

Cómo aplicar **NoSQL** en casos reales

Predicción de impagos en e-commerce

Búsqueda de documentos optimizada
en la administración pública

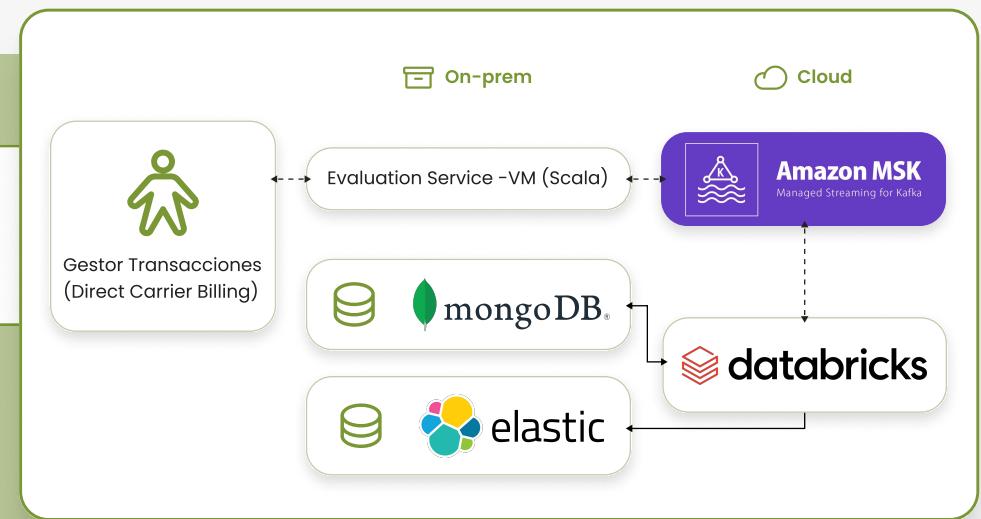


Cómo aplicar

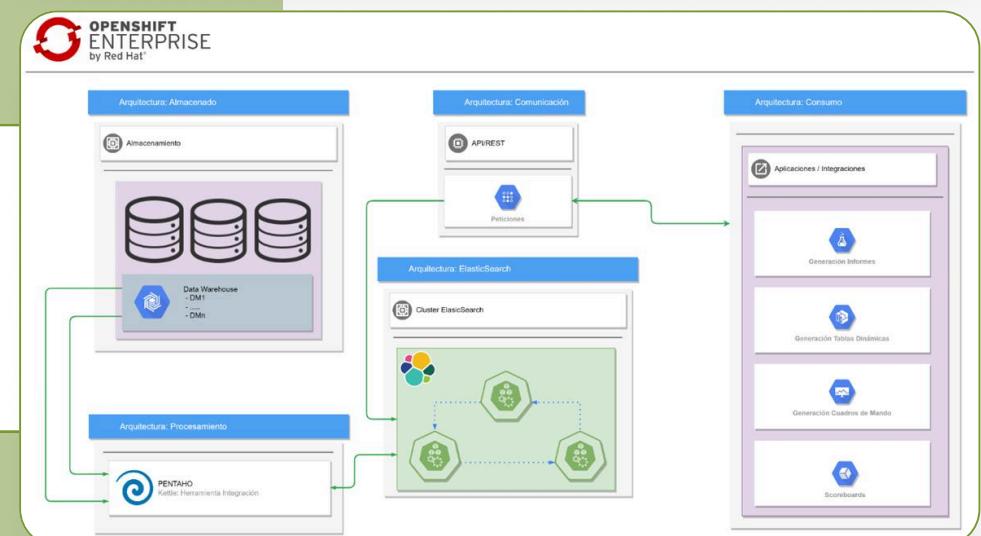
NosQL

en casos reales

 | Predicción de impagos en e-commerce



 | Búsqueda de documentos optimizada en la administración pública





1. Predicción de impagos en e-commerce

Objetivo

Rechazar las transacciones con alta probabilidad de impago

- Ej. Apple Store, Google Play, Play Station, ...
- Facturación única a través del operador de telefonía (Direct Carrier Billing).

