**Tema 4: Sistemas Gestores de Bases de Datos**

**[1.1] datos y archivos**

**[1.1.1]la necesidad de gestionar información**

En el mundo actual existe una cada vez mayor demanda de gestión de la información. No es una demanda nueva, todas las sociedades a lo largo de la historia han tenido esta necesidad. Desde el principio de los tiempos se ha necesitado disponer de herramientas que facilitaran la gestión de los datos ya que, como herramienta, el ser humano al principio sólo disponía de su memoria y cálculo inmediato y, como mucho, de la ayuda de sus dedos.

La escritura fue la herramienta que permitió al ser humano almacenar información y realizar cálculos sobre ella. Además de permitir compartir esa información entre diferentes personas, también posibilitó que los datos se guardaran de manera continua e incluso estuvieran disponibles para las siguientes generaciones. Los problemas actuales con la privacidad ya aparecieron con la propia escritura y así el cifrado de datos es una técnica tan antigua como la propia escritura para conseguir uno de los todavía requisitos fundamentales de la gestión de datos, la seguridad.

Cuanto más se han desarrollado las sociedades, mejores métodos se han desarrollado para gestionar la información y, a la vez, más datos se han necesitado gestionar. Para poder almacenar datos y cada vez más datos, el ser humano ideó nuevas herramientas: archivos, cajones, carpetas y fichas en las que se almacenaban los datos.

Antes de la aparición de las computadoras, el tiempo requerido para manipular estos datos era enorme. Sin embargo el proceso de aprendizaje era relativamente sencillo, ya que se usaban elementos que las personas manejaban desde su infancia.

Por esa razón, la informática se adaptó para que la terminología en el propio ordenador se pareciera a los términos de organización de datos clásicos. Así, en informática se sigue hablado de ficheros, formularios, carpetas, directorios,....

En estos últimos años, la demanda ha crecido a niveles espectaculares debido al acceso multitudinario a Internet y a los enormes flujos de información que generan los usuarios. Cada año la necesidad de almacenar información crece exponencialmente en un frenesí que, por ahora, no parece tener fin.

Desde la aparición de las primeras computadoras, se intentó automatizar la gestión de los datos. El propio nombre **Informática** hace referencia al hecho de ser una ciencia que trabaja con información. Por ello las bases de datos son una de las aplicaciones más antiguas de la informática.

**[1.1.2]datos e información**

Tradicionalmente siempre se ha separado el concepto de **dato** sobre el de **información**.

Un dato es una propiedad en *crudo*, es decir, sin contextualizar. Por ejemplo *Sánchez* y *32* son datos. Desde el punto de vista de la computación, ambos datos se pueden almacenar. Sin embargo, no podemos considerarlos información hasta que no les demos significado. Si decimos que *Sánchez* es mi primer apellido y que *32* son los grados centígrados de temperatura que hace ahora en la calle, esos datos pasan a ser información.

La información tiene una importancia humana, es interpretable, reconocible, presentable,.. El dato es simplemente el paso previo. Resumiendo: todo dato debe de ser procesado para obtener información, y es la información lo que nos interesa a los seres humanos.

Como veremos más adelante, los datos se convierten en información gracias a los **metadatos**: datos que sirven para describir otros datos.

**[1.1.3]Sistemas de Información**

**la empresa como sistema**

Según la RAE, la definición de sistema es *“Conjunto de cosas que ordenadamente relacionadas entre sí contribuyen a un determinado objeto”* .

La clientela fundamental del profesional de la informática es la empresa, sin distinguir entre la empresa privada y las entidades públicas. Una empresa está formada por diversos elementos que son comunes: el **capital**, los **recursos humanos**, los **inmuebles**, los **servicios** que presta, etc. Todos ellos forman el sistema de la empresa.

El sistema completo que forma la empresa, por otra parte, se suele dividir en los siguientes subsistemas:

* **Subsistema productivo.** También llamado subsistema real o físico. Representa la parte de la empresa encargada de gestionar la producción de la misma.
* **Subsistema financiero**. Encargado de la gestión de los bienes económicos de la empresa
* **Subsistema directivo**. Encargado de la gestión organizativa de la empresa

Hay que hacer notar que cada subsistema se asocia a un departamento concreto de la empresa.

**Sistemas de Información**

Los sistemas que aglutinan los elementos que intervienen para gestionar la información que manejan los subsistemas empresariales es lo que se conoce como Sistemas de Información. Se suele utilizar las siglas **SI** o **IS** (de *Information Server*) para referirse a este concepto.

Vale también para gestionar la información de cualquier sistema, aunque no sea empresarial. Pero la teoría está basada en el sistema empresarial.

Realmente un sistema de información sólo incluye aquello que nos interesa de la empresa y los elementos necesarios para gestionar esa información.

Un sistema de información genérico está formado por los siguientes elementos:

* **Recursos físicos.** Carpetas, documentos, equipamiento, discos,...
* **Recursos humanos.** Personal que maneja la información
* **Protocolo.** Normas que debe cumplir la información para que sea manejada (formato de la información, modelo para los documentos,...)

Las empresas necesitan implantar estos sistemas de información debido a la competencia que las obliga a gestionar de la forma más eficiente sus datos para una mayor calidad en la organización de las actividades de los subsistemas empresariales.

**componentes de un Sistema de Información Electrónico**

En el caso de una **gestión electrónica de la información** (lo que actualmente se considera un **sistema de información electrónico**), los componentes son:

* **Datos.** Se trata de la información relevante que almacena y gestiona el sistema de información. Ejemplos de datos son: *Sánchez*, *12764569F*, *Calle Mayo 5*, *Azul*…
* **Hardware.** Equipamiento físico que se utiliza para gestionar los datos. cada uno de los dispositivos electrónicos que permiten el funcionamiento del sistema de información: servidores, máquinas de los clientes, routers, switches, impresoras,…
* **Software.** Aplicaciones informáticas que se encargan de la gestión de la base de datos y de las herramientas que facilitan su uso.
* **Recursos humanos.** Personal que maneja el sistema de información.

**[1.2] tipos de gestión de información mediante el ordenador**

En la evolución de los sistemas de información ha habido dos puntos determinantes, que han formado los dos tipos fundamentales de sistemas de información electrónico.

**[1.2.1]sistemas de gestión de ficheros**

Este tipo de sistemas hace referencia a la forma que inicialmente se desarrolló en la informática para gestionar ficheros (y que aún se usa). En realidad, es una forma que traducía la manera clásica de gestionar sistemas de información (con sus archivadores, carpetas,…) al mundo electrónico.

