ACCESO A DATOS

UT1 – MANEJO DE FICHEROS


API
Memoria principal
Almacenamiento secundario
Guardar
Datos transitorios
Recuperar
Crear
Programa
de aplicación


Registros
Caché
Nivel1
Caché Nivel2
Memoria Principal
(DRAM)
Memoria Secundaria (Discos
Duros)
Almacenamiento Terciario (Sistemas
de almacenamiento distribuidos)

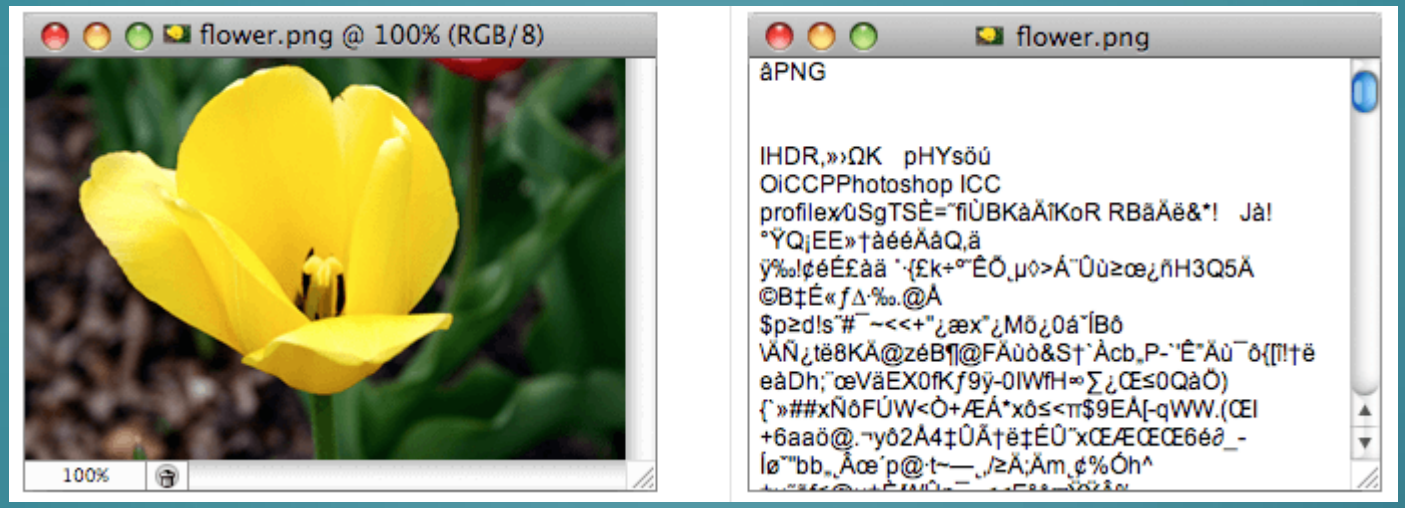
¿QUÉ ES UN FICHERO?

Conjunto de bits almacenados en un dispositivo (disco duro)

- Persistencia de los datos → no volátil
- Fichero = Ruta/path + nombre del fichero (+ extensión)
- Un único nombre idéntico en el directorio

¿QUÉ ES UN FICHERO?

- Ruta absoluta
  - C:/AccesoDatos/tema1/ejercicio.txt
- Ruta relativa
  - . /tema1/ejercicio.txt



Ficheros binarios

- Contenido codificado en binario. Pueden ser ficheros alfabéticos, numéricos, imagen, audio, vídeo, multimedia…

Ficheros de texto

- Ficheros con caracteres, nos permiten leer y escribir la información que contengan

ENCODING

Sistema utilizado para transformar los caracteres que usa cada lenguaje en un símbolo que un ordenador pueda interpretar.

