# Tecnología Big Data

**EQUIPO: 4**
Yasmin Mohamed, Vanessa Casais, Juan Ramón Rodríguez y Santiago B. Pérez

2 de Febrero de 2019

## Resumen

En este documento se expone una posible solución mediante tecnología Big Data a las necesidades de una empresa de gestión de peajes. Para ello hemos propuesto una arquitectura sencilla de alto nivel que utiliza como piezas fundamentales diferentes aplicaciones de la plataforma **Google Cloud** y hemos evaluado los costes que dicha arquitectura supondría para la empresa.

## 1. Alcance del proyecto y supuestos

Se trataría de una empresa interesada en aplicar nuevas tecnologías en la gestión de una autopista con los siguientes objetivos:

- Recopilar diariamente los datos almacenados en los servidores de los peajes.
- Implantar un carril de 'velocidad mínima asegurada' de 80 km/h, que se garantizará regulando el precio en tiempo real en base a los datos procedentes de distintos sensores.
- Visualizar y analizar los datos en tiempo real o cercano a tiempo real mediante cuadros de mando accesibles desde una app.
- Obtener en tiempo real el estado de los dispositivos, para garantizar su funcionamiento y detectar averías.
- Disponer de un histórico de todos los datos recopilados, accesible e ilimitado.
- Procesar y analizar de forma masiva y en ventanas de tiempo dicha información histórica.

Para alcanzar estos objetivos nos colocamos en el supuesto de una autopista ubicada en una gran ciudad española, que cuenta con sensores de velocidad, de condiciones climáticas, contadores de vehículos, servidores en los peajes y en general con toda la infraestructura necesaria para llevar a cabo el proyecto.

## 2. Flujo de los datos

En primer lugar hemos separado dos procesos en la ingesta de datos. Por un lado tenemos los datos procedentes de los servidores, que se enviarán a la nube de forma diaría, constituyendo así un proceso tipo *batch*. Por otro lado tenemos los datos de los sensores, que deben ser ingeridos inmediatamente, tratándose de un proceso tipo *streaming*.

Una vez ingestados los datos estos deben ser procesados, separando la información sobre el estado de las máquinas, de la información sobre el estado de la autopista, nutriendo esta y acoplándola a algún formato (e.g. json). Sobre esta información ejecutaremos automáticamente algún algoritmo que nos permita determinar el precio y la información sobre el estado de los dispositivos se volcará de forma directa en un panel tipo TUI (*Text User Interface*) para detectar averías.

Toda la información procente del procesado se irá insertando en un *Data Warehouse* que será al que los empleados lanzarán sus consultas. Con el fin de garantizar (más allá de las garantías que ofrece Google) la permanencia de los datos, haremos un segundo volcado que emplearemos como *Back Up*.

Finalmente con la información del *Data Warehouse* realizaremos unos paneles informativos que serán accesibles desde una app, de forma que los usuarios de la autopista puedan consultarlos, y los empleados puedan además manipularlos.

## 3. Arquitectura propuesta

Con el fin de conseguir los objetivos marcados y seguir el flujo de datos indicado hemos propuesto la siguiente arquitectura.

