Examen Data Engineering (EA)

Canseco García Edgar Jesús

.

Julio de 2021

Ejercicio 3

1. Propuesta de orquestador en GCP

La propuesta técnica es la siguiente:

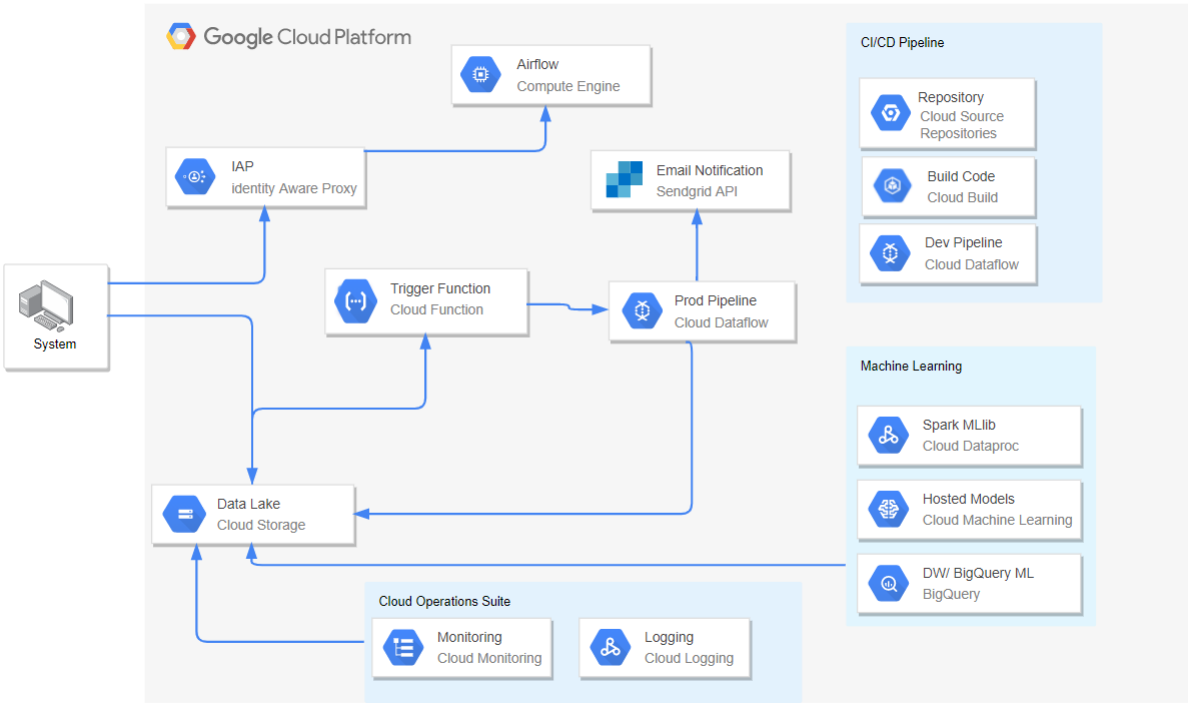

Figura 1- Arquitectura en GCP

- El sistema enviará los archivos Parquet en el folder correpondiente al bucket con el Servicio de Google Cloud Storage, en este punto es importante configurar los ACL en los bucket, para tener un control granular acerca de que permisos tiene cada archivo y folder dentro de cada bucket, ya que su acceso no debe ser público, y la lectura, modificación, escritura y eliminación de los objetos debe ser controlada con esos permisos. Además que es una buena práctica configurar un ciclo de vida para los archivos, por ejemplo, un archivo cargado después de 3 meses puede que solo se acceda una vez al mes, por lo que pasar a un tipo de almacenamiento nearline o coldline ayudará a que los costos de almacenamiento sean menores. Al ser GCS el data lake a utilizar, es importante notar que es conectada a múltiples servicios que se explicarán a continuación.

