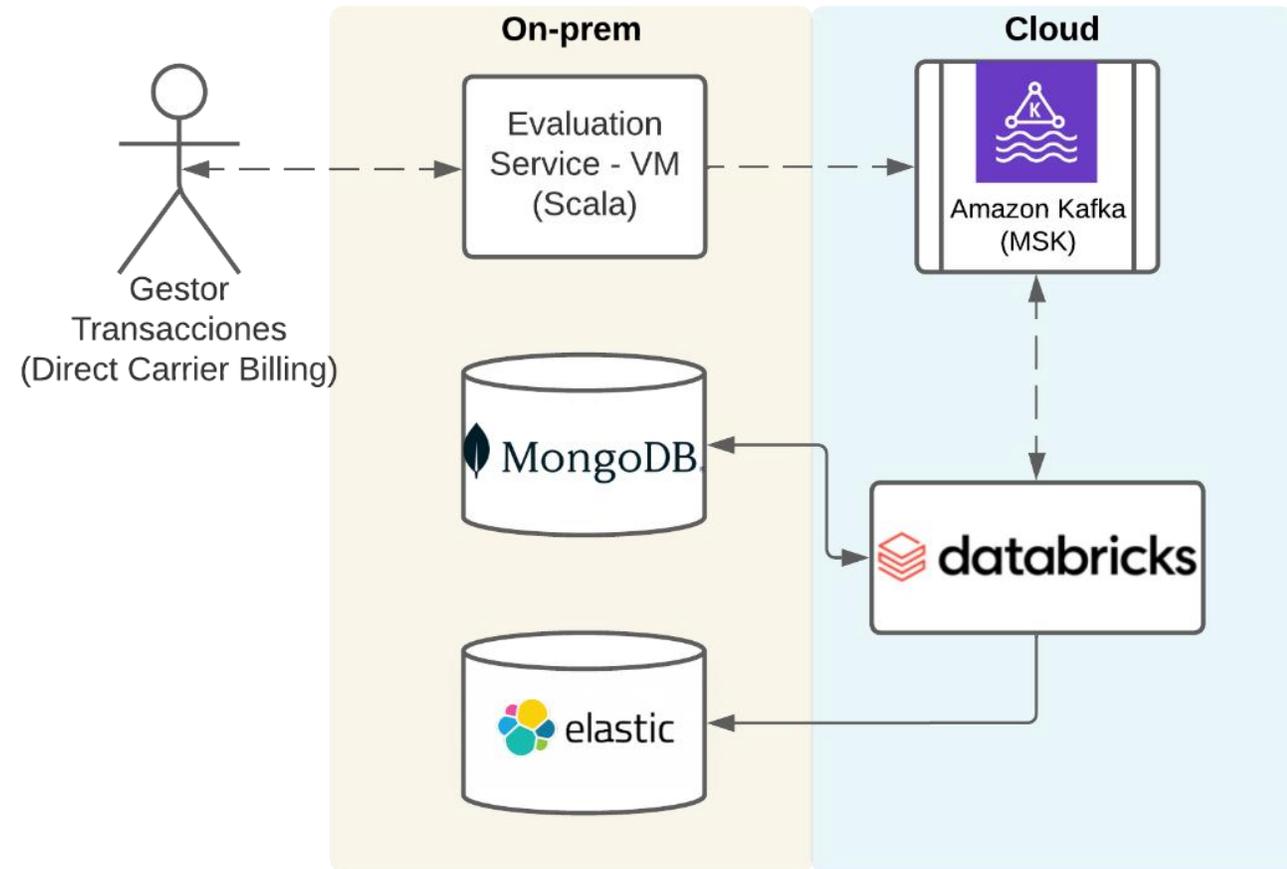


COMO APLICAR NoSQL EN CASOS REALES

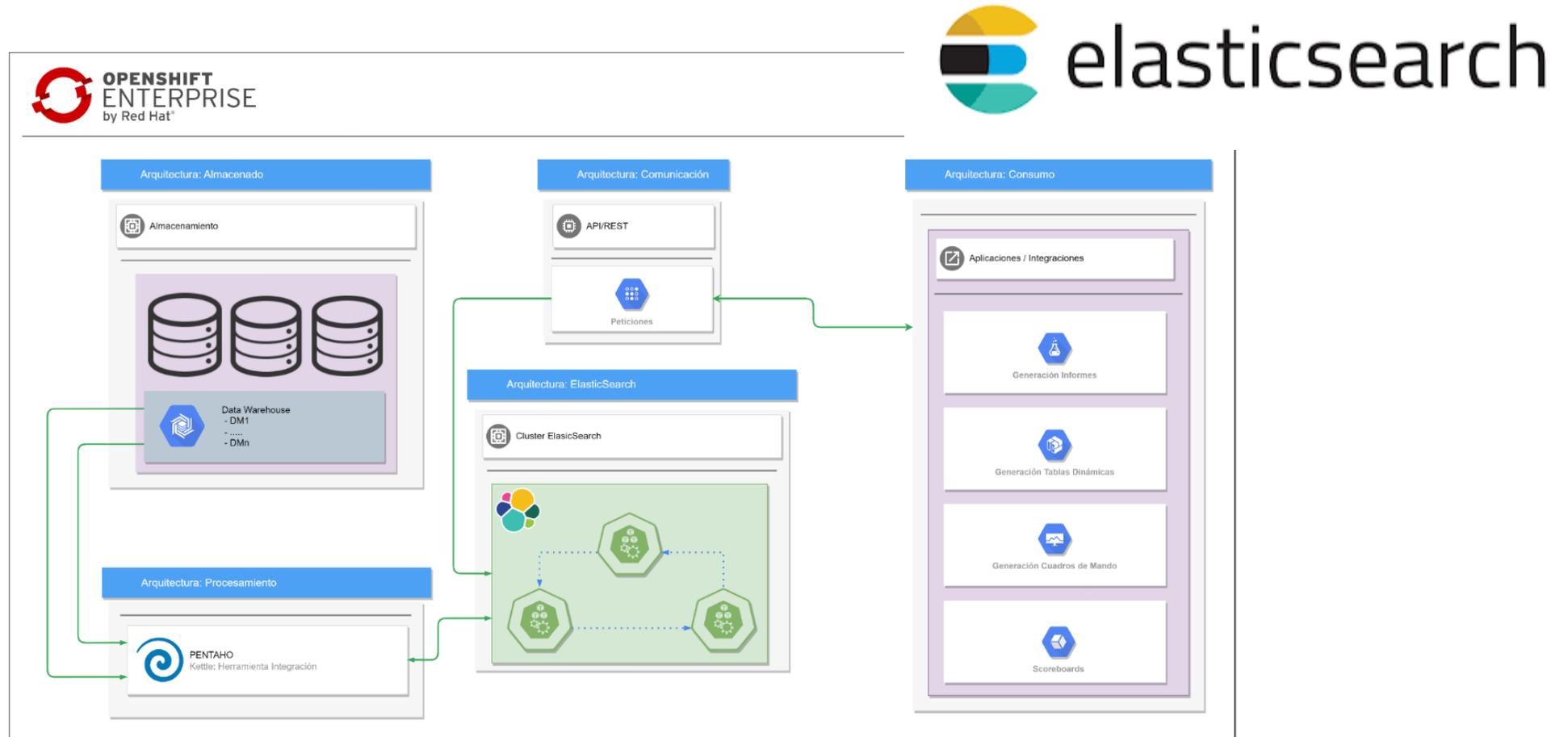
1. Predicción de impagos

en e-commerce



COMO APLICAR NoSQL EN CASOS REALES

■ 2. Búsqueda de documentos optimizada en las AAPP



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)