A picture containing text, candle

Description automatically generated

La idea es que los datos se almacenan en ficheros y se crean aplicaciones (cuyo código posee la empresa que crea dichas aplicaciones) para acceder a los ficheros. Cada aplicación organiza los datos en los ficheros como le parece mejor y si incorporamos aplicaciones nuevas, estas usarán sus propios ficheros.

Cada aplicación almacena y utiliza sus propios datos de forma un tanto caótica. La ventaja de este sistema (la única ventaja), es que los procesos son independientes por lo que la modificación de uno no afecta al resto. Pero tiene grandes inconvenientes:

* **Programación de aplicaciones compleja.** Ya que los programadores se deben de encargar de lo que tiene que hacer la aplicación y además de estructurar los datos en disco.
* **Datos redundantes.** Ya que se repiten continuamente. Podría, por ejemplo, ocurrir que una segunda aplicación utilice datos de personales, que resulta que ya estaban almacenados en los ficheros de una primera aplicación, pero como ambas son independientes, los datos se repetirán.
* **Datos inconsistentes.** En relación con el problema anterior, yaque un proceso cambia sus datos y no los del resto. Por lo que la misma información puede tener distintos valores según qué aplicación acceda a él.
* **Difícil acceso a los datos.** Cada vez que se requiera una consulta no prevista inicialmente, hay que modificar el código de las aplicaciones o incluso crear una nueva aplicación. Esto hace imposible pensar en nuevas consultas e instantáneamente obtener sus resultados; inviable para aplicaciones que requieren grandes capacidades de consultas y análisis de datos.
* **Coste de almacenamiento elevado.** Al almacenarse varias veces el mismo dato, se requiere más espacio en los discos. Además, las aplicaciones también ocupan mucho al tener que pensar en todas las posibles consultas sobre los datos que la organización precisa.
* **Dependencia de los datos a nivel físico.** Para poder saber cómo se almacenan los datos, es decir qué estructura se utiliza de los mismos, necesitamos ver el código de la aplicación; es decir el código y los datos no son independientes.
* **Dificultad para el acceso simultáneo a los datos.** El acceso simultáneo requiere que varios usuarios al puedan acceder a la misma información. Con este tipo de sistemas es extremadamente difícil conseguir esta capacidad.
* **Dificultad para administrar la seguridad del sistema.** Ya que cada aplicación se crea independientemente. Es, por tanto, muy difícil establecer criterios de seguridad uniformes. Es decir, los permisos que cada usuario tiene sobre los datos, se establecen de forma muy confusa (y nada uniforme ya que cada aplicación puede variar la seguridad).

Se consideran también sistemas de gestión de ficheros, a los sistemas que utilizan programas ofimáticos (como **Word** o **Excel** por ejemplo) para gestionar sus datos. Esta última idea, la utilizan muchas pequeñas empresas para gestionar los datos, debido al presupuesto limitado del que disponen. Gestionar la información de esta forma produce los mismos (si no más) problemas.

**[1.2.2]Sistemas de Bases de Datos**

Estos serán los sistemas que estudiaremos en estos apuntes.

En este tipo de sistemas, los datos se centralizan en una **base de datos** común a todas las aplicaciones. Un software llamado **Sistema Gestor de Bases de Datos (SGBD)** es el que realmente accede a los datos y se encarga de gestionarlos. Las aplicaciones que creen los programadores, no acceden directamente a los datos, de modo que la base de datos es común para todas las aplicaciones.

De esta forma, hay, al menos, dos capas a la hora de acceder a los datos. Las aplicaciones se abstraen sobre la forma de acceder a los datos, dejando ese problema al SGBD. Así se pueden concentrar exclusivamente en la tarea de conseguir una interfaz de acceso a los datos para los usuarios.

Diagram

Description automatically generated

Cuando una aplicación modifica un dato, la modificación será visible inmediatamente para el resto de aplicaciones; ya que todas utilizarán la misma base de datos.

**ventajas**

* **Independencia de los datos y los programas.** Esto permite modificar los datos sin modificar el código de las aplicaciones y viceversa.
* **Menor redundancia**. Este modelo no requiere que los datos se repitan para cada aplicación que los requiera., en su lugar se diseñan los datos de forma independiente a las aplicaciones. Los programadores de aplicaciones deberán conocer la estructura creada para los datos y la forma en la que deben acceder a ellos.
* **Integridad de los datos**. Al estar centralizados, es más difícil que haya datos incoherentes. Es decir, que una aplicación muestre información distinta al resto de aplicaciones, ya que los datos son los mismos para todas.
* **Mayor seguridad en los datos.** El SGBD es el encargado de la seguridad y se puede centrar en ella de forma independiente a las aplicaciones. Como las aplicaciones deben atravesar la capa del SGBD para llegar a los datos, no se podrán saltar la seguridad.
* **Visiones distintas según el usuario.** Nuevamente, centralizar los datos facilita crear políticas que permitan que los usuarios vean la información de la base de datos de forma distinta.
* **Datos más documentados.** Las bases de datos tienen mucho mejor gestionados los **metadatos,** que permiten describir la información de la base de datos y que pueden ser consultados por las aplicaciones.
* **Acceso a los datos más eficiente.** Esta forma de organizar los datos produce un resultado más óptimo en rendimiento ya que los sistemas gestores centralizan el acceso pudiendo ejecutar políticas diferentes en función de la demanda.
* **Menor espacio de almacenamiento.** Puesto que hay muy poca redundancia.
* **Acceso simultáneo a los datos.** Nuevamente el SGBD tiene más capacidad de conseguir esto. Cuando hay varias aplicaciones que intentan acceder a los datos en los sistemas orientados a los ficheros, compiten por los datos y es fácil el bloqueo mutuo. En el caso de los sistemas orientados a bases de datos, toda petición pasa la capa del SGBD y esto permite evitar los bloqueos.

**desventajas**

* **Instalación costosa.** El control y administración de bases de datos requiere de un software y hardware poderoso.
* **Requiere personal cualificado.** Debido a la dificultad de manejo de este tipo de sistemas.
* **Implantación larga y difícil.** En relación a los puntos anteriores. La adaptación del personal y del equipamiento es mucho más complicada y lleva bastante tiempo.
* **Ausencia de estándares totales.** Lo cual significa una excesiva dependencia hacia los sistemas comerciales del mercado. Aunque, hoy en día, hay un funcionamiento base y un lenguaje de gestión (**SQL**) que desde hace tiempo se considera estándar (al menos en las bases de datos relacionales).

