Diseño del DAaaS

Definición de la estrategia del DAaaS

La estrategia que se busca es automatizar las facturas de los proveedores de las empresas clientes minimizan la intervención humana.

Lo que se busca es la recepción de documentos en diferentes formatos como pueden ser PDF, Word, Excel, google doc, google sheet o imagen. Obtener los metadatos y convertir en texto, fechas o valores numéricos los datos de los documentos aportados por el proveedor bien por web, app o enviándolo a un mail de Gmail corporativo.

La empresa obtendrá un ahorro en los costes del procesamiento de las facturas, tickets y facturas simplificadas. Permitirá un archivado estructurado de la documentación de los proveedores con posibilidad de hacer búsquedas inteligentes y ahorrará espacio en su almacén por el archivado de dicha documentación ya que este software tendría la homologación previa de la AEAT como software para la digitalización certificada por la Hacienda Española.

Además se traducirían todos los conceptos de las facturas extranjeras a conceptos en español de forma automática.

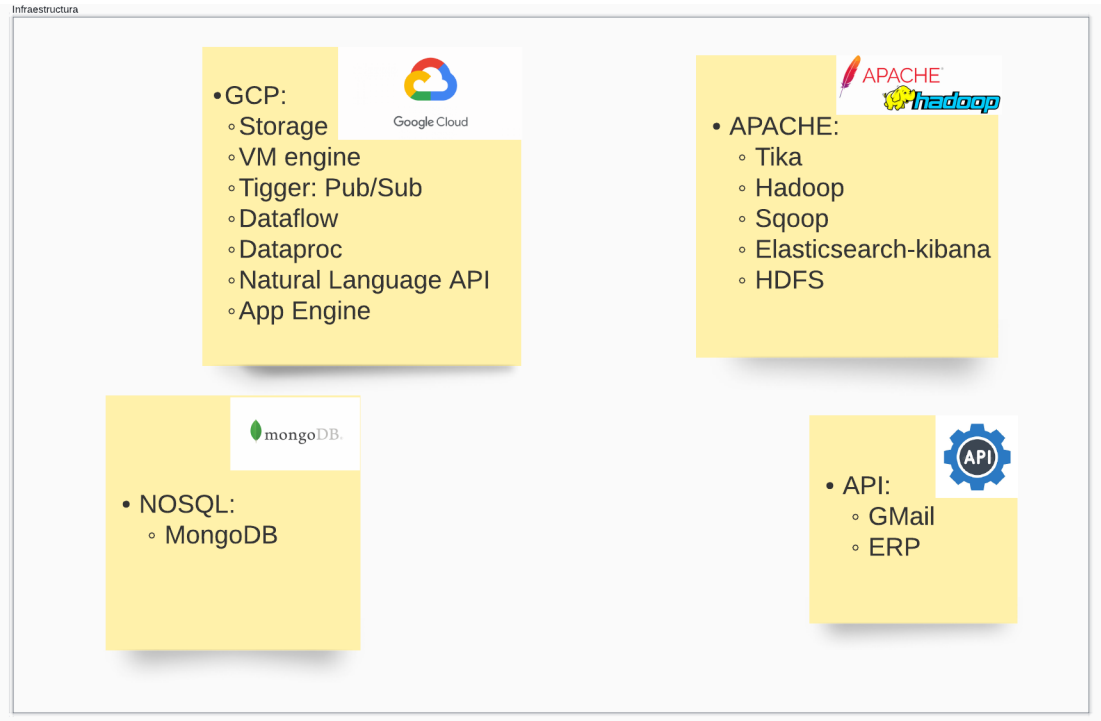
Así permitimos que los trabajadores se dediquen a las actividades de mayor valor añadido como es procesar la información recibida casi en tiempo real ya que el proceso trabaja 24/7 (24 horas al día/7 días a la semana) y no centrarse en tiqueo de datos datos en el sistema de forma reiterativa. Siendo más barato el coste de este servicio que el coste hora de un trabajador del departamento de contabilidad.

Se evitan los errores a la hora de contabilizar ya que el proceso es automático y según unas reglas lógicas y se abaratan los costes de auditorias externas e internas ya que lo único que se debería de auditar son las reglas de la modelización y los algoritmos para poder validez al área de proveedores en dichos trabajos.