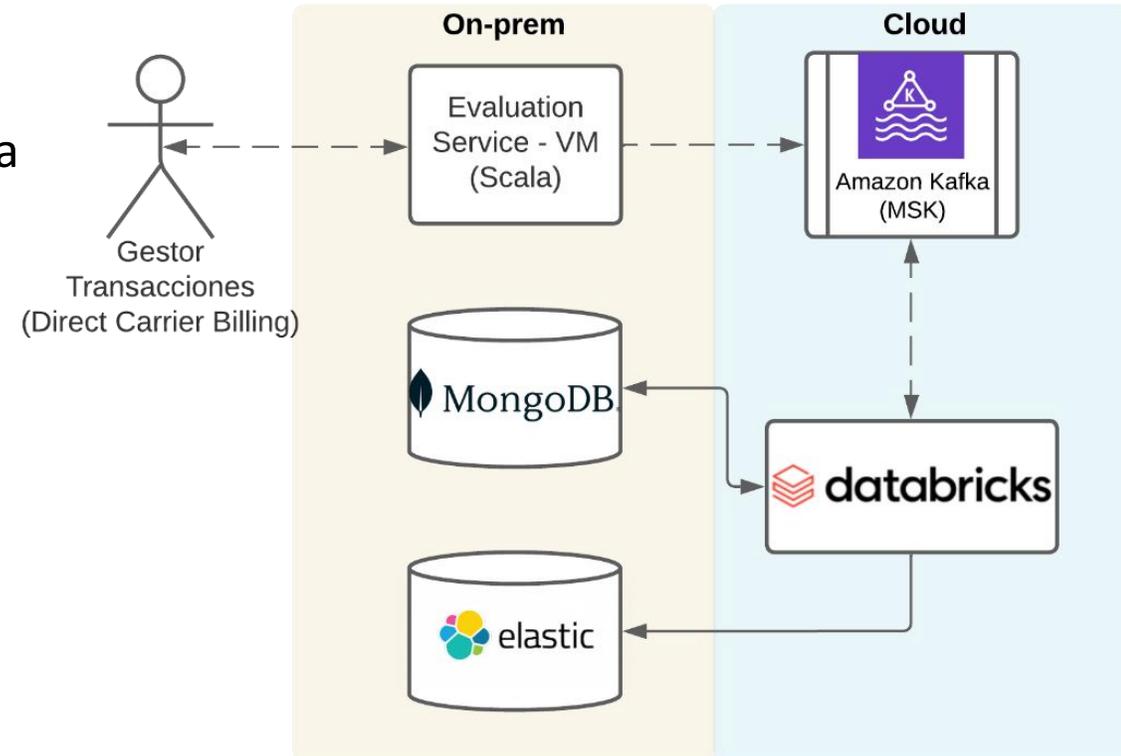
- **Escenario de datos**

- **Clientes:** id, alta en DCB, tipo de contrato, tipo de cliente, perfil de riesgo...
- **Transacciones:** id de cliente, site (ej. Google, Apple,...) , operator (Vodafone, Más móvil, 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