**[1.3] funcionamiento de un Sistema Gestor de Bases de Datos**

**[1.3.1]funciones. lenguajes de los SGBD**

Los SGBD tienen que realizar tres tipos de funciones para ser considerados válidos. A continuación se describen estas tres funciones.

**función de descripción o definición**

Permite al diseñador de la base de datos crear las estructuras apropiadas para integrar adecuadamente los datos. Se dice que esta función es la que permite definir las tres estructuras de la base de datos (relacionadas con los tres niveles de abstracción de las mismas).

* **Estructura interna**
* **Estructura conceptual**
* **Estructura externa**

Más adelante se explican estas tres estructuras, relacionadas con las tres formas (o niveles) de ver la base de datos.

Realmente la función de definición lo que hace es gestionar los **metadatos**. Los metadatos son la estructura de la dispone el sistema de base de datos para documentar cada dato. Los metadatos también son datos que se almacenan en la propia base de datos; pero su finalidad es describir los datos.

Un metadato nos permite para saber a qué información real se refiere cada dato. Por ejemplo: *Sánchez*, *Rodríguez* y *Crespo* son datos. Pero *Primer Apellido* es un metadato que, si está correctamente creado, nos permite determinar que *Sánchez*, *Rodríguez* y *Crespo* son primeros apellidos.

Dicho de otra forma, sin los metadatos, no podríamos manejar los datos como información relevante. Por ello son fundamentales. Son, de hecho, la base de la creación de las bases de datos.

Los metadatos pueden indicar cuestiones complejas. Por ejemplo, que de los *Alumnos* almacenamos su *dni* el cual lo forman 9 caracteres. Incluso podremos indicar que en el *dni* los 8 primeros son números y el noveno un carácter en mayúsculas que además cumple una regla concreta y sirve para identificar al alumno.

Por lo tanto, en realidad, **la función de definición sirve para crear, eliminar o modificar metadatos**.

Resumiendo: con la función de definición podremos:

* Especificar el significado de los datos
* Organizar la información en estructuras más complejas
* Relacionar los datos de forma precisa
* Especificar reglas especiales que deben cumplir los datos
* Crear todos los elementos estructurales de la base de datos (incluidos los usuarios)

Un lenguaje conocido como **lenguaje de descripción de datos** o **DDL**, es el que permite realizar la función de definición en las bases de datos.

**función de manipulación**

Permite cambiar y consultar los **datos** de la base de datos. Se realiza mediante un **lenguaje de modificación de datos** o **DML.** Mediante este lenguaje se puede:

* Añadirdatos
* Eliminardatos
* Modificardatos
* Consultar datos

Actualmente se suele diferenciar la **función de consulta de datos**, diferenciándola del resto de operaciones de manipulación de datos. Se habla de que la función de consulta se realiza con un **lenguaje de consulta de datos** o **DQL** (*Data Query Language*)**.**

**función de control**

Mediante esta función los administradores poseen mecanismos para proteger los datos. De manera que se permite a cada usuario ver ciertos datos y otros no, o bien usar ciertos recursos concretos de la base de datos y prohibir otros. Es decir, es la función encargada de establecer los permisos de acceso a los elementos que forman parte de la base de datos.

El lenguaje que implementa esta función es el **lenguaje de control de datos** o **DCL.**

**[1.3.2]utilidad de los sistemas gestores de bases de datos**

Un sistema gestor de bases de datos o **SGBD** (aunque se suele utilizar más a menudo en los libros especializados las siglas **DBMS** procedentes del inglés, *Data Base Management System*) es el software que permite a los usuarios procesar, describir, administrar y recuperar los datos almacenados en una base de datos.

En estos sistemas se proporciona un conjunto coordinado de programas, procedimientos y lenguajes que permiten a los distintos usuarios realizar sus tareas habituales con los datos, garantizando además la seguridad de los mismos.

El éxito del SGBD reside en mantener la seguridad e integridad de los datos. Lógicamente tiene que proporcionar herramientas a los distintos usuarios.

Además de las tres funciones principales comentadas anteriormente, hoy en día los SGBD son capaces de realizar numerosas operaciones. Para ello proporcionan numerosas herramientas, muchas de ellas permiten trabajar de una forma más cómoda con las . Las más destacadas son:

* **Herramientas para la creación y especificación del diccionario de datos**. El diccionario de datos es la estructura de la base de datos que almacena los metadatos. Es decir el diccionario de datos contiene la descripción de todos los datos de la base de datos.
* **Herramientas para administrar y crear la estructura física de la base de datos**. El SGBD proporciona herramientas para especificar la forma en la que se almacenarán los datos en la computadora (o computadoras) que alojen la base de datos. Estas herramientas nos permitirán diseñar una forma de almacenamiento centrada en optimizar el acceso a los datos.
* **Herramientas para la manipulación de los datos.** Nos permitirán añadir, modificar, suprimir o consultar datos (función de manipulación) de la forma más sencilla posible.
* **Herramientas de recuperación** en caso de desastre. Si ocurre un mal funcionamiento del sistema, un fallo en la alimentación del sistema, errores de red, etc. En ese caso los buenos SGBD poseen y proporcionan mecanismos para que se recupere la máxima información posible y se asegure su integridad.
* **Herramientas para la creación y restablecimiento de copias de seguridad.** Es una de las tareas fundamentales, ya que permite recuperar la información en caso de pérdida de datos.
* **Herramientas para la gestión de la comunicación** de la base de datos. Encargadas de configurar el hardware y software de conexión a la red. Así como los mecanismos necesarios para configurar adecuadamente el software que se encarga de recibir y comunicar las peticiones de los clientes.
* **Herramientas para la creación de aplicaciones de usuario**. Es decir, herramientas para los programadores de aplicaciones, los cuales crean el software con el que los usuarios accederán de forma cómoda a la base de datos.
* **Herramientas de instalación y configuración** de la base de datos.
* **Herramientas para la exportación e importación** de datos a o desde otros sistemas.
* **Herramientas para gestionar la seguridad**. Permiten establecer privilegios y permisos diferentes para los usuarios, así como impedir el acceso no deseado (función de control).

**[1.3.3]niveles de abstracción de una base de datos**

**introducción**

En cualquier software siempre hay dos puntos de vista:

* **Nivel externo**. Esta es la visión del software que tienen los usuarios
* **Nivel interno**. Visión de los creadores del software, que determina su forma de funcionar.