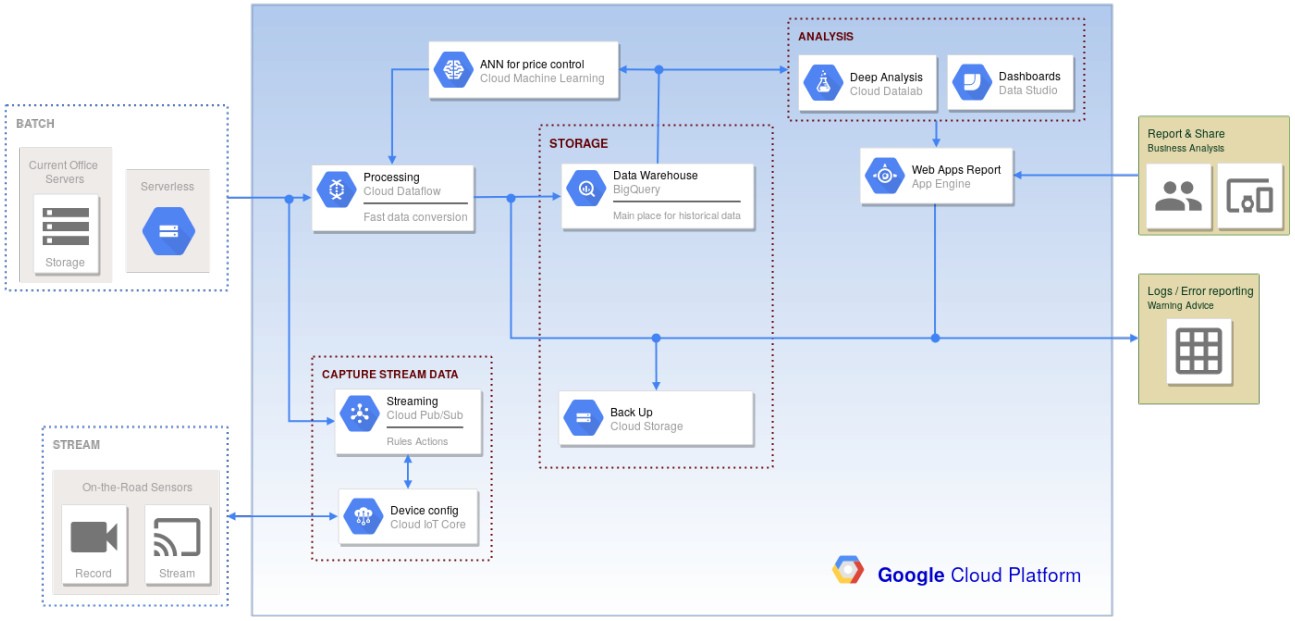


Figura 1: Arquitectura de Google Cloud Platform

En primer lugar, aunque conservamos la idea de que los datos que se encuentran en los servidores de los peajes lleguen en *batch* a la nube, también hemos propuesto una arquitectura alternativa *serverless* en la que este *batch* procedería de **Cloud Storage**.

Para los datos recibidos en *stream* utilizaremos el servicio **Cloud IoT Core** que conecta, monitorea y gestiona de forma segura los dispositivos y sensores encargados de generar los datos en tiempo real a través de un protocolo MQTT. **Cloud IoT Core** enviará los datos, que serán recibidos por **Cloud Pub/Sub** dentro de los diferentes temas creados, dependiendo del tipo de sensor, del tipo de información (mediciones, estados del dispositivo, ...), de su ubicación, etc. **Cloud Pub/Sub** es capaz de recibir mensajes en variedad de formatos y escalar de manera automática, lo que garantiza la buena recepción de los datos aunque tengamos picos de carga. Sabemos que **Cloud Pub/Sub** no garantiza la perfecta ordenación temporal de los mensajes, pero esta información será, a priori, redundante, y podemos permitirnos cierto desorden dado que manejamos una escala de tiempo de respuesta de algunos segundos. Además **Cloud Pub/Sub** nos permite enviar información a los dispositivos, funcionalidad que emplearemos para publicar el precio calculado en los paneles de la autopista.

Una vez realizada la ingesta, los datos se procesan inmediatamente, para lo que emplearemos **Cloud Data-Flow**. Con esta herramienta podemos agregar los datos, nutrirlos (sin gran complejidad, para no entorpecer el proceso) y finalmente alimentar mediante integración contínua nuestro *Data Warehouse*, para el cual optamos por el servicio **Cloud BigQuery**, que será el encargado de almacenar la información consolidada y particionada de forma conveniente para su posterior análisis. Elegimos **BigQuery** como la mejor alternativa por ser compatible con SQL y por su capacidad de realizar consultas de hasta petabytes en los tiempos de respuesta que estamos manejando.

**Cloud Machine Learning** será el servicio elegido para nuestro modelo de aprendizaje estadístico, dando así respuesta a la funcionalidad de 'carril de velocidad mínima asegurada'. Para ello, el modelo se nutrirá del histórico de datos almacenado en **BigQuery** para su fase de entrenamiento y una vez entrenado será capaz de realizar predicciones en tiempo real, calculando el precio que será atractivo o disuasorio en función de las variables captadas por los sensores (clima, ocupación, velocidad, ...) tal que se garanticen esos 80 km/h. El precio calculado por nuestro modelo se envía a una cola de **Pub/Sub**. A partir de ahí, la predicción puede ser consumida por el dispositivo encargado de mostrar al conductor el precio a pagar.

*Mediante un mecaninsmo similar, podríamos crear otro modelo que ayude a detectar futuros problemas en los dispositivos de medición, anticipándonos a los problemas que ello pudiera generar y mejorando el funcionamiento de nuestra infraestructura, a la vez que aliviamos la carga de trabajo de los encargados de la detección de averías, sin embargo, esta mejora sobre el objetivo del proyecto no la hemos acometido a nivel de diseño.*

Finalmente la misma información que almacenamos en **BigQuery** la almacenaremos también en **Cloud Data Storage Coldline**, comprimida en ficheros binarios y en otra región, con la intención de poder restituirnos en caso de desastre.

Llegados a este punto tenemos dos herramientas de análisis enfocadas a dos perfiles de trabajadores. Por un lado **Cloud Data Studio** que nos permitirá transformar nuestra información en resultados numéricos, tablas y gráficos que sean fácilmente comprendidos e interpretados por diferentes tipos de usuarios sin necesidad de que tengan un nivel analítico elevado. **Data Studio** nos permite generar estos *dashboards* de forma automática, ayudando al área de negocio a mejorar la toma de decisiones. Por otro lado disponemos de **Cloud DataLab** una herramienta más enfocada a los científicos de datos, que les permite manipular la información con toda la libertad que otorga un lenguaje de programación. Con ella podrán lanzar consultas a **BigQuery** y aplicar estadística avanzada para mejorar la información que se presente en los *dashboards*.