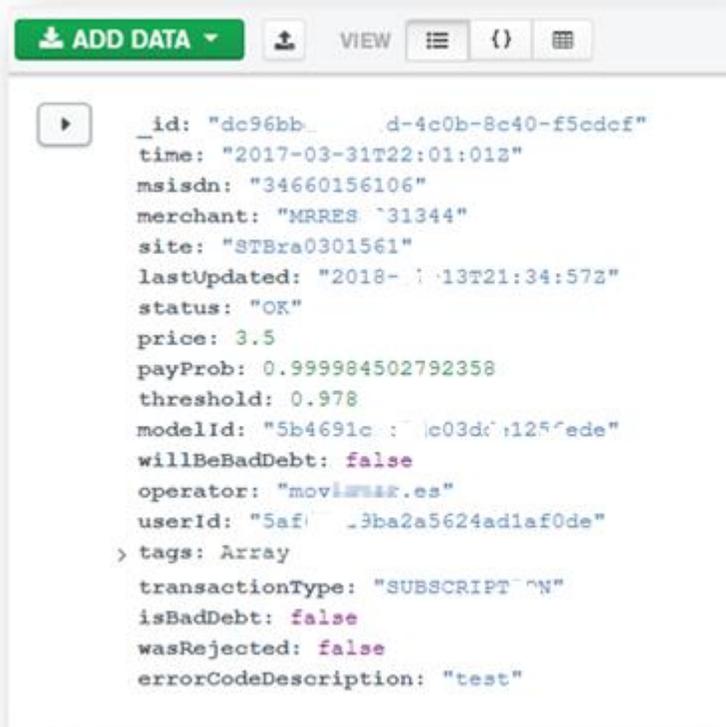


- **Arquitectura híbrida Big Data cloud/on-premise**
 - Cloud: Amazon AWS Databricks, Kafka MSK
 - On-premise: **Mongo DB, Elasticsearch, MySQL**
- **Mongo DB como almacenamiento principal del sistema**
 - Clúster de 3 nodos
 - **Carga de transacciones y usuarios en tiempo real**
 - **Lectura masiva** de esos datos para el entrenamiento de modelos de clasificación
 - **Almacenamiento de los modelos** entrenados



2. Predicción de impagos en e-commerce

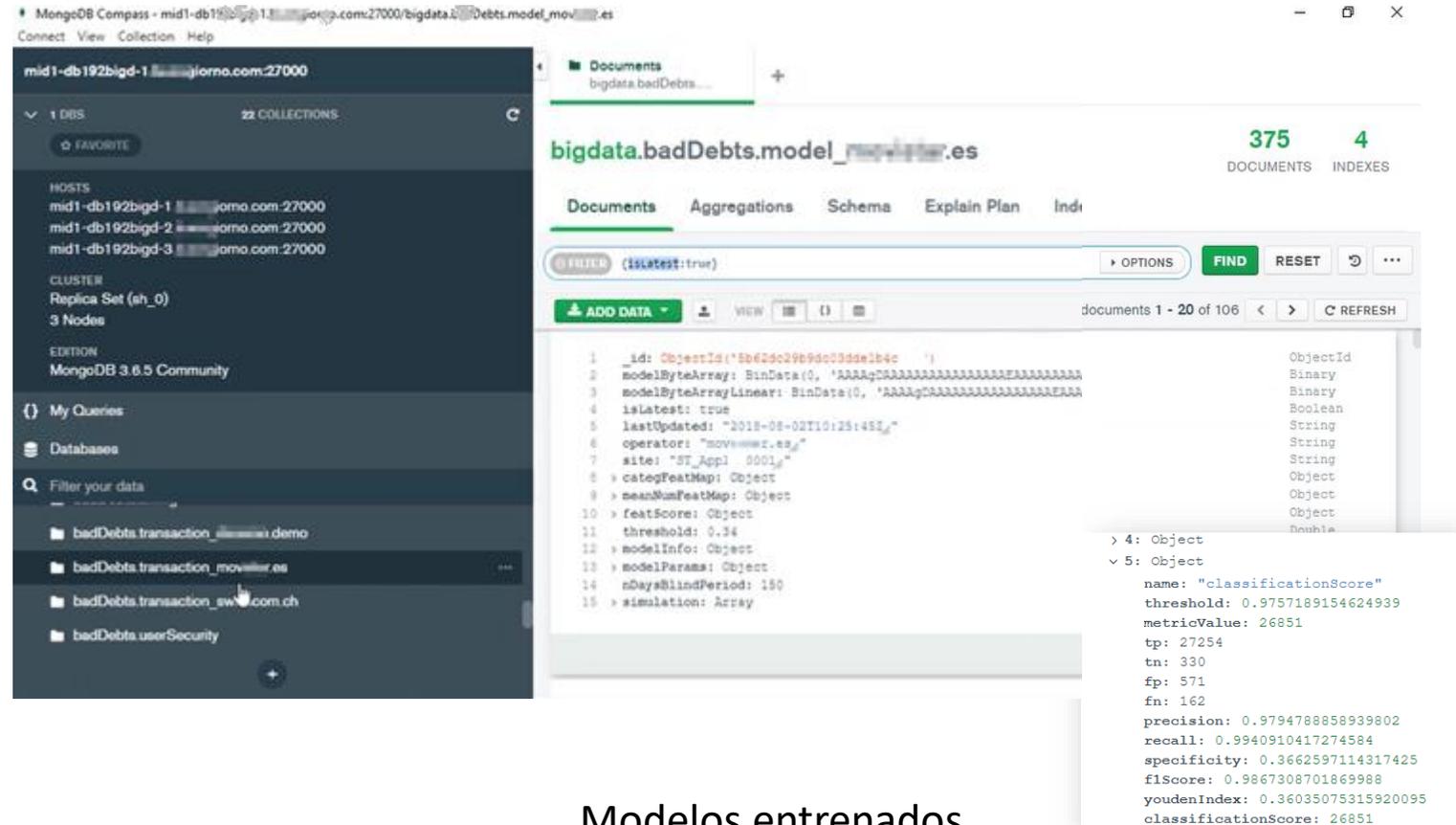
- **Mongo DB** como almacenamiento principal del sistema



```

_id: "dc96bb...d-4c0b-8c40-f5cdef"
time: "2017-03-31T22:01:01Z"
msisdn: "34660156106"
merchant: "MRRES `31344"
site: "STBra0301561"
lastUpdated: "2018-08-13T21:34:57Z"
status: "OK"
price: 3.5
payProb: 0.999984502792358
threshold: 0.978
modelId: "5b4691c...c03d...125fede"
willBeBadDebt: false
operator: "movistar.es"
userId: "5af...3ba2a5624ad1af0de"
tags: Array
transactionType: "SUBSCRIPTION"
isBadDebt: false
wasRejected: false
errorCodeDescription: "test"
  