- Aquí hay otro punto importante a considerar, y es la complejidad de cómo se desea activar la ETL y el presupuesto que se tenga para su implementación. Por ejemplo, si la ETL solo se va a activar al finalizar la carga de un archivo parquet, es suficiente

con programar una Cloud Function que se active mediante un trigger cuando la carga de un archivo a un determinado bucket se termine, activando así un job en Dataflow (siendo este servicio la ETL). sí además de esto se requiere calendarizar cuando se va a ejecutar la ETL, puede usarse el servicio de cloud Scheduler para programar la activación del job en dataflow. Si bien con esto podemos activar la ETL,perdemos muchas características que tendríamos con un orquestador como Airflow, que además de realizar las dos acciones anteriores, puede ser muy útil en otras tareas, como la monitorización, programar tareas más complejas mediante DAGs, y la configuración de alertas sobre esos Dags programados, por lo que una buena opción es hacer uso de esa orquestación con Airflow, desplegándolo en una VM, asegurándose de las buenas prácticas de seguridad, configurando una red y subnets en el proyecto, las reglas de firewall necesarias, y conectarse a la máquina mediante un túnel con IAP.

- Dataflow es la herramienta ETL, la cual es escalable de acuerdo a la cantidad de tareas a procesar. Funcionando con el framework de apache Beam, las estructuras de PCollection son distribuidas y capaz de escalar este procesamiento a diferentes nodos optimizando la carga, por lo que de acuerdo al presupuesto, se limitará el número máximo de nodos. Es necesario en este proceso tener un archivo en GCS configurado como tabla perimétrica, el cual contenga un punto de control para saber cuál fue la última data procesada. Hay dos opciones de donde cargar esta data procesada, y es solo dejarla en un bucket en GCS, o mediante Airflow, hacer que BQ también cargue está data, lo cuál será útl para analizarla mediante un lenguaje SQL, o incluso hacer pruebas de concepto rápidas con BigQueryML, esto dependerá de más de las necesidades futuras pero es importate tenerlo considerado. Si al procesar esta información en Dataflow, hay estructuras o datos que no cumplen las especificaciones esperadas para el modelo, con un servicio como Sendgrid se puede enviar un correo electrónico, para enterarse rápidamente que es lo que está pasando y actuar, este servicio de correo da más 1000 envíos gratis al mes, por lo que es una opción viable debido a la restricción de dinero. En la parte de trazabilidad es importante en Dataflow mantener una cultura DevOps para hacer posibles pipelines CI/CD, y aquí los servicios importantes son 2: Cloud Source Repositories mantendrá de forma privada y versionadas el código aplicado a la ETL, y junto a Cloud Buid ayudará a construír, testear y lanzar las versiones de forma serverless.

- Si bien es posible que en Dataflow se hagan las transformaciones y cálculos en el Proceso de ETL, dependiendo de la complejidad y caso de uso de los datos para las variables generadas, o análisis con código, es posible aprovisionar una Jupyter Lab

con Dataproc y trabajar con apache Spark. Dataproc permite gestionar la cantidad de nodos maestros y worker para esto, por lo que si se necesita ajustar mucho el presupuesto, es posible lanzar solo un nodo maestro y así trabajar con una Notebook y los servicios integrados de GCP ya mencionados. En pasos posteriores o futuros también este modelo puede ser lanzado el modelo en Cloud Machine Learning, aunque si se requiere una ciclo completo de MLOps para esta tarea, sería evaluar la opción de la nueva suite de Vertex y hacer un pipeline completo, todo dependerá de las características propias del alcance y propósito del proyecto.

- Como las herramientas para monitorear y Loggs, se encuentra la suite de operaciones de GCP que permitirá establecer métricas y analizar logs acerca de los servicios utilizados, un ejemplo muy útil sería saber si el job de Dataflow resultó en éxito, y en que tiempo, y notificar si falla el job.

- Es importante mantener los service accounts y roles asignados al proyecto, siguiendo el principio del menor privilegio, todo esto con Cloud IAM, y a sí tener en control quién puede ver acceder a la data, y que se puede hacer con ella, por ejemplo.