Con nuestros datos ya transformados en información valiosa y fácilmente legible, la colgamos en aplicaciones web alojadas en **Cloud AppEngine**. De esta forma simplificamos el acceso a la información y podemos discriminar a los trabajadores de los usuarios de la autopista, teniendo los primeros también acceso al reporting, análisis histórico, dispositivos averiado, etc. mientras que el acceso de los segundos se limite al estado del tráfico, retenciones, obras, accidentes, tiempos de recorrido, áreas de servicio, ofertas y servicios en ruta.
*Por supuesto una forma inmediata de mejorar la calidad de nuestro servicio sería exprimir la información que nos brinda Google sobre estas aplicaciones (control de logs, datos de navegación, etc.) utilizándola para nutrir los datos de uso de nuestra autopista y poder perfilar con mucha más precisión a nuestros clientes. No obstante esto excede el alcance de nuestro trabajo.*

# 4. Presupuesto

En último termino hemos realizado, empleando **Google Cloud Calculator**, lo que consideramos una estimación al alza de los posibles costes derivados de los recursos consumidos por esta arquitectura.

Suponemos un tráfico promedio diario de 40.000 vehículos que es el doble de la media estatal de 2015[1]. Estimamos la carga *batch* en formato .json con unos 15 campos asociados a cada vehículo que llegan diariamente a DataFlow. En cuanto al *streaming* consideramos 300 sensores que envían en torno a 40 bytes de datos por minuto.

$$40000\ \tfrac{vehiculos}{dia} \times 30\ \tfrac{dias}{mes} \times 15\ campos \times 35\ \tfrac{bytes}{campo} = 630000000 \tfrac{bytes}{mes} \approx 650 \tfrac{MB}{mes} \quad (1)$$

$$300\ sensores \times 40\ \tfrac{bytes}{minuto} \times 60\ \tfrac{minutos}{hora} \times 24\ \tfrac{horas}{dia} \times 30\ \tfrac{dias}{mes} = 518400000\ \tfrac{bytes}{mes} \approx 550\ \tfrac{MB}{mes} \quad (2)$$

Con estas estimaciones tenemos:

- Pub/Sub (550 MB) $\longrightarrow$ 0.00 €/mes
- IoT Core (550 MB) $\longrightarrow$ 1.18 €/mes
- DataFlow $\longrightarrow$ 5.88 €/mes
  - 3 x n1-standard-1 workers in Batch Mode
  - Region: Europe
  - Total vCPU Hours: 90
  - Total Memory Hours: 337.5 GB/Hours
  - PD Local Storage: 54 GB/hours
- DataFlow $\longrightarrow$ 165.67 €/mes
  - 3 x n1-standard-1 workers in Streaming Mode
  - Region: Europe
  - Total vCPU Hours: 2160
  - Total Memory Hours: 8100 GB/Hours
  - PD Local Storage: 1188 GB/hours

Suponiendo que los datos hayan multiplicado por 3 su tamaño, que las *queries* exigirán un procesado de 10TB (unas 120 consultas mensuales DIFERENTES sobre una historia de 2 años), solicitando 850GB de espacio (más de 20 años de historia) y considerando el mismo espacio para Cloud Storage (donde los datos deberían ocupar menos al estar comprimidos) tenemos:

- BigQuery $\longrightarrow$ 54.09 €/mes
  - Location: Europe
  - Storage 850 GB
  - Streaming Inserts 550 MB
  - Queries 10 TB
- Cloud Storage Coldline $\longrightarrow$ 5.21 €/mes
  - Finland
  - Regional storage: 850 GB

[1] `http://www.fomento.gob.es/AZ.BBMF.Web/documentacion/pdf/D-050_2015.pdf`

Finalmente al módulo de Machine Learning y a App Engine les asignamos parámetros coherentes:

- Cloud Machine Learning ⟶ 1.17 €/mes
  - Region: Europe
  - ML Training Units: 2.473
  - Job run time: 60 minutes
- App Engine ⟶ 2198.97 €/mes
  - Belgium
  - Instance Hours: 51100 per month

**DataLab** y **DataStudio** no están disponibles en la calculadora, **DataStudio** es un servicio gratuito y por lo tanto lo excluímos del presupuesto, no así **DataLab** que hemos leído que requiere un disco persistente de al menos 220 GB sobre una entidad de **Compute Engine** para funcionar, así que para presupuestarlo hemos incluido una máquina virtual.

- Load Balancing (global) ⟶ 23.96 €/mes
  - Belgium
  - Forwarding rules: 2
  - Network ingress: 2,000 GB
- Persistent Disk ⟶ 74.36 €/mes
  - Belgium
  - SSD Provisioned Space: 500 GB

**PRESUPUESTO TOTAL: 2530.49 €/mes**. Suponiendo que consume un 5 % del presupuesto mensual de la empresa y el precio del peaje es 2€ (lo cual sería una ganga) la empresa sacaría un beneficio de 352,682.40€ anuales.