Esta separación distingue al usuario o usuaria, del programador o programadora que ha creado la aplicación, y es crucial que sea así. La persona que utiliza el software evita tener que Del mismo modo una casa se la puede observar desde el punto de vista del inquilino de la misma o bien de las personas que la construyeron. Los primeros ven la función real de la misma y los constructores nos podrán hablar de los materiales empleados por ejemplo.

En el caso de las bases de datos hay más niveles, más formas de observar la base de datos y estos niveles son manejados por los distintos usuarios de la base de datos. A eso se le llama los niveles de abstracción porque nos permite efectivamente abstraernos para observar la base de datos en base a diferentes intereses. Los usuarios podrán entender la base de datos sin conocer los entresijos técnicos y los administradores podrán trabajar con base de datos sin conocer la forma en la que los usuarios realmente añaden los datos.

Los niveles habituales son:

* **Nivel físico**. Nos permite saber la forma en la que está almacenada la base de datos. Por ejemplo en qué discos duros, qué archivos utiliza, de qué tipo son los archivos, bajo qué sistema operativo,… Este nivel es el que está más cercano a la visión de la base de datos que posee la computadora, por lo que es absolutamente dependiente del hardware y el software (especialmente del Sistema Operativo).
* **Nivel interno**. Un poco más cercano a la visión que tenemos las personas. Permite observar la base de datos como un conjunto de estructuras que relacionan la información humana con la información digital. A este nivel no se depende del hardware concreto que tengamos; es decir, no se habla de discos, servidores, archivos,… sino de las estructuras que disponemos en nuestro SGBD en particular para organizar los datos.
* **Nivel conceptual**. Es el nivel de mayor abstracción y el más importante. Se trata de una visión organizativa de los datos independiente tanto del hardware como del software que tengamos. Es el plano o modelo general de la base de datos y a este nivel es al que trabajan las o los analistas y diseñadores cuando crean el primer esquema de la base de datos. En ningún momento queda influido por el SGBD en particular que usemos.
* **Nivel externo**. Se trata de la visión de los datos que poseen los usuarios y usuariasfinales de la base de datos**.** Esa visión es la que obtienen a través de las aplicaciones. Las aplicaciones creadas por los desarrolladores abstraen la realidad conceptual de modo que el usuario no conoce las relaciones entre los datos, como tampoco conoce dónde realmente se están almacenando los datos. Es la forma en la que cualquier persona desea manejar una base de datos a través de formularios, informes, listas,…

La idea de estos niveles procede de la normalización hecha en el modelo **ANSI/X3/SPARC** (Véase [[a2] estándares y modelo ANSI en la página 46](https://jorgesanchez.net/manuales/gbd/sgbd.html#_idTextAnchor029)) y sigue estando muy presente en la gestión actual de las bases de datos.

Este modelo dictó que podemos pasar de unos modelos a otros de manera casi automática utilizando un software adecuado. El modelo ANSI llama a ese software **procesador de modelos** y hoy en día es lo que se conoce como **herramientas CASE (Computer Aided for Software Engineering,** *Asistente Computerizado para Ingeniería del Software*). Para cada nivel se realizan esquemas relacionados con ellos. Así hay **esquemas externos** (varios), **esquema conceptual**, **esquema interno** y **esquema físico** que forman todos los aspectos de la base de datos.

Diagram

Description automatically generated

En la imagen se observa la distancia que poseen los usuarios de la base de datos respecto a la realidad física de la base de datos (representada con el cilindro). La física son los datos en crudo, es decir en formato binario dentro del disco o discos que los contienen. El esquema físico es el que se realiza pensando más en esa realidad y los esquemas externos los que se crean pensando en la visión de los usuarios.

Las dos columnas que aparecen en la imagen reflejan dos fronteras a tener en cuenta:

* **Independencia Lógica**. Los esquemas de los niveles conceptual y externo son independientes del software concreto de base de datos que usemos; no dependen en absoluto de él. Por ello esos esquemas nos valdrían para cualquier SGBD que utilicemos.
* **Independencia Física**. La da la barrera entre el esquema físico y el interno e indica que el esquema interno es independiente del hardware concreto que usemos. El esquema físico se diseña en base a un hardware concreto, pero él interno no. Eso permite concentrarse en detalles más conceptuales.

**[1.3.4]recursos humanos de las bases de datos**

Intervienen (como ya se ha comentado) muchas personas en el desarrollo y manipulación de una base de datos. Se describen, a continuación, los actores más importantes.

**informáticos**

Lógicamente, son los profesionales que definen y preparan la base de datos. Pueden ser:

* **Directivos/as.** Organizadores y coordinadores del proyecto a desarrollar y máximos responsables del mismo. Esto significa que son los encargados de decidir los recursos que se pueden utilizar, planificar el tiempo y las tareas, la atención al usuario y de dirigir las entrevistas y reuniones pertinentes.

Son especialistas en gestión de recursos, tanto materiales como humanos.

* **Analistas**. Son los encargados de controlar el desarrollo de la base de datos aprobada por la dirección. Dirigen a los desarrolladores y operadores. Normalmente son, además, los **diseñadores de la base de datos**: es decir, crean el esquema conceptual de la misma.
* **Administradores/as de las bases de datos.** Encargados de crear el esquema interno de la base de datos. También gestionan el correcto funcionamiento del SGBD. Sus tareas incluyen la planificación de copia de seguridad, gestión de usuarios y permisos, optimización del rendimiento, monitorización de problemas y creación de los objetos de la base de datos.
* **Desarrolladores/as o programadores/as.** Encargados de la realización de las aplicaciones de usuario para que estos accedan a la base de datos.
* **Equipo de mantenimiento.** También se les llama **operadores**. Encargados de dar soporte a los usuarios en el trabajo diario (suelen incorporar además tareas administrativas como la creación de copias de seguridad por ejemplo o el arreglo de problemas de red por ejemplo).