```

Transacciones



MongoDB Compass - mid1-db192bigd-1...bigdata.badDebts.model_movistar.es

mid1-db192bigd-1...bigdata.badDebts.model_movistar.es

Documents: 375, Indexes: 4

Filter: (isLatest:true)

```

1  _id: ObjectId('5b4691c...c03d...125fede')
2  modelByteArray: BinData(0, 'AAAAA...')
3  modelByteArrayLinear: BinData(0, 'AAAAA...')
4  isLatest: true
5  lastUpdated: "2018-08-02T10:25:45Z"
6  operator: "movistar.es"
7  site: "ST_Appl_0001"
8  categFeatMap: Object
9  meanNumFeatMap: Object
10 featScore: Object
11 threshold: 0.94
12 modelInfo: Object
13 modelParams: Object
14 nDaysBlindPeriod: 150
15 simulation: Array
  
```

Object

```

> 4: Object
> 5: Object
  name: "classificationScore"
  threshold: 0.9757189154624939
  metricValue: 26851
  tp: 27254
  tn: 330
  fp: 571
  fn: 162
  precision: 0.9794788858939802
  recall: 0.9940910417274584
  specificity: 0.3662597114317425
  f1Score: 0.9867308701869988
  youdenIndex: 0.36035075315920095
  classificationScore: 26851
  
```

Modelos entrenados



2. Búsqueda de documentos optimizada en la administración pública (Elasticsearch)

■ Objetivos / Caso de estudio

- Proporcionar un **sistema de búsqueda** que permita obtener los **tramites 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**