Escenario de datos

- **Clientes:** id, alta en DCB, tipo de contrato, tipo de cliente, perfil de riesgo, ...
- **Transacciones:** id de cliente, site (ej. Google, Apple, ...), operator (Vodafone, Másmóvil, Movistar, ...), precio, status, ...
 - Se reciben en tiempo real. Actualmente unos **100 millones de transacciones por cada operador.**
- **Bad-Debts:** Información real de facturas impagadas, proporcionada mensualmente por cada operador





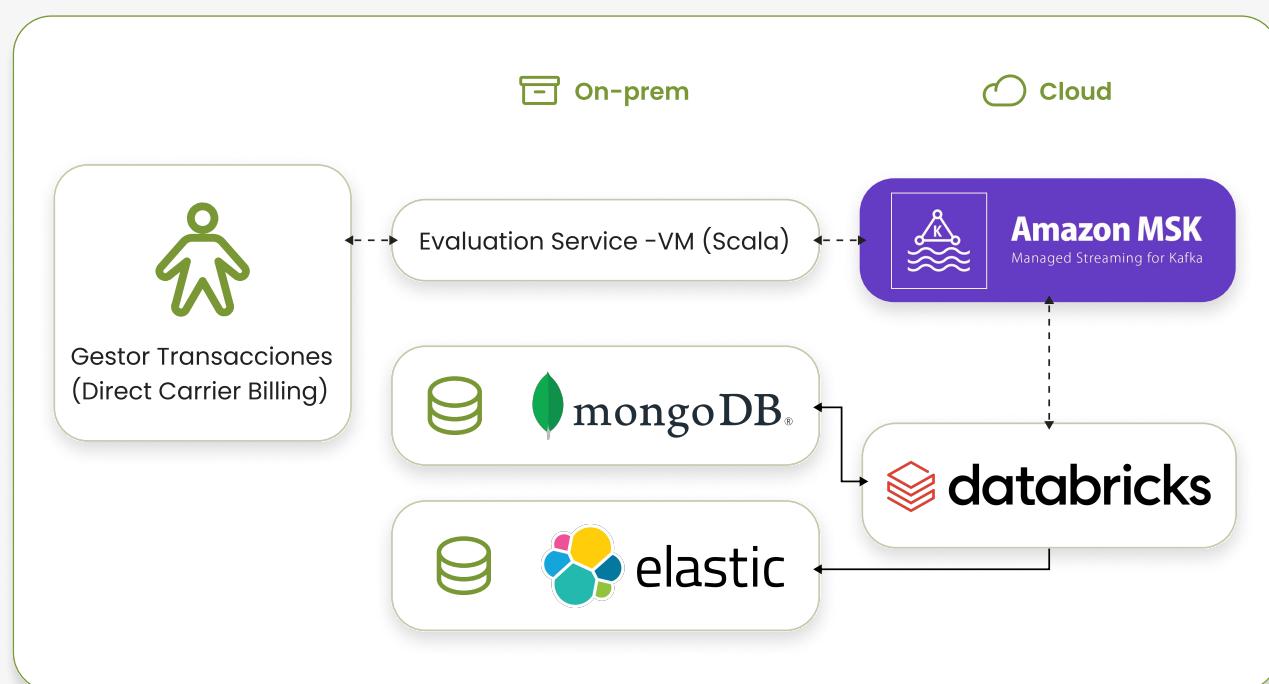
1. Predicción de impagos en e-commerce

Arquitectura híbrida Big Data cloud/on-premise

- **Cloud:** Amazon AWS Databricks, Kafka MSK
- **On-premise:** MongoDB, Elasticsearch, MySQL

Mongo DB como almacenamiento principal del sistema

- Clúster de 3 nodos
- **Carga** de transacciones y usuarios en **tiempo real**
- **Lectura masiva** de esos datos para el entrenamiento de modelos de clasificación
- **Almacenamiento de los modelos** entrenados