A modo de resumen se adjunta el siguiente cuadro:

Arquitectura DAaaS

La arquitectura a nivel de resumen es la siguiente:



Las etapas del proceso de nuestro modelo operativo serían:

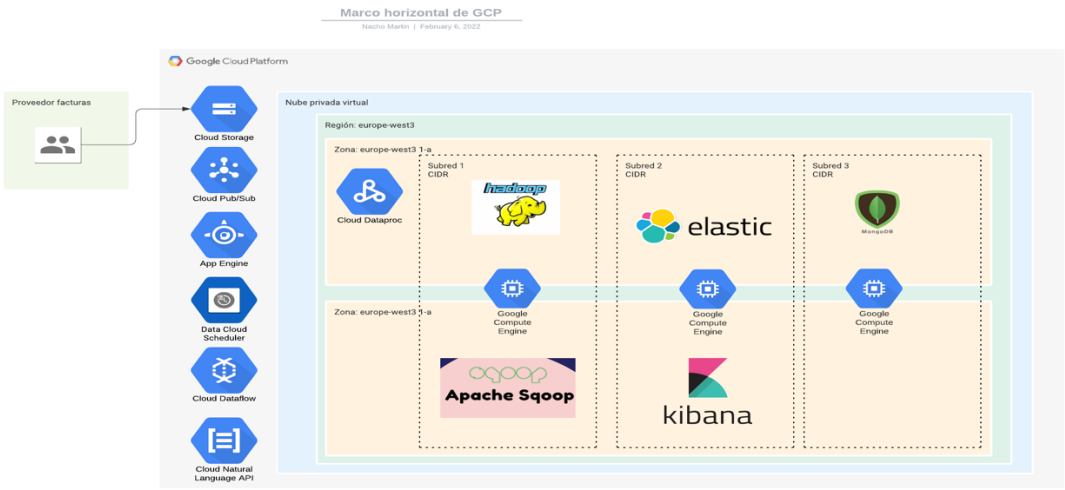
1. Etapa ingesta: Aquí se recibirían los documentos vía web o una app donde se podrían adjuntar los documentos por parte de los proveedores. Previamente se les ha dado unas claves de autentificación al sistema. Otra posibilidad que se posibilita es enviar los documentos a una dirección de Gmail corporativa.
   En está última vía del mail, se programaría la API de Gmail para poder obtener mediante programación el deposito de los documentos adjuntos en una staging área de documentos en Google Storage.
   A los documentos se le pasaría el Parser del ecosistema Hadoop que es Tika para obtener los metadatos y realizar la extracción de los datos en bruto.
   Necesidades en esta fase:
   - API Gmail
   - Google Dataproc 24/7
   - Google Engine, instancia de VM levantada para Hadoop-HDFS-Tika24/7
   - Data Function: Tigger Pub/Sub
   - Dataflow
   - Google Storage
2. Etapa preparación: Estos datos en bruto se archivarán en un data lake en HDFS y son datos no estructurados. Se guardará el link a la ruta del documento originario que se habrá archivado en Google Storage. Y se integraran con los datos que desease el proveedor, si así lo demandase como pueden ser ordenes de pedido emitidas por el cliente al proveedor para que después sean casadas en la lógica del proceso de contabilización, tabla de artículos inventariables para el cliente con sus códigos únicos, etc.
   Necesidades en esta fase:
   - Google Dataproc 24/7
   - Google Engine, VM Hadoop-HDFS 24/7
   - Data Function: Tigger Pub/Sub
   - Dataflow
   - Google Storage

3. Etapa Análisis: En esta etapa sobre los datos no estructurados se va una normalización y modelización con Google Natural Language API. Estos datos normalizados se van a archivar en una base de datos NoSQL, como es MongoDB, por que son en tiempo real y lo importante es la velocidad más que la integridad. Por último habrá un procesamiento de los datos con la herramienta del sistema Hadoop que es Elasticsearch para permitir el realizar una búsqueda documental de la información o del texto que aparece en los metadatos o en las entidades.
   Necesidades en esta fase:
   - Google Dataproc 24/7
   - Google Engine con tres maquinas virtuales levantadas, VM Hadoop-HDFS 24/7+VM Elasticsearch 24/7+VM MongoDB 24/7