**usuarios**

* **Expertos/as.** Realizan operaciones avanzadas sobre la base de datos. Normalmente conocen el lenguaje de manipulación de datos (**DML**) para acceder a la base de datos. Son usuarios, por lo tanto, con conocimientos informáticos que se encargan en las empresas de los clientes de algunas acciones más complejas sobre la base de datos que las que realizan los usuarios habituales.
* **Habituales.** Utilizan las aplicaciones creadas por los desarrolladores para consultar y actualizar los datos. Son los que trabajan en la empresa a diario con estas herramientas y el objetivo fundamental de todo el desarrollo de la base de datos.
* **Ocasionales**. Sonusuarios que utilizan un acceso mínimo a la base de datos a través de una aplicación que permite consultar ciertos datos. Serían por ejemplo los usuarios que consultan el horario de trenes a través de Internet. Aunque se les llama ocasionales son el núcleo del trabajo con la base de datos ya que son los que más la utilizan (ya que son sus usuarios más numerosos) y son, por ejemplo, los que visitan la base de datos para realizar compras o para informarse del negocio representado en la base de datos.

Diagram

Description automatically generated

**[1.3.5]proceso de creación y manipulación de una base de datos**

**fase de creación:**

[1]El **analista** o **diseñador** crea el esquema conceptual. En muchas ocasiones, utilizando una **herramienta CASE** para diseñar el esquema de forma más cómoda.

[2]El **administrador** de la base de datos (**DBA**) recoge ese esquema y crea el esquema interno de la base de datos. También se encarga, previamente, de determinar el SGBD idóneo y de configurar el software del SGBD así como de establecer las políticas de copia de seguridad.

[3]Los **desarrolladores** también recogen el esquema conceptual y utilizan las aplicaciones necesarias para generar los esquemas externos, que realmente se traducirán en programas y aplicaciones, que necesitan los usuarios.

**fase de manipulación:**

Ocurre con la base de datos ya creada y en funcionamiento.

[1]El usuario realiza una operación sobre la base de datos (una consulta, modifica o añade un dato, etc.)

[2]Las aplicaciones las traducen a su forma conceptual utilizando el diccionario de datos, que posee todos los metadatos necesarios.

[3]El esquema conceptual es traducido por la SGBD a su forma interna, nuevamente con ayuda del Diccionario de Datos.

[4]EL SGBD se comunica con el Sistema Operativo para pedir que acceda al disco (estamos, por lo tanto ya en el nivel físico) y recoja los datos requeridos (siempre con ayuda del Diccionario de Datos).

[5]El Sistema Operativo accede al almacenamiento físico correspondiente y devuelve los datos al SGBD.

[6]El SGBD transforma los datos internos en datos conceptuales y los entrega a la aplicación.

[7]La aplicación muestra los datos habiéndolos traducido a una forma (externa) amigable y apta para ser entregada al usuario que hizo la petición.

**[1.3.6]estructura multicapa**

Diagram

Description automatically generated

El proceso que realiza un SGBD para acceder a los datos está en realidad formado por varias capas que actúan como interface. El usuario nunca accede a los datos directamente. Estas capas son las que consiguen implementar los niveles de abstracción de la base de datos.

Fue el propio organismo ANSI (en su modelo ANSI/X3/SPARC) la que introdujo una mejora de su modelo de bases de datos en 1988 a través de un grupo de trabajo llamado **UFTG** (*User Facilities Task Group,* grupo de trabajo para las facilidades de usuario). Este modelo toma como objeto principal al usuario habitual de la base de datos y modela el funcionamiento de la base de datos en una sucesión de capas cuya finalidad es ocultar y proteger la parte interna de las bases de datos.

Desde esta óptica, para llegar a los datos hay que pasar una serie de capas que desde la parte más externa poco a poco van entrando más en la realidad física de la base de datos. Esa estructura se muestra en la [Ilustración 5](https://jorgesanchez.net/manuales/gbd/sgbd.html).

Este marco sigue teniendo vigencia actualmente e indica que el acceso a los datos no es instantáneo, que los datos están protegidos de los usuarios que pasan (sin saberlo) por varias capas de proceso antes de que sus peticiones a la base de datos sean atendidas.

Se explican las capas en detalle

**aplicaciones de usuario**

Es la capa a la que acceden los usuarios. Proporciona el SGBD a los usuarios un acceso más sencillo a los datos. Son, en definitiva, las páginas web y los programas con las que los usuarios manejan la base de datos. Permite abstraer la realidad de la base de datos a las usuarias y usuarios, mostrando la información de una forma más humana.

**capa de acceso a datos**

La capa de acceso a datos es la que permite comunicar a las aplicaciones de usuario con el diccionario de datos. Es un software (un *driver* o controlador, en realidad) que se encarga traducir las peticiones del usuario para que lleguen de forma correcta a la base de datos y ésta pueda responder de forma adecuada.

**diccionario de datos**

Se trata de una estructura interna del SGBD que contiene todos los metadatos. Esta estructura es la que permite pasar de un nivel a otro.

núcleo

El núcleo de la base de datos es la capa encargada de traducir todas las instrucciones requeridas y prepararlas para su correcta interpretación por parte del sistema. Realiza la traducción física de las peticiones.

**sistema operativo**

Es una capa externa al software SGBD pero es la única capa que realmente accede a los datos en sí. En realidad los SGBD no acceden directamente al disco, sino que piden al Sistema Operativo que lo haga, ya que es el que maneja el sistema de discos.

**[1.3.7]funcionamiento del SGBD**

La imagen siguiente, presenta el funcionamiento típico de un SGBD. En ella se reproduce la comunicación entre un usuario que desea acceder a los datos y el SGBD:

[1]Los usuarios utilizan una aplicación para acceder a los datos. Estamos en el nivel externo de la base de datos, por lo que la propia aplicación traduce la petición que hizo el usuario de forma sencilla, a una petición entendible por la capa de acceso a los datos.

[2]El proceso cliente es el software de acceso a la base de datos y que está instalado en el lado del cliente. Se encarga simplemente de recoger y enviar la petición (comprobando antes si hay comunicación con el servidor de la base de datos).

Diagram

Description automatically generated

[3]A través de la red (normalmente) el proceso cliente se comunica con el proceso servidor, que es el software de comunicación instalado en el lado del servidor. Ambos procesos (cliente y servidor) forman la capa de acceso a los datos.

[4]Estando ya en el servidor, la petición pasa al software del Sistema Gestor de Bases de Datos (habrá aquí, como se ha visto en el apartado anterior una traducción de datos, desde el nivel externo al nivel interno).

[5]El SGBD, comprobando el diccionario de datos, comprueba si la petición es correcta.