- También se puede establecer un Budget a nivel proyecto para tener un presupuesto de cuanto es lo estimado al mes de gastar, y saber cuando nos acercamos a ese presupuesto máximo, además de que con el paso del tiempo, y de acuerdo a nuestras cargas de trabajo con lso servicios, gcp puede hacer predicciones de gastos futuros.

- Como medida adicional, podemos activar Data Catalog para tener una mejor versión de la metadata y así un control y entendimiento de los datos más apropiados, esto será más útil si también se ocupan servicios como BigQuery

2.- Propuesta técnica

La arquitectura propuesta es la siguiente:

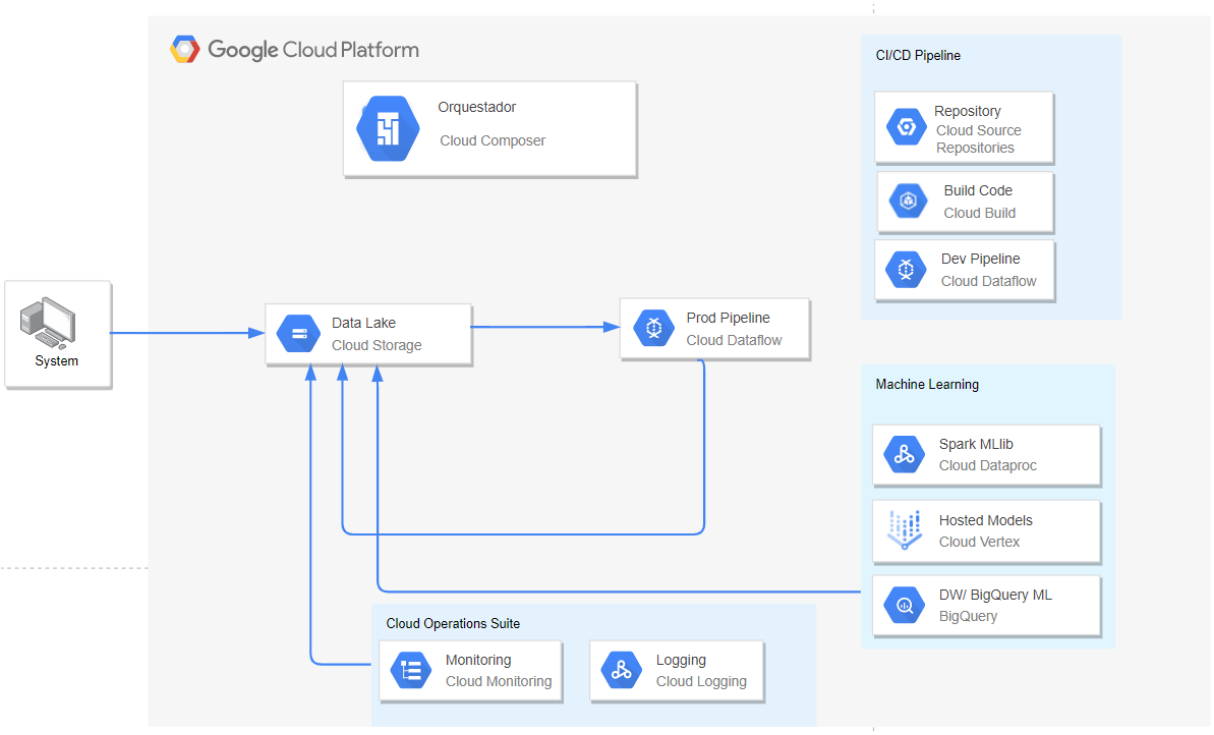

Figura 2- Arquitectura en GCP

Realmente la propuesta no cambia demasiado a la arquitectura descrito en la parte 1 con sus buenas prácticas, incluso parece más simplificada, y es por el uso de Composer, ya que permite un servicio administrado en GCP de Airflow, en dónde se tiene también una suite de herramientas para desplegar estos Dags, monitorearlos, etc, además que está integrado con mucho de los servicios, por lo que la logística de orquestar se reduce a una estancia de Composer, configurar correctamente el scope de los demás servicios a comunicarse y escribir los DAGs, ya que las instancias de kubernetes son manejadas por GCP.