- Google Natural Language API- App Engine
- Data Function: Tigger Pub/Sub
- Dataflow
- Google Storage

4. Etapa distribución: En esta etapa se hará la distribución de los datos que están en MongoDB ya normalizados y que cumplen con la modelización para poder ser entendidos por la API del ERP del cliente. Así para conectar la Base de Datos con la API del ERP utilizaremos la herramienta del ecosistema Hadoop que es Sqoop. Con Sqoop y una Google Function que es Google Scheduler programaremos cuando se va a realizar esta comunicación de los datos para que sean procesados en el ERP de nuestro cliente.
   También tendremos una visualización de como va en proceso através del dashboard de Kibana y los dashboard de Elasticsearch para saber el trabajo de los nodos y del Google Cloud Platform para saber información sobre el uso de los recursos y posible coste de la infraestructura.
   Así mismo se podría conectar la base de datos de MongoDB con Tableau para realizar diferentes visualizaciones que embebiendolas en nuestra web y dándoles claves a nuestro cliente las podría consultar y navegar.
   Necesidades en esta fase:
   - Google Dataproc 24/7
   - Google Engine con tres maquinas virtuales levantadas, VM Hadoop-HDFS-Sqoop 24/7+VM Elasticsearch-Kibana 24/7+VM MongoDB 24/7
   - Data Function: Tigger Pub/Sub, Google Scheduler
   - Dataflow
   - API ERP

En formato visual los recursos necesarios para esta infraestructura son:



Nota: Dentro del logo de Hadoop estaría incluido Hadoop, Tika y HDFS

DAaaS Operating Model Design and Rollout

El Modelo operativo es el siguiente:

1. Un proveedor introduce las facturas previa identificación con claves en una parte de nuestra web o por app. También se le da la posibilidad de enviarlas como documentos adjuntos a una dirección de email de Gmail.
2. Los documentos de la web y de la app son depositados directamente en un bucket de Google Storage como documentos pendientes de procesar. En relación a los documentos enviados por mail, se lanza una trigger de Pub/Sub que conecta con la API de Gmail para que cada vez que reciba un documento adjunto un programa de Python, programado por un data science, extrae los documentos adjuntos y lance un pipeline de Dataflow para que lo archive en el Google Storage en el bucket de documentos pendientes de procesar.
   Los documentos que existe en el bucket son de diferente formato como archivos de PDF, Excel, Word, Google Sheet, Google Doc e imágenes.
3. Se tiene que levantar un servicio Google Dataproc 24/7 porque la ingesta de los datos tiene que ser en tiempo real y tan pronto como llegue. Además se levantarán por un técnico de sistemas 3 instancias para máquinas virtuales (VM) en Google Cloud Platform dentro del apartado de Machine Engine, que también tendrían disponibilidad 24/7 dado el volumen de operaciones que se espera y que prima la inmediatez.
   Las tres VM son:
   - VM para Hadoop, HDFS y Sqoop
   - VM para Elasticsearch y Kibana
   - VM para MongoDB
4. Un técnico de sistemas ha programado un trigger Pub/Sub mediante Python que lanza un job de Hadoop para que los documentos que están en el bucket de documentos pendientes de procesar sean procesado por un parser del ecosistema Hadoop que es Tika. Tika realiza una lectura y extracción de los datos con independencia del formato, extrayendo los metadatos y las entidades de los ficheros. Además lanza un pipeline de Dataflow que trapasa los documentos a otro bucket en Google Storage como documentos procesados y graba dentro de los metadatos la ruta del acceso a dicho archivo. Con lo cual hemos pasado de tener archivos a tener unos metadatos y entidades con estructura clave-valor de manera no estructurada para el proceso de contabilización en un ERP de nuestro cliente. Los datos obtenidos se guaran en HDFS con un timestamp de la fecha de este proceso de extracción de datos.
5. También se ha programado por un técnico de sistemas que le lance otro trigger Pub/Sub en Python para que procese por lotes los nuevos datos archivados en HDFS creando un fichero CSV y dejando registrado con un nuevo timestamp de cuando se ha generado este lote. En dicho trigger se lanza

