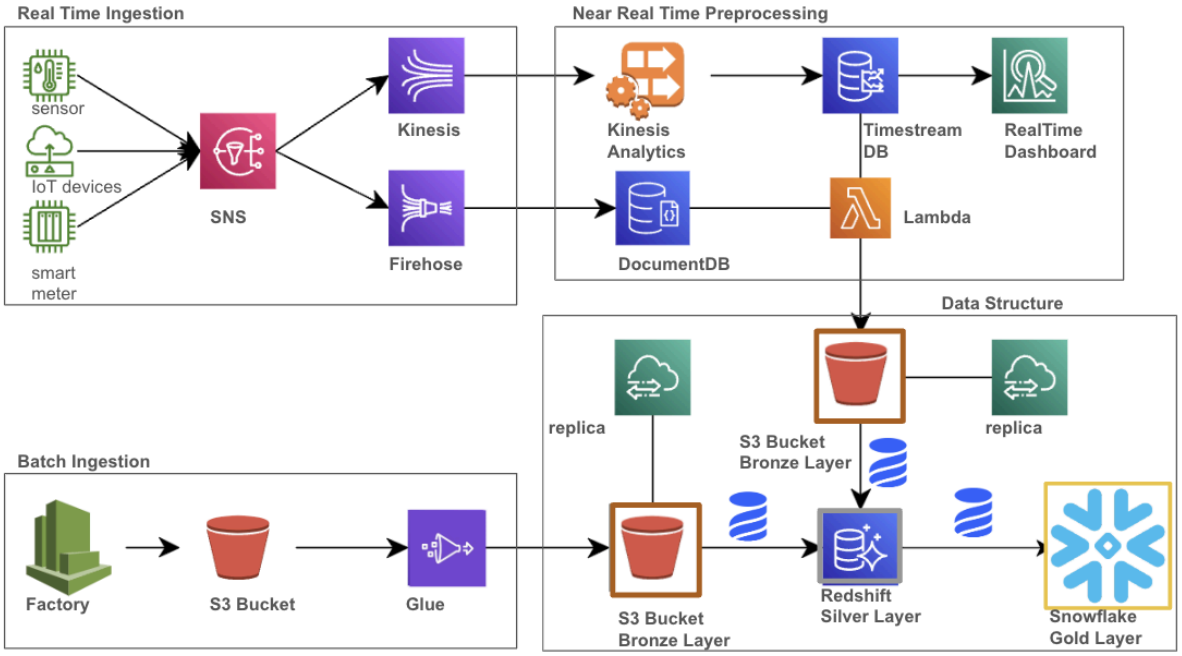
Arquitectura Batch & Real Time

Hace cuatro años en Landis+Gyr creamos un producto mínimo viable para validar la migración de una arquitectura de datos *on-premise* a AWS. Era un reto en el que tuvimos que mantener la base de datos operativa (en snowflake) al tiempo que implementamos una nueva solución para un nuevo cliente del sector automovilístico. Actualmente es la arquitectura de datos operativa de unos de los productos principales de la compañía y, aunque en la versión actual algunos servicios de AWS como Firehose, Kinesis o Lambda y han sido sustituidos por una plataforma microservicios montada en EKS, en esencia la estructura es la misma. Al tratarse de un producto mínimo viable contamos con los servicios autogestionados de AWS, esto te permite despreocuparte de la gestión de la escalabilidad y soporte de la integración de sus diferentes componentes.



El flujo de datos comienza por la ingesta de un conjunto de datos que proceden en tiempo real desde unos sensores de humedad y temperatura, unos sensores IoT en brazos robóticos y los contadores eléctricos de la factoría. El dato es enviado a un sistema de notificación tipo Pub/Sub - *AWS SNS* - que no solo captura el dato sino que permite filtrar *dead letters*, deduplicar y ordenar los eventos. En este punto el dato se duplica, por una lado entra en un data stream de *AWS Kinesis* un servicio que permite ingresar gigabytes en tiempo real y ejecutar un servicio de preprocesamiento tipo Flink que se activa bajo demanda cuando alcanza una determinada capacidad digamos 5GB. En este punto el proceso pasa de real-time a near-real-time, de este modo al no tener los servicios activados en streaming bajamos mucho el coste operativo. Este servicio de Kinesis Analytics limpia y filtra el dato en tiempo real y lo guarda en una base de datos de series temporales tipo InfluxDB - *AWS Timestream* - que nos permite hacer queries de baja latencia y con ello crear una conexión para un dashboard que muestra el estado del flujo eléctrico de la factoría minuto a minuto. Por otro lado, tenemos otro suscriptor que manda el datos

generado en tiempo real a un canal de *AWS Firehose,* un servicio que permite tener un proceso de ETL en streaming que filtra, agrega y carga el dato en una base de datos tipo MongoDB - *AWS DocumentDB.* En este punto finaliza un proceso de ingestión de datos en tiempo real que permite almacenar los eventos creados en las últimas 168 horas de forma segura y autoescalable. Para guardar el histórico del dato, una función *AWS Lambda* se ejecuta cada hora para almacenar el dato en un *bucket S3* que a su vez está replicado en una *availability zone* distinta, este dato guardado en parquet ya pertenece a la capa bronce de la base de datos.

Hay otro proceso de ingesta de datos, esta vez, en batch, que parte de los datos generados por la factoría en términos de potencia eléctrica, volatilidad, precio, riesgo de blackout, etc. Son datos necesarios para el monitoreo de los modelos de predicción de demanda, modelos de detección de anomalías y modelos de seguridad. Estos datos se generan cada 8 horas y se reciben en un servicio tipo SFTP que provee *AWS S3*, y que mediante ua ETL tipo Spark - *AWS Glue* - vamos a filtrar y limpiar el dato y guardarlo en otro bucket de S3, de nuevo replicado en una *availability zone* diferente; en este punto el dato ya pertenece a la capda bronce de la base de datos.

Un vez el dato está en la capa bronze, tanto el dato generado en tiempo real como el dato procedente de la factoría, será manejado por Liquibase una herramienta que permite manejar una base de datos y controlar su versionado y los pipelines que la crean usando código que se ejecuta dentro del proceso de CI/CD, con esto todos los cambios y actualizaciones serían rastreables y consistentes. Esta herramienta de migración, en comparación a la que se estaba usando hasta entonces - Redbase Flyway - permite usar tablas externas en S3 y conectar a Snowflake de forma nativa; además ofrece el changelog, de forma que se puede ver los cambios en cada pipeline y en cada migración y tambíen permite hacer el rollback con varias conexiones (redshift y snowflake) al mismo tiempo.

Con esta herramienta podemos crear la base de datos a partir de un conjunto de queries y un conjunto de conexiones (Redshift, Redshift Spectrum y Snowflake). No solo la orquestación de los pipelines sino los procesos de CI/CD se manejan desde esta herramienta. Aunque la monitorización del volumen de datos ingestados lo hacemos usando *AWS CloudWatch* todo el resto en relación a la observabilidad, calidad y gobierno del dato recae en servicios que ofrece Liquibase, una herramienta que desde que usamos en este MVP forma parte de muchas otras arquitectura de datos en Landis and Gyr.

Con respecto a la seguridad de la arquitectura usamos las herramientas disponibles en AWS. De este modo teníamos la arquitectura aislada usando VPC y grupos de seguridad para el acceso a diferentes subnets cuando conectan dos availability zonas distintas. Por otro lado, los servicios de la ingesta en tiempo real están protegidos a su vez con *AWS Private Link* para conectar los servicios a la VPC de forma biyectiva.