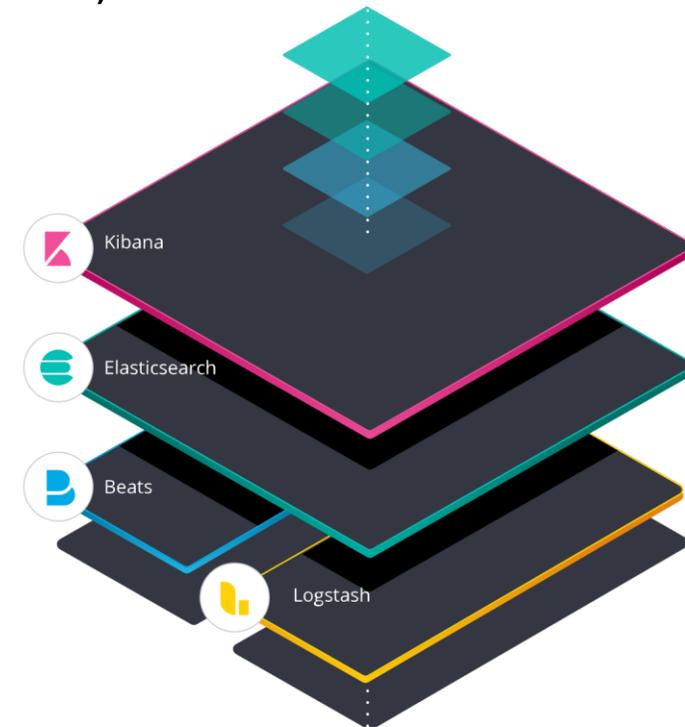


2. Búsqueda de documentos optimizada en la administración pública (Elasticsearch)



■ ¿Qué es Elasticsearch?

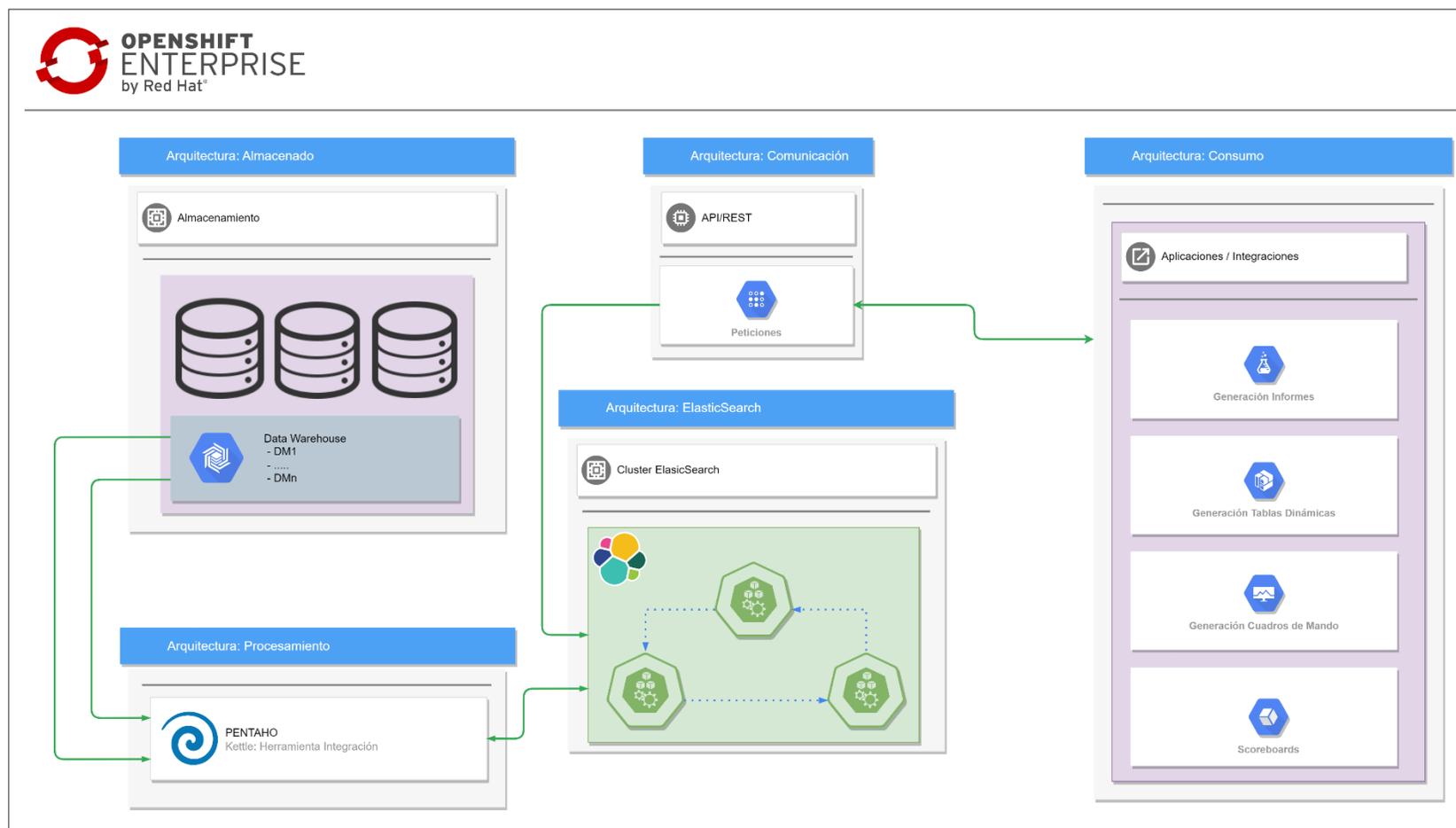
- Motor de búsqueda **Open-source**, desarrollado en Java y de tipo **NoSQL Documental**
 - Respuestas instantáneas mediante el uso de **Índices Invertidos** (Inverted Index)
- El ecosistema completo de Elastic se conoce como **Elastic Stack (ELK)**:
 - Elasticsearch: módulo principal de búsqueda y almacenamiento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