- ASCII
- Unicode
- ISO-8859

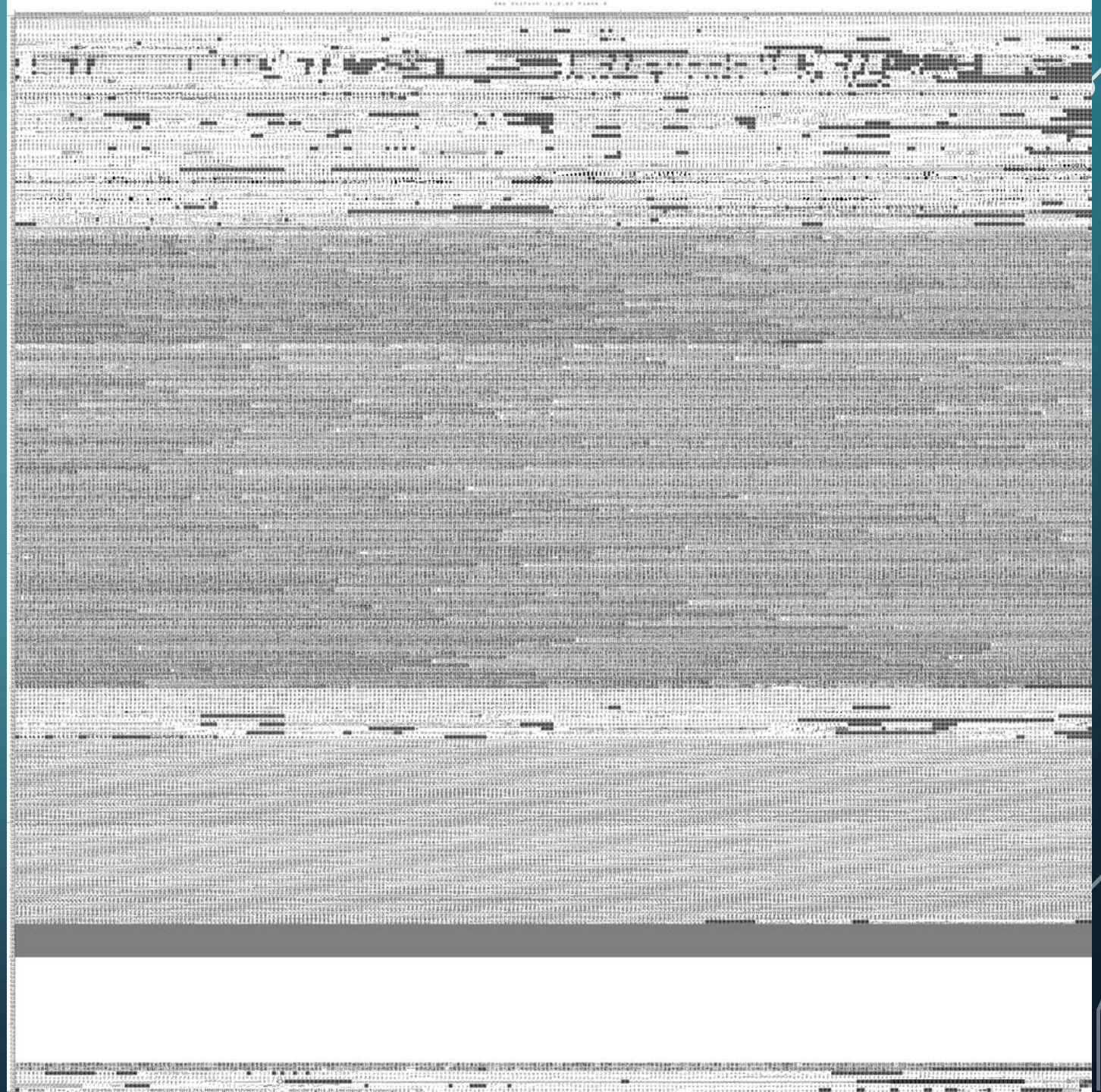
ENCODING

- ASCII
- UTF-8

Byte	Cod.	Char	Byte	Cod.	Char	Byte	Cod.	Char	Byte	Cod.	Char
00000000	0	Null	00100000	32	Spc	01000000	64	@	01100000	96	`
00000001	1	Start of heading	00100001	33	!	01000001	65	A	01100001	97	a
00000010	2	Start of text	00100010	34	"	01000010	66	B	01100010	98	b
00000011	3	End of text	00100011	35	#	01000011	67	C	01100011	99	c
00000100	4	End of transmit	00100100	36	$	01000100	68	D	01100100	100	d
00000101	5	Enquiry	00100101	37	%	01000101	69	E	01100101	101	e
00000110	6	Acknowledge	00100110	38	&	01000110	70	F	01100110	102	f
00000111	7	Audible bell	00100111	39	'	01000111	71	G	01100111	103	g
00001000	8	Backspace	00101000	40	(	01001000	72	H	01101000	104	h
00001001	9	Horizontal tab	00101001	41	)	01001001	73	I	01101001	105	i
00001010	10	Line feed	00101010	42	*	01001010	74	J	01101010	106	j
00001011	11	Vertical tab	00101011	43	+	01001011	75	K	01101011	107	k
00001100	12	Form Feed	00101100	44	,	01001100	76	L	01101100	108	l
00001101	13	Carriage return	00101101	45	-	01001101	77	M	01101101	109	m
00001110	14	Shift out	00101110	46	.	01001110	78	N	01101110	110	n
00001111	15	Shift in	00101111	47	/	01001111	79	O	01101111	111	o
00010000	16	Data link escape	00110000	48	0	01010000	80	P	01110000	112	p
00010001	17	Device control 1	00110001	49	1	01010001	81	Q	01110001	113	q
00010010	18	Device control 2	00110010	50	2	01010010	82	R	01110010	114	r
00010011	19	Device control 3	00110011	51	3	01010011	83	S	01110011	115	s
00010100	20	Device control 4	00110100	52	4	01010100	84	T	01110100	116	t
00010101	21	Neg. acknowledg	00110101	53	5	01010101	85	U	01110101	117	u
00010110	22	Synchronous idle	00110110	54	6	01010110	86	V	01110110	118	v
00010111	23	End trans. block	00110111	55	7	01010111	87	W	01110111	119	w
00011000	24	Cancel	00111000	56	8	01011000	88	X	01111000	120	x
00011001	25	End of medium	00111001	57	9	01011001	89	Y	01111001	121	y
