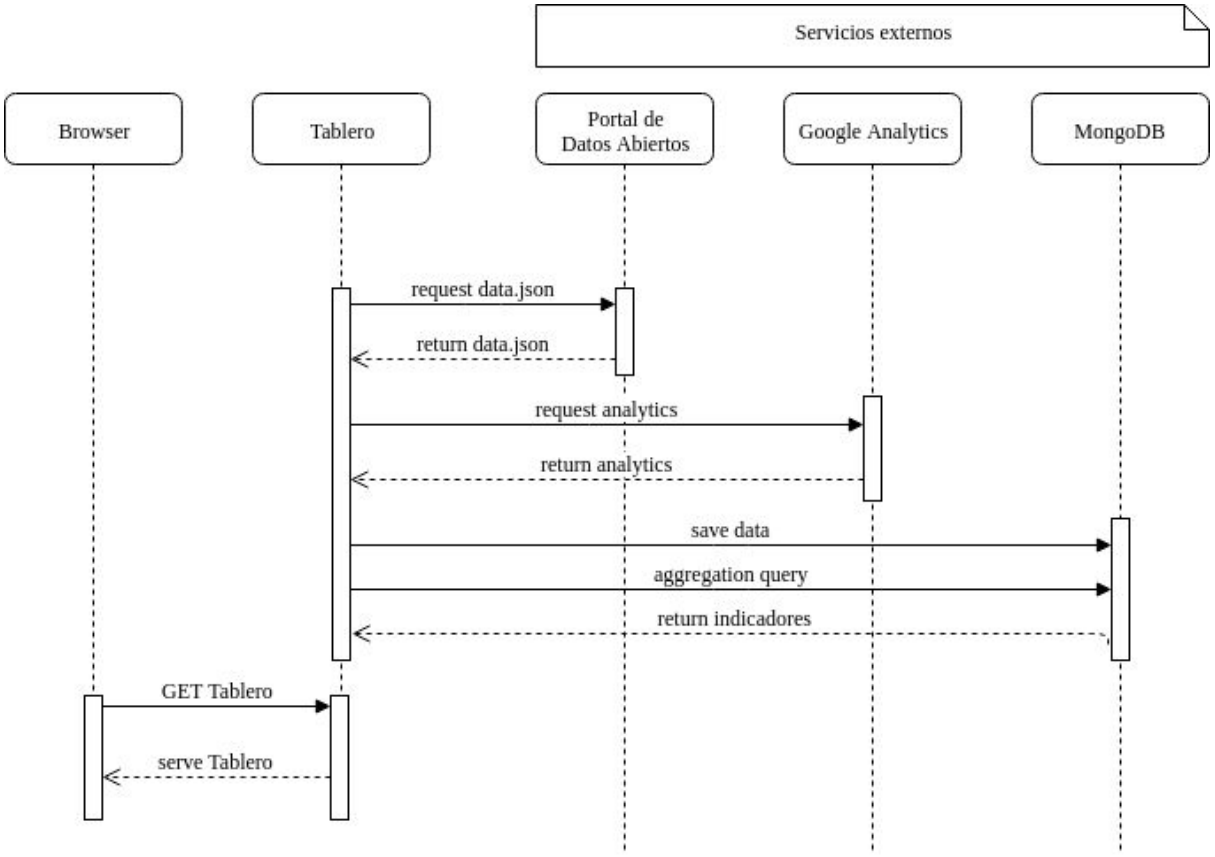
A continuación un diagrama de estos elementos y sus comunicaciones.



Procesos de la aplicación

El servidor web accede a un archivo json ubicado en el mismo servidor de la aplicación, junto con el código, para armar las visualizaciones y servir el sitio web del tablero.
Una vez iniciada la aplicación un proceso cron es iniciado, este proceso corre diariamente y se comunica con el portal de datos abiertos para acceder a un archivo json y al servicio de Google Analytics (GA) para acceder a métricas del tráfico del portal de datos abiertos, las métricas son combinadas con el data.json en base a las URLs de los datasets, luego se guarda el data.json enriquecido con métricas de GA en la base de datos, también se guardan las búsquedas realizadas en el portal, luego de guardar los datos del día, se consulta a la base de datos usando agregaciones que crearán los indicadores, estos se guardan en un archivo json localmente, cada día se regenera este archivo.

A continuación un diagrama del flujo de los procesos.



Pantallas

Visualmente el sistema consiste en tres tableros muy similares estructuralmente.
El layout obedece los lineamientos de la guía de estilos oficial BAstrap, por lo tanto se incluyen las librerías necesarias para ésto.

Las visualizaciones son interactivas, esto se logra con la librería Plotly, todos los componentes menos el header y el footer utilizan React.js. Tanto Plotly como React.js son usados indirectamente ya que son implementados por el framework para el lenguaje Python, Dash.

Backend

La aplicación, que corre en un sistema operativo CentOS 7.
Para armar el tablero, se consulta el archivo json y se utiliza para extraer los datos necesarios para cada visualización, las visualizaciones son manejadas por Dash un framework basado en Flask creado por los creadores de Plotly, una famosa librería de visualizaciones de datos en Python, Dash simplifica el uso de Plotly directamente desde el backend.
Se ejecuta un proceso en segundo plano que cada día ejecutará la rutina explicada anteriormente.

Despliegue

El proceso de despliegue se automatiza utilizando Docker.
En el Dockerfile provisto en el repositorio se parte de una imagen base oficial CentOS 7, luego se procede a instalar las dependencias, estas son, Cron y Python 3.6, luego se instalan las dependencias de los paquetes Python definidas en el archivo requirements.txt también provisto en el repositorio. Finalmente al ser ejecutado el container se inicializa el proceso cron y la aplicación web.
Las distintas entregas se harán usando versiones semánticas que estarán reflejadas tanto en el repositorio Git como en las imágenes Docker.

La configuración se encuentra en el archivo config.yml (se proveen ejemplos de los valores en el archivo config.dist.yml) y precisa las siguientes variables:

Variable	Descripción
ga	Objeto con configuración para GA
mongo_url	URI de conexión a la base de datos MongoDB
data_json	URL del archivo data.json

Referencias

- Repositorio público (Github)
- Repositorio oficial ASI

- Dash