1. Predicción de impagos en e-commerce

Mongo DB

como almacenamiento principal del sistema

ADD DATA

`_id: "dc96bb...d-4c0b-8c40-f5cdcf"
time: "2017-03-31T22:01:01Z"
msisdn: "34660156106"
merchant: "MRRES_31344"
site: "STBra0301561"
lastUpdated: "2018-01-13T21:34:57Z"
status: "OK"
price: 3.5
payProb: 0.999984502792358
threshold: 0.978
modelId: "5b4691c...lc03dc125`ede"
willBeBadDebt: false
operator: "movistar.es"
userId: "5af...9ba2a5624ad1af0de"
> tags: Array
transactionType: "SUBSCRIPTION"
isBadDebt: false
wasRejected: false
errorCodeDescription: "test"`

Transacciones

bigdata.badDebts.model_movistar.es

375 DOCUMENTS

4

`_id: ObjectId("5b62dc29b9dc03ddeb1b4c...")
modelByteArray: BinData(0, 'AAAA...')
modelByteArrayLinear: BinData(0, 'AAAA...')
isLatest: true
lastUpdated: "2018-08-02T10:25:45Z"
operator: "movistar.es"
site: "ST_App_0001"
> cateFeatMap: Object
> meanNumFeatMap: Object
> featScore: Object
threshold: 0.34
> modelInfo: Object
> modelParams: Object
nDaysBlindPeriod: 150
> simulation: Array`

Modelos entrenados





2. Búsqueda de documentos optimizada en la administración pública

Objetivos / Caso de estudio

- Proporcionar un **sistema de búsqueda** que permita obtener los **trámites solicitados por el ciudadano** de una forma **ágil, accesible** y con unos **tiempos de respuesta ultra rápidos**.
- Como requisito **adicional**, es necesario disponer de un **sistema de recomendación** para el ciudadano **basado en las búsquedas realizadas**.