[6]El SGBD también revisa el diccionario de datos (si la petición es correcta) para saber con exactitud en qué archivos y en qué parte dentro de ellos, se encuentran los datos requeridos

[7]Con la información sobre dónde están los datos, el SGBD hace una petición al Sistema Operativo, que es el que tiene capacidad realmente de acceder a los archivos de datos. Por ello la petición del SGBD se traduce al formato utilizado por el Sistema Operativo.El Sistema Operativo accede a los datos.

[8]El Sistema Operativo recibe los datos.

[9]Se entregan los datos al Sistema Gestor de Bases de Datos o, si ha habido un error al acceder a los datos, se indica el error ocurrido.

[10]El SGBD traduce los datos a una forma más conceptual y se los entrega al proceso servidor.

[11]Los datos se entregan al proceso cliente.

[12]Los datos llegan a la aplicación.

[13]La aplicación de usuario traduce los datos recibidos en información presentada de la forma más conveniente para el usuario.

**[1.3.8]formas de ejecución de un SGBD**

**SGBD monocapa**

Se trata de Sistemas Gestores instalados en una máquina desde la que se conectan los propios usuarios y administradores. Es decir, todo el sistema está en una sola máquina.

Graphical user interface, website

Description automatically generated

Es un modelo que sólo se utiliza con bases de datos pequeñas y poca cantidad de conexiones. La popular **Access** de Microsoft es considerada un sistema gestor monocapa (aunque tiene algunas posibilidades para utilizar en dos capas).

**SGBD bicapa**

A picture containing graphical user interface

Description automatically generated

Usa un modelo de funcionamiento tipo **cliente/servidor**. La base de datos y el sistema gestor se alojan en un servidor al cual se conectan los usuarios desde máquinas clientes. Un software de comunicaciones se encarga de permitir el acceso a través de la red. Los clientes deben instalar el software cliente de acceso según las instrucciones de configuración del administrador.

Hay dos posibilidades:

* Arquitectura **cliente/servidor único**. Un solo servidor gestiona la base de datos, todos los clientes se conectan a él para realizar las peticiones a la base de datos.
* Arquitectura **cliente/multiservidor**. La base de datos se distribuye entre varios servidores. El cliente no sabe realmente a qué servidor se conecta; el software de control de comunicaciones se encargará de dirigir al usuario al servidor adecuado. De forma lógica, es como si se tratara de un solo servidor aunque físicamente sean muchos (el cliente no percibe que haya más de un servidor).

**SGBD de tres o más capas**

En este caso entre el cliente y el servidor hay al menos una capa intermedia (puede haber varias). Esa capa (o capas) se encarga de recoger las peticiones de los clientes y luego de comunicarse con el servidor (o servidores) de bases de datos para recibir la respuesta y enviarla al cliente.

El caso típico es que la capa intermedia sea un servidor web, que recibe las peticiones a través de aplicaciones web; de este modo para conectarse a la base de datos, el usuario solo requiere un navegador web, que es un software muy habitual en cualquier máquina y por lo tanto no requiere una instalación de software adicional en la máquina cliente.

Graphical user interface

Description automatically generated

Este modelo es el que más se está potenciando en la actualidad por motivos de seguridad y portabilidad de la base de datos.

**[1.4] tipos de SGBD**

**[1.4.1]introducción**

Como se ha visto en los apartados anteriores, resulta que cada SGBD puede utilizar un modelo diferente para los datos. Por lo que hay modelos conceptuales diferentes según que SGBD utilicemos. Esto da lugar a un diagrama de trabajo para los profesionales de la base de datos que permite saber qué esquemas hay que realizar (y en qué orden) para crear una base de datos.

Diagram

Description automatically generated

El punto de partida es el uso en el mundo real que tendrá la base de datos. Ese punto es en el que están los usuarios y es crucial tenerle muy claro. El punto final es el almacenamiento físico de la base de datos.

En este esquema aparece el llamado **Esquema lógico**, que permite pasar de forma más gradual del esquema conceptual al esquema interno.

No obstante existen modelos lógicos comunes, ya que hay SGBD de diferentes tipos. En la realidad el modelo conceptual clásico se modifica para que existan dos modelos internos: el modelo lógico (referido a cualquier SGBD de ese tipo) y el modelo conceptual propiamente interno (aplicable sólo a un SGBD en particular). De hecho, en la práctica, al definir las bases de datos desde el mundo real hasta llegar a los datos físicos se pasa por todos los esquemas señalados en la [Ilustración 10](https://jorgesanchez.net/manuales/gbd/sgbd.html).

Por lo tanto la diferencia entre los distintos SGBD está en que proporcionan diferentes modelos lógicos.

**diferencias entre el modelo lógico y el conceptual**

* El modelo conceptual es independiente del DBMS que se vaya a utilizar. El lógico depende de un **tipo** de SGBD en particular
* El modelo lógico está más cerca del modelo físico, el que utiliza internamente el ordenador
* El modelo conceptual es el más cercano al usuario, el lógico es el encargado de establecer el paso entre el modelo conceptual y el modelo físico del sistema.

Algunos ejemplos de modelos conceptuales son:

* **Modelo Entidad Relación**
* **Modelo RM/T**
* **Modelo UML**

Ejemplos de modelos lógicos son:

* **Modelo relacional**
* **Modelo Codasyl**
* **Modelo Jerárquico**

A continuación se comentarán los modelos lógicos más importantes.

**[1.4.2]modelo jerárquico**

Era utilizado por los primeros SGBD, desde que IBM lo definió para su IMS (*Information Management System,* Sistema Administrador de Información) en 1970. Se le llama también modelo en árbol debido a que utiliza una estructura en árbol para organizar los datos.

La información se organiza con un jerarquía en la que la relación entre las entidades de este modelo siempre es del tipo **padre / hijo**. De esta forma hay una serie de nodos que contendrán atributos y que se relacionarán con nodos hijos de forma que puede haber más de un hijo para el mismo padre (pero un hijo sólo tiene un padre).

Graphical user interface, application, website

Description automatically generated

Los datos de este modelo se almacenan en estructuras lógicas llamadas **segmentos**. Los segmentos se relacionan entre sí utilizando **arcos.**

La forma visual de este modelo es de árbol invertido, en la parte superior están los padres y en la inferior los hijos.

Este esquema está en absoluto desuso ya que no es válido para modelar la mayoría de problemas de bases de datos. Su virtud era la facilidad de manejo ya que sólo existe un tipo de relación (padre/hijo) entre los datos; su principal desventaja es que no basta para representar la mayoría de relaciones. Además no mantenía la independencia con la física de la base de datos.