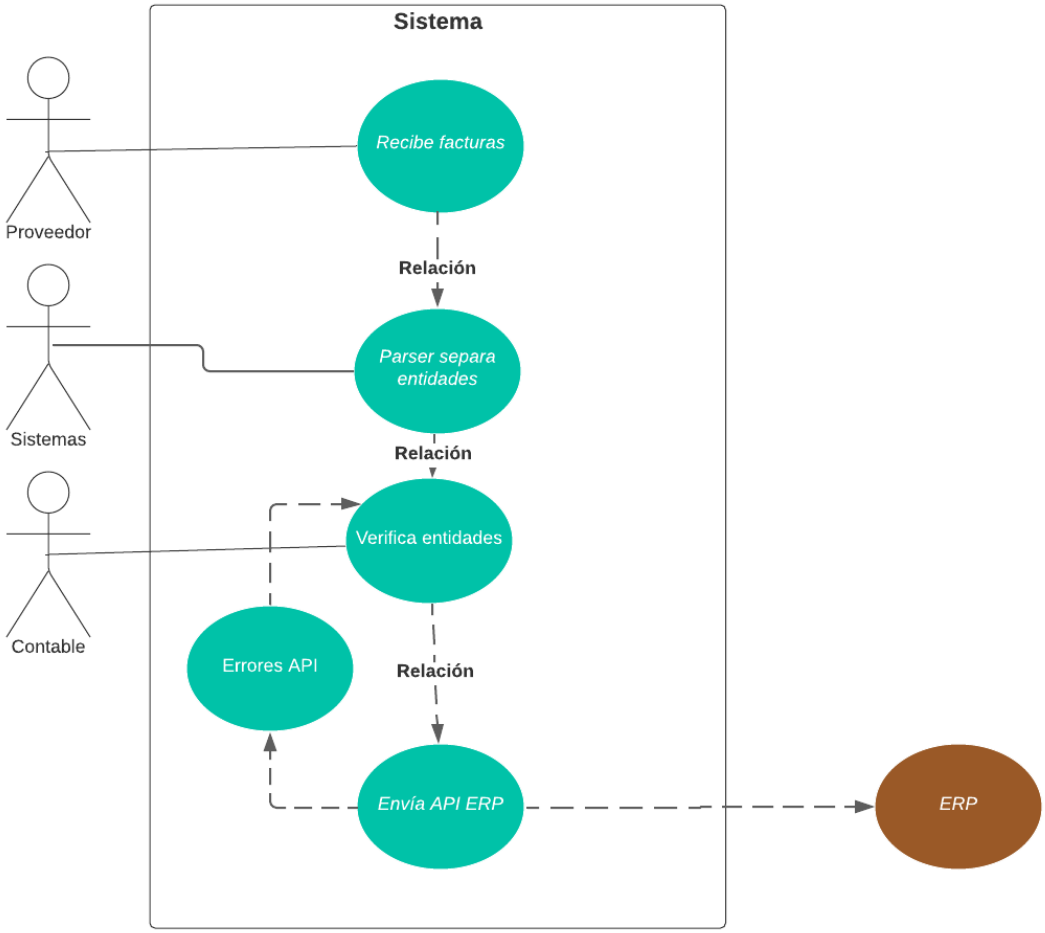
también un Dataflow para enrutar dicho fichero CSV para realizar un análisis NLP que en Google lo realiza la herramienta Natural Language API.