elasticsearch

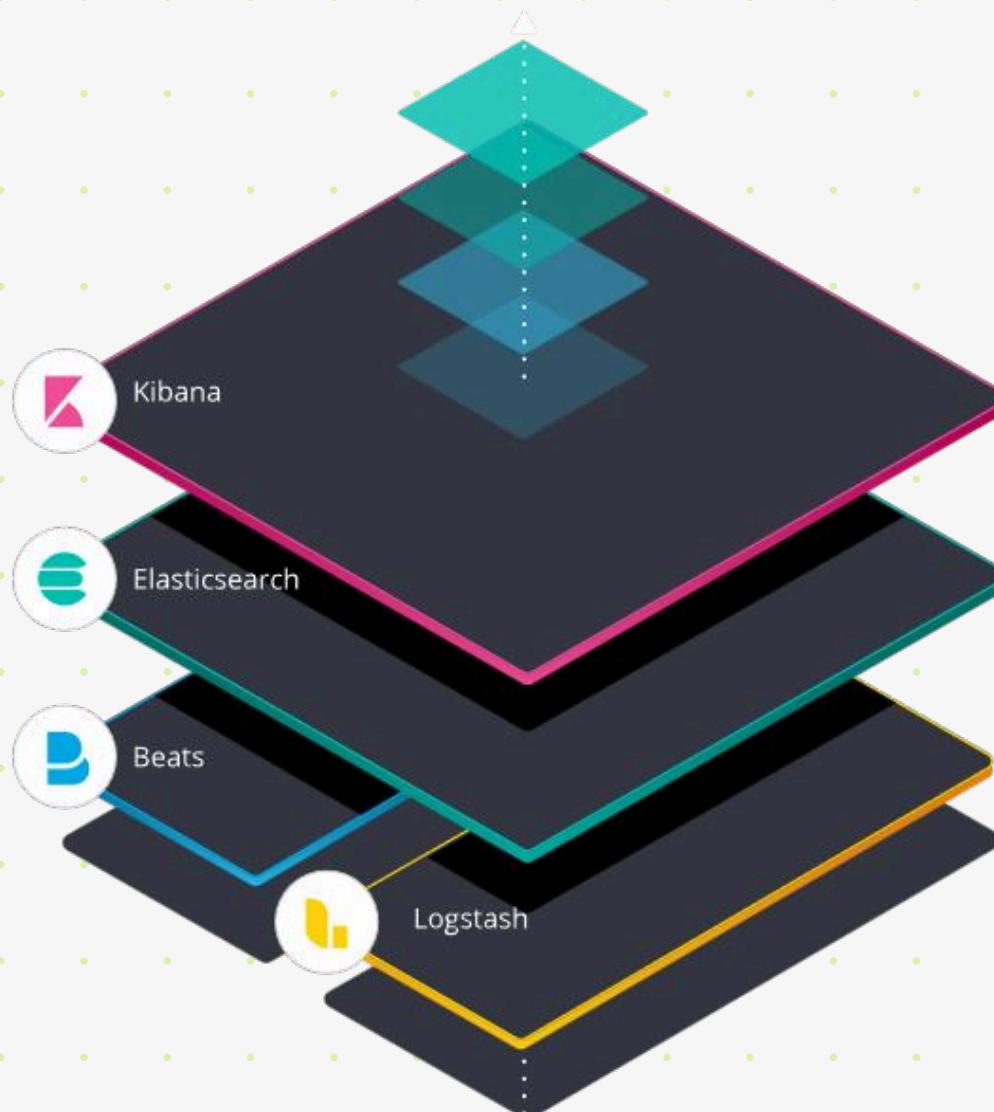




2. Búsqueda de documentos optimizada en la administración pública

¿Qué es Elasticsearch?

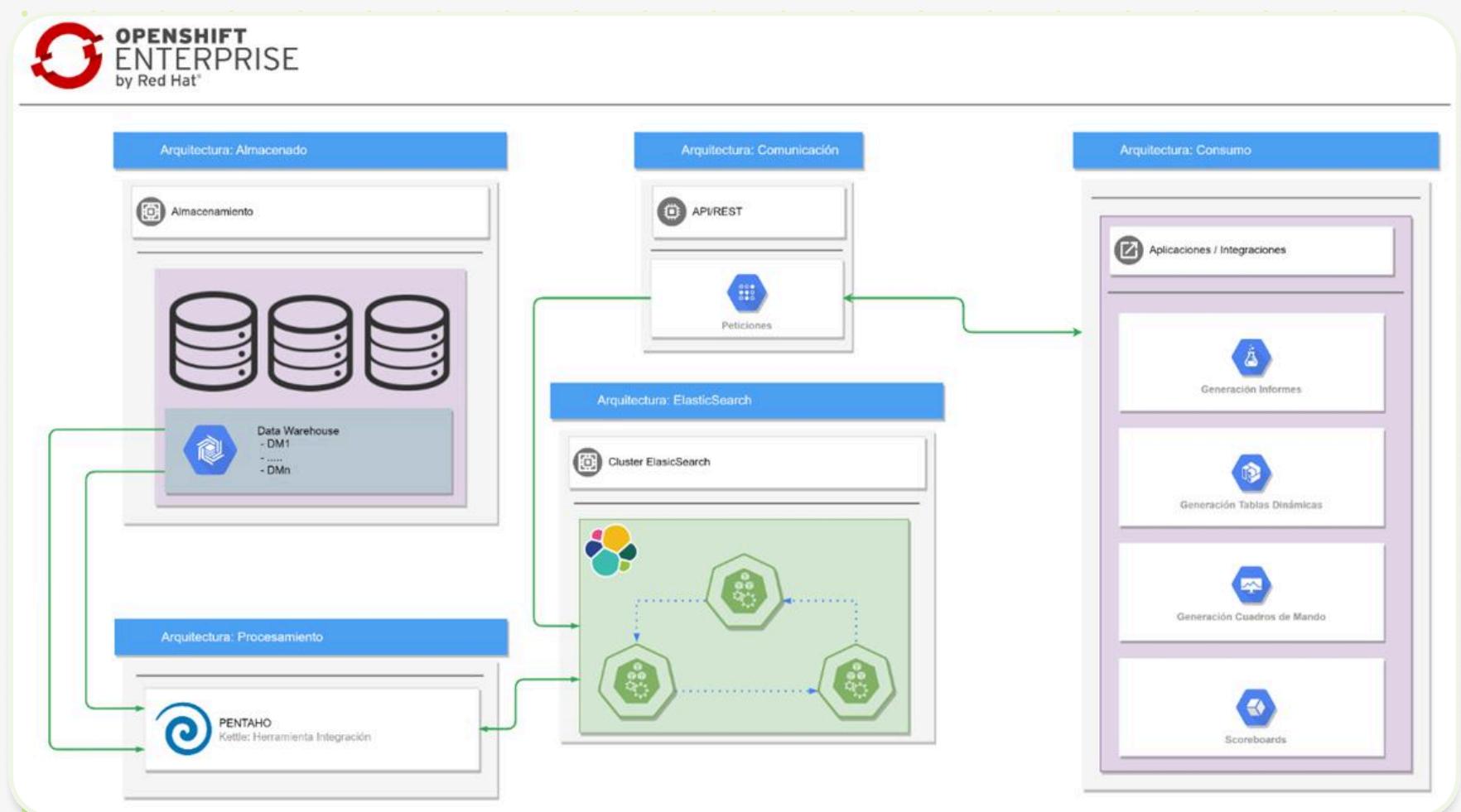
- Motor de búsqueda **Open-source**, desarrollado en Java y de tipo NoSQL Documental
 - **Respuestas instantáneas mediante el uso de Índices Invertidos (Inverted Index)**
- El ecosistema completo de Elastic se conoce como **ElasticStack (ELK)**:
 - **ElasticSearch**: módulo principal de búsqueda y almacenamiento
 - **Logstash**: modulo de ingesta
 - **Kibana**: herramienta de visualización y monitorización