**[1.4.3]modelo en red (Codasyl)**

Es un modelo que ha tenido una gran aceptación (aunque apenas se utiliza actualmente). En especial se hizo popular la forma definida por el estándar Codasyl a principios de los 70 que se convirtió en el modelo en red más utilizado.

Graphical user interface, application

Description automatically generated

El modelo en red organiza la información en **registros** (también llamados **nodos**) y **enlaces**. En los registros se almacenan los datos, mientras que los enlaces permiten relacionar estos datos. Las bases de datos en red son parecidas a las jerárquicas sólo que en ellas puede haber más de un padre.

En este modelo se pueden representar perfectamente cualquier tipo de relación entre los datos (aunque el Codasyl restringía un poco las relaciones posibles), pero hace muy complicado su manejo.

Poseía un lenguaje poderoso de trabajo con la base de datos. El problema era la complejidad para trabajar con este modelo tanto para manipular los datos como programar aplicaciones de acceso a la base de datos. Tampoco mantenía una buena independencia con la física de la base de datos.

**[1.4.4]modelo relacional**

Es el modelo más popular. Los datos se organizan en tablas y estas en columnas y filas de datos. Las tablas se relacionan entre sí para ligar todos los datos.

Se basa en la teoría de conjuntos y consigue una gran separación entre lo conceptual y lo físico, consiguiendo su total independencia. Tiene un lenguaje considerado estándar, el SQL y una enorme red de usuarios y documentación que facilita su aprendizaje. Además dota de una gran facilidad para establecer reglas complejas a los datos.

El problema es que la simplicidad de manejo y la independencia que consigue se logra a base de un software muy complejo que requiere también un hardware poderoso.

**[1.4.5]modelo de bases de datos orientadas a objetos**

Desde la aparición de la programación orientada a objetos (**POO** u **OOP**) se empezó a pensar en bases de datos adaptadas a estos lenguajes. La programación orientada a objetos permite cohesionar datos y procedimientos, haciendo que se diseñen estructuras que poseen datos (**atributos**) en las que se definen los procedimientos (**operaciones**) que pueden realizar con los datos. En las bases orientadas a objetos se utiliza esta misma idea.

A través de este concepto se intenta que estas bases de datos consigan arreglar las limitaciones de las relacionales. Por ejemplo el problema de la herencia (el hecho de que no se puedan realizar relaciones de herencia entre las tablas), tipos definidos por el usuario, disparadores (triggers) almacenables en la base de datos, soporte multimedia...

Se supone que son las bases de datos de tercera generación (la primera fue las bases de datos en red y la segunda las relacionales), lo que significa que el futuro parece estar a favor de estas bases de datos. Pero siguen sin reemplazar a las relacionales, aunque son el tipo de base de datos que más está creciendo en los últimos años.

Su modelo conceptual se suele diseñar usando la notación **UML** y el lógico usando **ODMG** (*Object Data Management Group,* grupo de administración de objetos de datos), organismo que intenta crear estándares para este modelo.

Sus ventajas están en el hecho de usar la misma notación que la de los programas (lo que facilita la tarea de su aprendizaje a los analistas y desarrolladores) y que el significado de los datos es más completo. Lo malo es que no posee un lenguaje tan poderoso como el modelo relacional para manipular datos y metadatos, que tiene más dificultades para establecer reglas a los datos y que al final es más complejo para manejar los datos.

**[1.4.6]bases de datos objeto-relacionales**

Tratan de ser un híbrido entre el modelo relacional y el orientado a objetos. El problema de las bases de datos orientadas a objetos es que requieren reinvertir capital y esfuerzos de nuevo para convertir las bases de datos relacionales en bases de datos orientadas a objetos. En las bases de datos objeto relacionales se intenta conseguir una compatibilidad relacional dando la posibilidad de integrar mejoras de la orientación a objetos.

Estas bases de datos se basan en el estándar **ISO** **SQL 2000** y los siguientes. En ese estándar se añade a las bases relacionales la posibilidad de almacenar procedimientos de usuario, triggers, tipos definidos por el usuario, consultas recursivas, bases de datos OLAP, tipos LOB,...

Las últimas versiones de la mayoría de las clásicas grandes bases de datos relacionales (**Oracle**, **SQL Server**, **DB2**, ...) son objeto relacionales.

**[1.4.7]bases de datos NoSQL**

En los últimos años ha aparecido todo un género de bases de datos (de varios tipos) que intentan paliar deficiencias detectadas en el modelo relacional.

El dominio de este modelo parecía demostrar, durante décadas, que era el tipo ideal de base de datos. El cambio de perspectiva se ha producido por la altísima demanda de servicios que requiere Internet. En especial si lo que se requiere es escribir o modificar datos, ya que actualmente todos los usuarios de Internet crean muchísimos datos cada día que requieren ser almacenados inmediatamente (el caso más claro es el de las redes sociales).

Con este panorama han aparecido nuevos tipos de bases de datos y se han modificado y actualizado tipos antiguos que ahora parecen útiles. Lo que aportan la mayoría de estos tipos de bases de datos, es el uso de otro tipo de esquemas conceptuales e internos más apropiados para este tipo de demandas de usuario.

En resumen las bases de datos NoSQL renuncian al modelo relacional para paliar las carencias del modelo relacional en estos aspectos:

* Aceptar un enorme cantidad peticiones de consulta y especialmente de modificación de datos por minuto
* Gestionar datos muy heterogéneos (irregulares, con tipos de datos cambiantes)
* Gestionar datos que se relacionan de manera muy compleja
* Usar otros lenguajes (diferentes a SQL), más aptos para otras tareas

Esto no significa que cada base de datos NoSQL sea capaz de mejorar en todos los aspectos anteriores, cada tipo de base de datos NoSQL está pensado para algunos de los puntos anteriores.

**[a2] estándares y modelo ANSI**

Es uno de los aspectos que todavía sigue pendiente. Desde la aparición de los primeros gestores de base de datos se intentó llegar a un acuerdo para que hubiera una estructura común para todos ellos, a fin de que el aprendizaje y manejo de este software fuera más provechoso y eficiente.

El acuerdo nunca se ha conseguido del todo, no hay estándares aceptados del todo. Aunque sí hay unas cuentas propuestas de estándares que sí funcionan como tales.