2. Búsqueda de documentos optimizada en la administración pública (Elasticsearch)

■ Arquitectura y flujos de datos

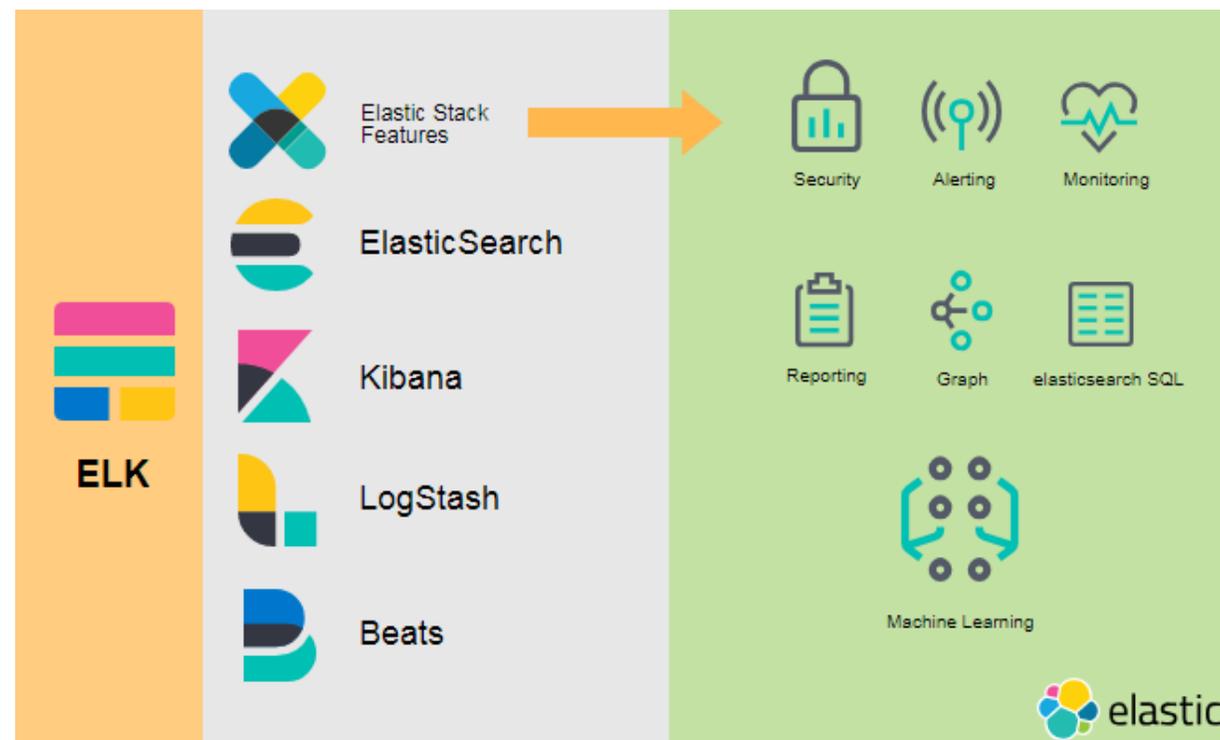




2. Búsqueda de documentos optimizada en la administración pública (Elasticsearch)

■ 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 (Elasticsearch)

■ Retos y soluciones

- Complejidad en el **mantenimiento**
 - 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 (Elasticsearch)



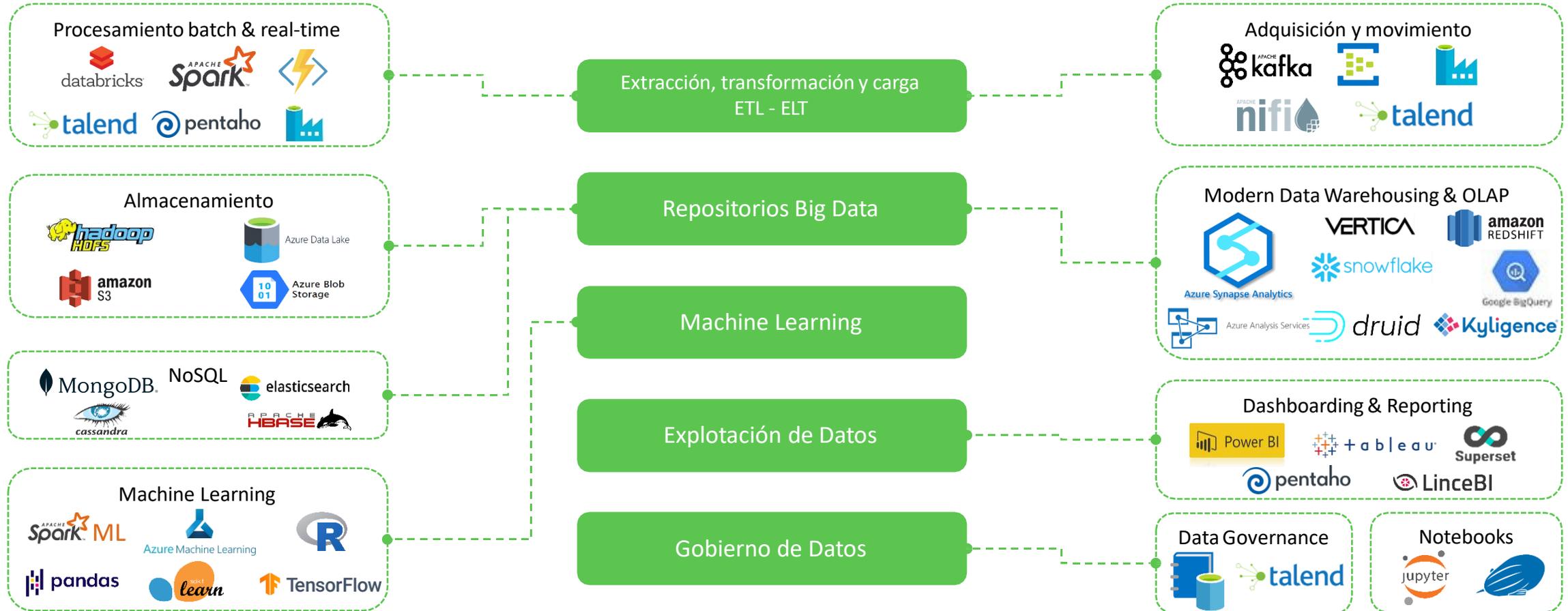
■ Conclusiones

- **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):



Uso de Arquitecturas Open Source

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