2. Búsqueda de documentos optimizada en la administración pública

Arquitectura y flujo de datos

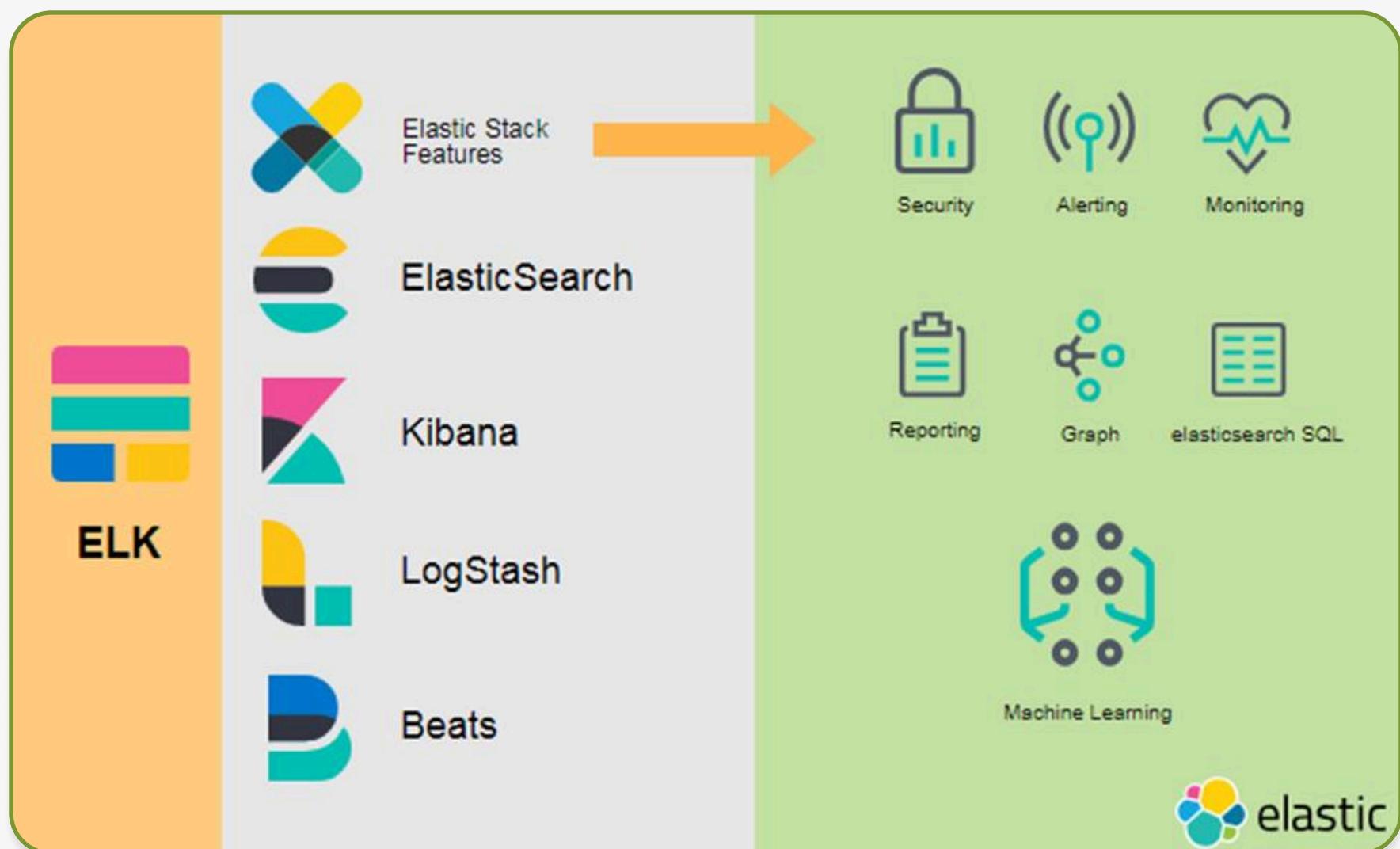




2. Búsqueda de documentos optimizada en la administración pública

Aspectos y técnicas destacables

- **Facilidad y rendimiento** a la hora de insertar, actualizar y/o borrar documentos
- Propio lenguaje de búsqueda a través de su **API/REST**
- Posibilidad de integración con cualquier aplicativo que haga uso de API/REST





2. Búsqueda de documentos optimizada en la administración pública

Retos y soluciones

- **Complejidad en el mantenimiento**

- Propio lenguaje de búsqueda a través de su **API/REST**
 - Desarrollo de procesos de carga incremental para asegurar la **integridad** del dato sobre los **índices**
- Opciones de **seguridad** avanzada sujetas a licencias.
 - Uso de integraciones terceras como Keycloak y Azure AD para proveer una mayor seguridad





2. Búsqueda de documentos optimizada en la administración pública

Conclusiones

- **ElasticSearch** ha conseguido cumplir con todos los requisitos especificados y proporcionar a los ciudadanos un motor de búsqueda ágil con tiempos de respuesta muy rápidos





Técnicas y tecnologías

Principales Aplicaciones y Tecnologías (Comerciales y Open Source)

Procesamiento batch & real-time



Extracción, transformación y carga
ETL – ELT

Adquisición y movimiento



Almacenamiento



Repositorios Big Data

NoSQL



Machine Learning

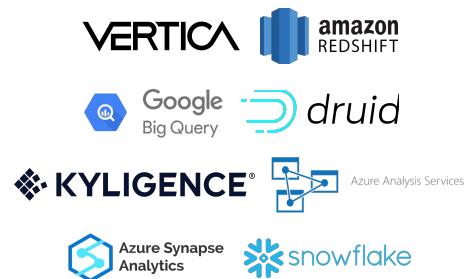
Machine Learning



Explotación de Datos

Gobierno de Datos

Modern Data Warehouse & OLAP



Dashboarding & Reporting



Notebooks



Data Governance





Uso de Arquitecturas Open Source

Ejemplo de Arquitectura 100% open-source para implementación Data Lakes

Visualización

Exploración y analítica avanzada



Cuadros de mando



Procesamiento

Batch, Real-Time y Machine Learning



Indexación y búsqueda



elasticsearch

Almacenamiento

Data Lake



Data Warehouse



NoSQL



Ingesta y orquestación



Otros

Gobernanza de Datos



DataHub

CICD



Jenkins

Repositorio de código



GitLab