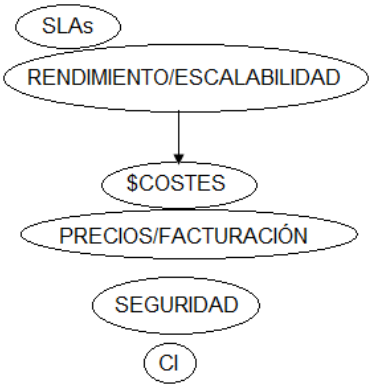
Rendimiento I. KPIs

OBSERVABILIDAD



- Detectar CAMBIOS
- Detectar ESTADO SERVICIO/SISTEMA
- Observar FLUJO DE DATOS
- MONITORIZACIÓN
- LOGS/MÉTRICAS

Servicio está funcionando:

- Forma que damos el servicio (SLA)
    - ¿Cómo es el SLA y cómo se lo damos al cliente?
    - Porcentaje de DISPONBILIDAD
    - RENDIMIENTO/ESCALABILIDAD
        - Repercute directamente → COSTES
    - De cara a los clientes → ¿Cómo van a ser los PRECIOS?
        - PRECIOS/FACTURACIÓN
        - Persona pone el precio basándose en los datos

        (Todo repercute en costes y precios, es de lo más importante)
    - SEGURIDAD → ej. Empresas que han cerrado por la brecha de datos
    - CI/CD → Nos actualiza el sistema

Hay dos tipos de datos:

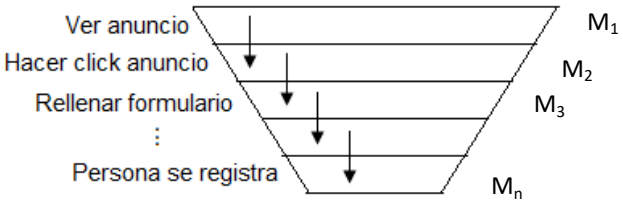
- DATOS DE OPERACIÓN
- DATOS DE NEGOCIO
    - ¿Cuánto le cobras al cliente por cada uso?

  Ej. Log de una petición a una Api
    - Utilizar sobre la carga → Datos Operación
    - Utilizar para facturación → Datos de Negocio

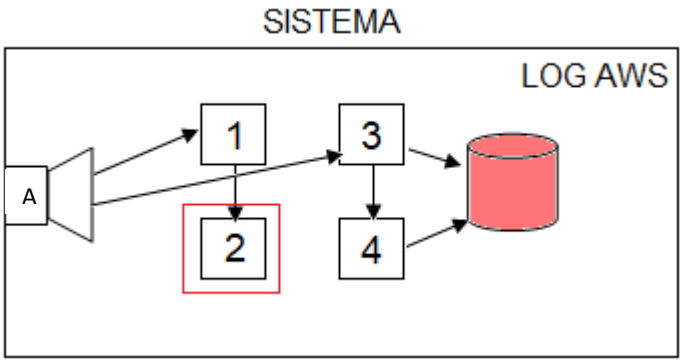
  Ej. Netflix
    - Servicio distribuido en la nube
        - Input/Output de disco
        - Conexiones BD
        - Nº de servicios funcionando en el momento
        - CPU de los clústeres
        - Pago por suscripción personal
        - Funnel de ventas
        - Cada acción que hacen los clientes genera un dato

DATOS OPERACIÓN (Disco, Red,…)

DATOS NEGOCIO
Luego se explotarán
(Saber cómo se van comportar los usuarios)



¿Cómo se sacan todos esos datos para generar información?
(Información: Datos de Operación Y Negocio)



¿De dónde viene el dato?

- Catalogado:
  - Inicialmente tenemos el BALANCEADOR (Kong)
    - *Este cliente desde esta IP ha realizado…. ha tardado …*
    - ACCESO (LOG de Acceso) [Nivel de balanceo] **(A)**
      - Log de Sistema (/var/log)
      - Syslog
  - Nivel de máquina: Si un servicio está funcionando o no. **(2)**
    - LOG DE MÁQUINA
    - MÉTRICAS: CPU, RAM…
  - LOG DE BD
  - LOG del propio servicio en sí:
    - SERVICIO LOG/MÉTRICAS **(4)**←Implementamos nosotros
  - LOGs AWS
    - Ej. Métricas de cómo está funcionando la máquina

Hay que acotar y priorizar. ¿Qué nos va a dar datos más rápido? EL BALANCIADOR
(Podemos ver si necesitamos escalar o no)