También se quitarían las limitaciones en Dataflow, habilitando siempre el escalamiento al flujo de trabajo. También es útil hacer que al trabajar con Dataproc, se habilite un nodo maestro y los worker sean escalables, y habilitar GPU para la eficiencia en tareas a Spark.

En la parte del Modelo, es recomendable trabajarlo con la nueva plataforma de Vertex y ver pensando en la construcción de un pipeline MLOps.

3.- Logro en 2 meses

Primero definir el alcance exacto del proyecto, estableciendo métricas claras y restricciones del proyecto, para estar los 3 con el mismo objetivo claro de alcance y centrarse en eso. También definir un presupuesto estimado de la cantidad que se gastará en OPEX para estos servicios de las nubes, y entender de mejor manera como se van a definir y operacionalizar los serivicos.

Adaptando una metodología de desarrollo ágil, con su marco de trabajo y buenas prácticas, como primer sprint es definir como va a ser la arquitectura final y tener un documento de consulta en base a ello, esto no debe pasar la semana de trabajo.

Una vez aplicada la metodología, se pueden estar dividiendo tareas especificas de acuerdo a los roles, trabajando muy a la mano los tres, en donde el ingeniero de datos estará enfocándose en hacer la parte funcional en Dataflow o el servicio ETL elegido, además de provisionar la infraestructura con los servicios con los que interactúa ese Pipeline, ingeniero en site reliability debe estar enfocado en provisionar los pipelines para un pipeline en CI/CD, monitorear los servicios utilizados y administrar correctamente el orquestador con Airflow, y el científico de datos en todo lo encargado al análisis y calidad de la data entregada, además de todo el proceso relacionado de entrenar el modelo en GCP, saber la calidad y empezar la visualización de futuros Pipelines con MLOps.

Es importante definir un documento de pruebas de calidad de las diferentes tareas, que alcance van a tener las pruebas, y por la cantidad limitada de personas, ir planificando que tipos de pruebas se deben hacer, y cuáles son los mínimos alcances que se desean alcanzar

Suponienod que se elige Dataflow, al cabo de dos meses, se puede tener una primer versión en producción de una arquitectura que permita realizar el proceso de ETL, con un trigger cuando llegan datos y de manera programada a una determinada hora del día, o varias veces al día, dependiendo del alcance del proyecto, además de contar con una primer versión de un pipeline para la automatización de la construcción de templates en Dataflow, y una primer versión del modelo corriendo en Cloid ML que lance predicciones de la data del ETL.

Es muy importante para alcanzar un objetivo así la comunicación constante del equipo, las juntas diarias en las que se conozcan avances, se analice que se está cumpliendo los sprints planeados y si existen stoppers en el proyecto, hacerlos saber rápidamente. Cómo no hay un PM, será un poco díficil la administación del proyecto, pero un rol como el ingeniero SRE, podría tomar temporalmente el control de algunas de estas actividades.

4.- Cuáles son los niveles que el sistema que se podrían comprometer.

Es difíciles estimar correctamente esta parte mientras no se tenga en claro el alcance y la arquitectura bien definidas, por ejemplo, es posible establecer un SLA del 99 %, pero generalmente los costos para tener servicios en alta disponibilidad requieren replicas, por lo que es más costoso, por lo que ahí la restricción del dinero juega un papel muy importante. al igual que el tiempo de procesamiento, porque si bien existen tecnologías para hacer distribuidas las cargas de trabajo, muchas veces requieren más nodos trabajando, lo que igual es complicado con restricciones de dinero, pero es posible.