**[a2.1]organismos de estandarización**

Los intentos por conseguir una estandarización han estado promovidos por organismos de todo tipo. Algunos son estatales, otros privados y otros promovidos por los propios usuarios. Los tres que han tenido gran relevancia en el campo de las bases de datos son **ANSI/SPARC/X3**, **CODASYL** y **ODMG** (éste sólo para las bases de datos orientadas a objetos). Los organismos grandes (que recogen grandes responsabilidades) dividen sus tareas en comités, y éstos en grupos de trabajo que se encargan de temas concretos.

**[a2.2]ISO/JTC1/SC21/WG3**

* **ISO** *(International Organization for Standardization)*. Es un organismo internacional de definición de estándares de gran prestigio**.**
* **IEC** *(International Electrotechnical Commission).* Organismo de definición de normas en ambientes electrónicos. Es la parte, en definitiva de ISO, dedicada a la creación de estándares.
* **JTC 1** *(Joint Technical Committee).* Comité parte de IEC dedicado a la tecnología de la información (informática). En el campo de las bases de datos, el subcomité **SC 21** (en el que participan otros organismos nacionales, como el español AENOR) poseeun grupo de trabajollamado **WG 3** que se dedica a las bases de datos. Este grupo de trabajo es el que define la estandarización del lenguaje SQL entre otras cuestiones.

Entre los trabajos que realiza el grupo WG3 está la normalización de **SQL**, además de otras normas de estandarización.

**[a2.3]DBTG/Codasyl**

Codasyl (*COnference on DAta SYstem Languages***)** es el nombre de una conferencia iniciada en el año 1959 y que dio lugar a un organismo con la idea de conseguir un lenguaje estándar para la mayoría de máquinas informáticas. Participaron organismos privados y públicos del gobierno de Estados Unidos con la finalidad de definir estándares. Su primera tarea fue desarrollar el lenguaje **COBOL** y otros elementos del análisis, diseño y la programación de ordenadores.

La tarea real de estandarizar esos lenguajes se la cedieron al organismo ANSI, pero las ideas e inicios de muchas tecnologías se idearon en el consorcio Codasyl.

En 1967 se crea un **grupo de tareas para bases de datos** *(Data Base Task Group*) y este grupo definió el **modelo en red de bases de datos** y su integración con COBOL. A este modelo en red se le denomina **modelo Codasyl** o modelo **DBTG** y fue finalmente aceptado por la ANSI.

**[a2.4]ANSI/X3/SPARC**

**ANSI** (*American National Standards Institute*) es un organismo científico de Estados Unidos que ha definido diversos estándares en el campo de las bases de datos. **X3** es la parte de ANSI encargada de los estándares en el mundo de la electrónica. Finalmente **SPARC,** *System Planning and Repairments Committee*, comité de planificación de sistemas y reparaciones es una subsección de X3 encargada de los estándares en Sistemas Informáticos en especial del campo de las bases de datos. Su logro fundamental ha sido definir un modelo de referencia para las bases de datos (que se estudiará posteriormente).

Graphical user interface, application, Teams

Description automatically generated

En la actualidad ANSI para Estados Unidos e ISO para todo el mundo son nombres equivalentes en cuanto a estandarización de bases de datos, puesto que se habla ya de un único modelo de sistema de bases de datos.

**[a2.5]modelo ANSI/X3/SPARC**

El organismo ANSI ha marcado la referencia para la construcción de SGBD. El modelo definido por el grupo de trabajo SPARC se basa en estudios anteriores en los que se definían tres niveles de abstracción necesarios para gestionar una base de datos. ANSI profundiza más en esta idea y define cómo debe ser el proceso de creación y utilización de estos niveles.

En el modelo ANSI se indica que hay tres modelos: **externo**, **conceptual** e **interno.** Se entiende por modelo, el conjunto de normas que permiten crear esquemas (diseños de la base de datos).

Los esquemas externos reflejan la información preparada para el usuario final, el esquema conceptual refleja los datos y relaciones de la base de datos y el esquema interno la preparación de los datos para ser almacenados.

El esquema conceptual contiene la información lógica de la base de datos. Su estructuración y las relaciones que hay entre los datos.

El esquema interno contiene información sobre cómo están almacenados los datos en disco. Es el esquema más cercano a la organización real de los datos.

En definitiva el modelo ANSI es una propuesta teórica sobre cómo debe de funcionar un sistema gestor de bases de datos (sin duda, la propuesta más importante). Su idea es la siguiente:

Diagram

Description automatically generated

Ilustración 18. Niveles en el modelo ANSI

En la [Ilustración 18](https://jorgesanchez.net/manuales/gbd/sgbd.html), el paso de un esquema a otro se realiza utilizando una interfaz o función de traducción. En su modelo, la ANSI no indica cómo se debe realizar esta función, sólo que debe existir.

La arquitectura completa ([Ilustración 19](https://jorgesanchez.net/manuales/gbd/sgbd.html)) está dividida en dos secciones, la zona de definición de datos y la de manipulación. Esa arquitectura muestra las funciones realizadas por humanos y las realizadas por programas.

En la fase de **definición**, una serie de interfaces permiten la creación de los **metadatos** que se convierten en el eje de esta arquitectura. La creación de la base de datos comienza con la elaboración del esquema conceptual realizándola el administrador de la empresa (actualmente es el diseñador, pero ANSI no lo llamó así). Ese esquema se procesa utilizando un procesador del esquema conceptual (normalmente una herramienta **CASE**, *interfaz 1* del dibujo anterior) que lo convierte en los metadatos (*interfaz 2*).

La *interfaz 3* permite mostrar los datos del esquema conceptual a los otros dos administradores: el administrador de la base de datos y el de aplicaciones (el desarrollador). Mediante esta información construyen los esquemas internos y externos mediante las *interfaces 4 y 5* respectivamente, los procesadores de estos esquemas almacenan la información correspondiente a estos esquemas en los metadatos (*interfaces 6 y 7*).

Diagram

Description automatically generated

En la fase de **manipulación** el usuario puede realizar operaciones sobre la base de datos usando la *interfaz 8* (normalmente una aplicación) esta petición es transformada por el transformador externo/conceptual que obtiene el esquema correspondiente ayudándose también de los metadatos (*interfaz 9*). El resultado lo convierte otro transformador en el esquema interno (*interfaz 10*) usando también la información de los metadatos (*interfaz 11*). Finalmente del esquema interno se pasa a los datos usando el último transformador (*interfaz 12*) que también accede a los metadatos (*interfaz 13*) y de ahí se accede a los datos (*interfaz 14*). Para que los datos se devuelvan al usuario en formato adecuado para él se tiene que hacer el proceso contrario (observar dibujo).