6. Los ficheros CSV son tratados por Natural Lenguage API de tal manera que mediante algoritmos y modelaje programado por un técnico data science permite obtener los datos clave-valor que se necesitan para ser procesados por el ERP del cliente. Este es un proceso de AI, de tal manera que establece un modelo de esquema de datos para ese proveedor-cliente-tipo fichero, de tal manera que cada vez que cambie algún parámetro se procede a la verificación externa por parte de un técnico contable del cliente para verificar que la propuesta asignada mediante Inteligencia Artificial se ajusta a su parametrización en su ERP. Esto se realiza mediante una comunicación push en la app, con un link a la dirección de archivado del archivo de la factura del proveedor y la propuesta de la extracción del modelo de esquema de datos propuesto por nosotros. Si es Ok se procede a archivar este modelado dentro del algoritmo de procesamiento y se no es OK se ve en que se ha modificado para que la AI interna proceda a ajustar mejor en futuros casos. Una vez procesado los ficheros CSV son borrados.
7. Los datos ya estructurados con Natural Language API son lanzados por Dataflow para que se generen ficheros CSV, cuyos datos son exportados a una base de datos MongoDB. Este fichero CSV es depositado en Google Storage en un nuevo bucket con el nombre de Elastic.
8. Un nuevo trigger Pub/Sub programado por un técnico de sistemas genera un software que lee los CSV y genera un índice en Elasticsearch para posteriormente lanzar un proceso MapReduce. Después de este proceso se podría realizar una búsqueda inteligente por documento, por proveedor, por palabras. Se activa el SSH y se conecta con Kibana para poder ver la información en el dashboard de Kibana y la información del proceso en el dashboard de Elasticsearch.
9. Del dashboard de Kibana se le da claves al cliente para que pueda ver en tiempo real la siguiente información:
    - Número de facturas procesadas
    - Proveedores procesados
    - Número de facturas procesadas por centro de trabajo en un mapa
    - Listado de productos más comprados
    - Proveedores más importantes e importe acumulado procesado
10. Un técnico ha programado en google cloud scheduler una acción Pub/Sub en Python para que cada 2 minutos sea lanzada para consultar en MongoDB los nuevas facturas registradas y lanza un job de Hadoop para que Sqoop que es el conector con bases de datos relacionales lance un log que permita transmitir la información de la consulta de MongoDB a la API del ERP del cliente para que dicho datos sean registrados en la Base de Datos del ERP y

contabilizadas las facturas de los proveedores según el modelaje y la paramentrización de cada cliente.

Del sistema se obtendrán una serie de informes para saber que el proceso es el correcto de tal manera que se podrán ver el dashboard de Google Cloud Platform para saber la utilización de las instancias y el uso del sistema, del dashboard de Elasticsearch se podrá ver el uso de los nodos y el tiempo de ejercución y por último en Kibana se podrá ver gráficamente el número de documentos procesados por cliente. Así mismo se podría establecer un túnel de información con Kafka se recoga la información de los logs de este proceso y lo analice con ElasticSearch para ver cuantos veces se produce la palabra error para poder analizar las causas de dichos erroes en tiempo real.

Un esquema del caso de uso podría ser el siguiente con las personar intervinientes:



Dentro de sistemas hay que incluir las dos facetas de los programadores de las triggers y de la orquestación de la infraestructura que sería un técnico del departamento de sistemas con conocimientos en Google Cloud Platform, ecosistema Hadoop y Python. Aparte también estaría el técnico data science que programaría la

inteligencia artificial en el proceso NPL que tendría que saber de Text Mining con Python y los algoritmos de NPL.

Diariamente se obtendría un informe de consumos del Google Cloud Platform por cada proyecto de cliente para saber las desviaciones según lo presupuestado y si son negativas proceder a realizar los ajustes oportunos.

Desarrollo de la plataforma DAaaS.

Del desarrollo de la plataforma se busca la máxima automatización del proceso y que la intervención sea la mínima posible una vez que haya sido modelizado las plantillas de cada proveedor y que el propio sistema con técnicas de AI vaya aprendiendo. Existen varios automatismos que se han programado con Python para que con una sola intervención la lógica del sistema pueda escalar horizontalmente para diferentes proveedores y en diferentes lenguajes ya que el propio sistema permite traducir la información a diferentes idiomas.