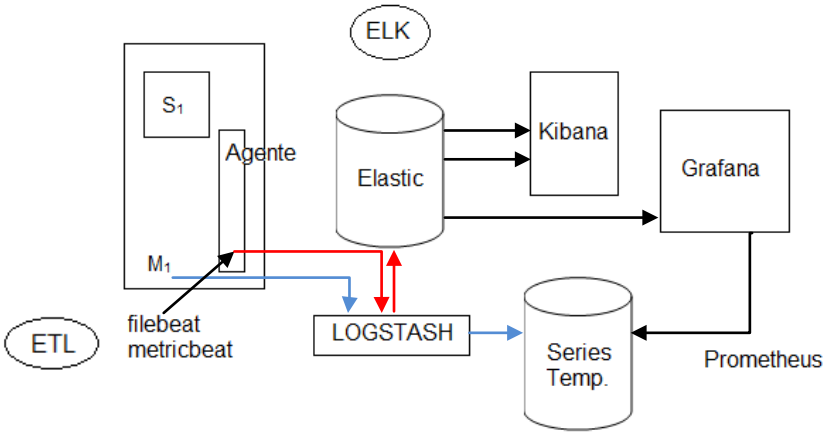
Hay que intentar homogeneizarlos para que todo se pueda interpretar de la misma manera.

DATO DE NEGOCIO: Director, jefe de empresa…
DATO DE OPERACIÓN: Desarrollador de producto, los que están en operaciones…

TIEMPO →Los datos no se pueden borrar porque están alimentando todos los cuadros de mando que están viendo la gente de dirección.
Ej. Mirar logs CPU cada 10 segundos puede ser de 6GB a 10 GB al día en un clúster de una herramienta que genera datos del sistema.

¿Cómo quito ese volumen, lo convierto en información y lo borro?
Ej. Log de operación, pongo el nº de registros en un campo y borro esos X registros. En el Log de negocio, eso no se puede hacer, ya que el cliente te puede pedir que demuestres porque le estas cobrando X cosa de tal día.

ARQUITECTURAS GENÉRICAS



Se añade un pequeño agente recolector para que coja esos logs y los envíe a un destino (Ej. Postgres).

Ir a la BD a estilos documentales (MongoDB, Elastic)

¿Cuándo queremos investigar que está ocurriendo?
Buscar X campo y devolver Y documentos.
Es una buena opción enviarlos a Elastic que manda a la herramienta Kibana para hacer el dashboard.

Para manipular los datos o hacer un catalogado (ETL) usamos logstash. Añadiendo está herramienta se aporta robustez.

Only SQL está pensado para series temporales (Prometheus)

Para consultar el elastic y bases de datos como Prometheus usaremos Grafana.

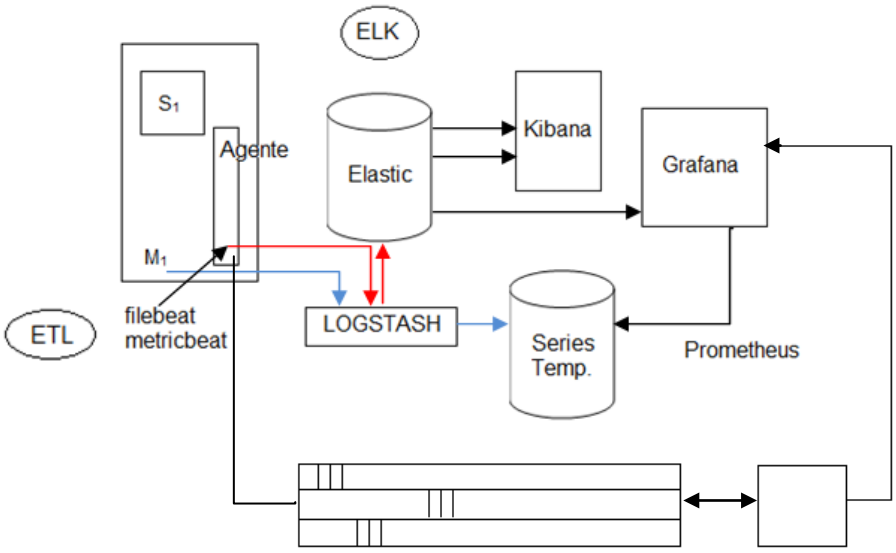
Elasticsearch al ser distribuido:

- ROBUSTEZ Y DISPONIBIIDAD
- Tasa de ingesta de datos muy alta

Es una arquitectura basada en peticiones síncronas.

Otro tipo de arquitecturas que funciones mediante publicación/suscripción:

- Tenemos un agente que escribe en distintas colas distribuidas y luego tengo otros agentes que los consumen.
  - La transmisión se hace de manera diferente, así no se carga el extremo.



- Es ROBUSTO al ser distribuido, si se cae el sistema se pueden cargar los datos.

Importante las alertas: Si llegan muchas alertas no se sabe dónde está el problema, hay que tener un sistema de prioridades de alertas para saber cuáles son las alertas principales.