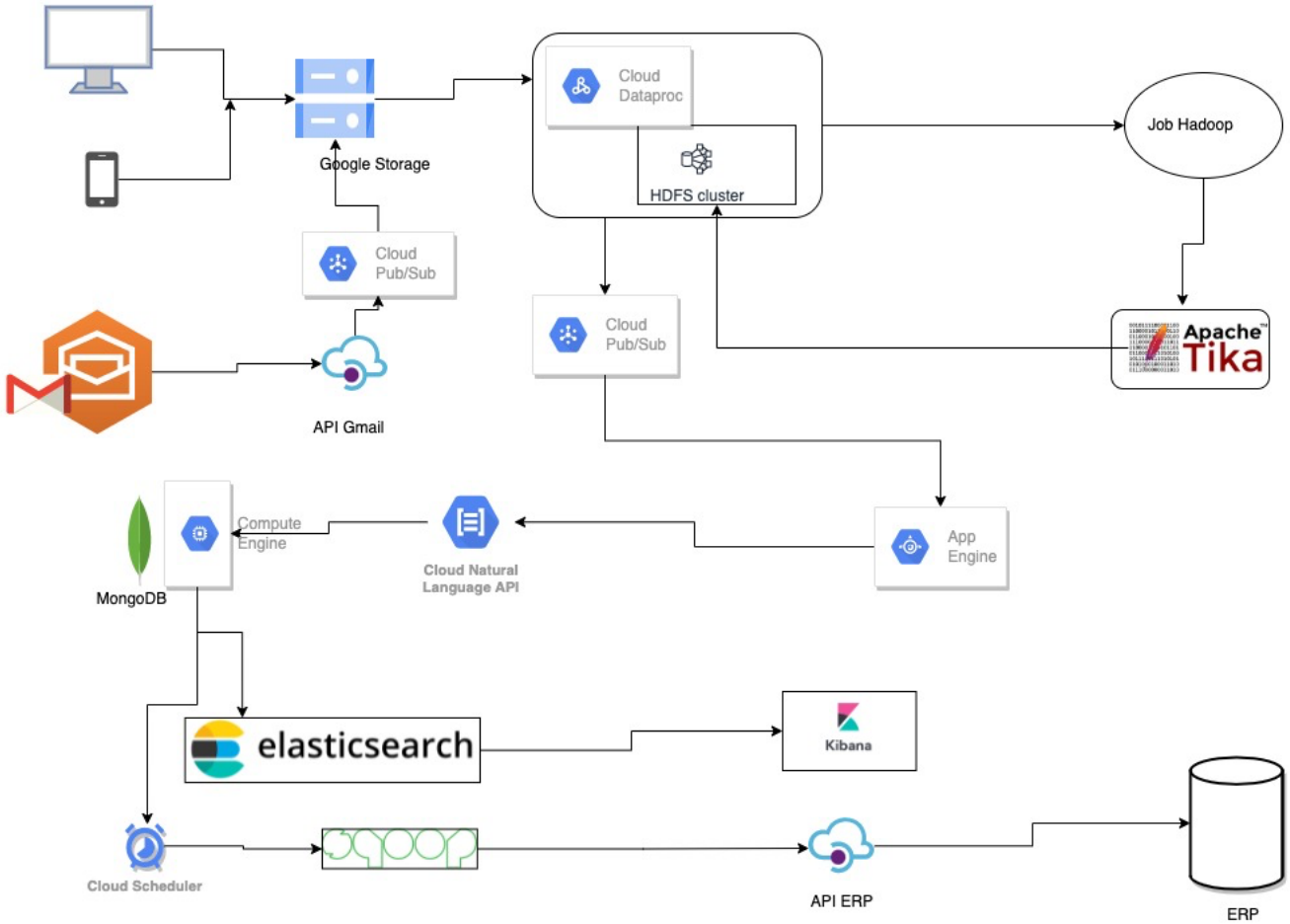
Cuando se obtiene un nuevo cliente se realiza un proceso de prueba para comprobar que los datos obtenidos por la plataforma se ajustan a las necesidades de nuestros clientes y hasta que no se ha verificado con distintos tipos de documentos y proveedores que esto es así no se sube a producción las plantillas modelizadas de dicho cliente.

La carga de datos es masiva para que el sistema este trabajando en un sistema 24/7 y permite un alto procesamiento de datos gracias al ecosistema Hadoop unido al escalado horizontal de maquinas que permite Google Cloud Platform.

Para el desarrollo de la plataforma se necesita:

- API Gmail y ERP
- Ecosistema Hadoop
- Google Cloud Platform (GCP)
- Python
- Link a librerías
- Link a documentación técnica
- Link a artículos
- Información del sistema en Kibana, Elasticsearch y Consola de GCP
- Posibles conexiones a herramientas de visualización externas como Tableau que permita analizar los datos existentes en MongoDB y los contabilizados en el ERP del cliente en relación a los proveedores

Diagrama:


Google Storage
Cloud Dataproc
HDFS cluster
Job Hadoop
Cloud Pub/Sub
Cloud Pub/Sub
API Gmail
Apache Tika
Compute Engine
Cloud Natural Language API
App Engine
MongoDB
elasticsearch
Kibana
Cloud Scheduler
API ERP
ERP