00011010	26	Substitution	00111010	58	:	01011010	90	Z	01111010	122	z
00011011	27	Escape	00111011	59	;	01011011	91	[	01111011	123	{
00011100	28	File separator	00111100	60	<	01011100	92	\	01111100	124	|
00011101	29	Group separator	00111101	61	=	01011101	93	]	01111101	125	}
00011110	30	Record Separator	00111110	62	>	01011110	94	^	01111110	126	~
00011111	31	Unit separator	00111111	63	?	01011111	95	_	01111111	127	Del

ENCODING

- UTF-16
- Y también existe UTF-32



ENCODING

- ISO-8859-X

Binario	Oct	Dec	Hex	1	2	3	4	5	6	7	8	9	10	11	13	14	15	16
1010 0000	240	160	A0	Espacio duro (NBSP)														
1010 0001	241	161	A1	¡	Ą	Ħ	Ą	Ë		‘		¡	Ą	ก	”	Ḃ	¡	Ą
1010 0010	242	162	A2	¢	˘		ĸ	Ъ		’	¢	¢	Ē	ข	¢	ḃ	¢	ą
1010 0011	243	163	A3	£	Ł	£	Ŗ	Ґ		£			Ģ	ฃ	£			Ł
1010 0100	244	164	A4	¤				Є	¤	€	¤		Ī	ค	¤	Ċ	€	
1010 0101	245	165	A5	¥	Ľ		Ĩ	S		₯	¥		Ĩ	ฅ	„	ċ	¥	„
1010 0110	246	166	A6	¦	Ś	Ĥ	Ļ	I		¦			Ķ	ฆ	¦	Ḋ	Š	
1010 0111	247	167	A7	§				Ї		§				ง	§			
1010 1000	250	168	A8	¨				J		¨			Ļ	จ	Ø	Ẁ	š	
1010 1001	251	169	A9	©	Š	İ	Š	Љ		©			Đ	ฉ	©			
1010 1010	252	170	AA	ª	Ş		Ē	Њ		˛	×	ª	Š	ช	Ŗ	Ẃ	ª	Ş
1010 1011	253	171	AB	«	Ť	Ğ	Ģ	Ћ		«			Ŧ	ซ	«	đ	«	
1010 1100	254	172	AC	¬	Ź	Ĵ	Ŧ	Ќ	ˌ	¬			Ž	ฌ	¬	Ẏ	¬	Ź
1010 1101	255	173	AD	Guion ortográfico (SHY)										ญ	SHY			
1010 1110	256	174	AE	®	Ž		Ž	Ў			®		Ū	ฎ	®			ź
1010 1111	257	175	AF	¯	Ż		¯	Ų		—	¯		Ɒ	ฏ	Æ	Ÿ	¯	Ż
1011 0000	260	176	B0	°				A		°				ฐ	°	Ḟ	°	
1011 0001	261	177	B1	±	ą	ħ	ą	Б		±			ą	ฑ	±	ḟ	±	

ENCODING

- ISO 8859-1 (Latin-1), para la zona de Europa occidental.
- ISO 8859-2 (Latin-2), para la zona de Europa occidental y Centroeuropa.
- ISO 8859-3 (Latin-3), para la zona de Europa occidental y Europa del sur.
- ISO 8859-4 (Latin-4), para la zona de Europa occidental y países bálticos (lituano, estonio y lapón).
- ISO 8859-5, para el alfabeto cirílico.
- ISO 8859-6, para el alfabeto árabe.
- ISO 8859-7, para el alfabeto griego.
- ISO 8859-8, para el alfabeto hebreo.
- ISO 8859-9 (Latin-5), para la zona de Europa occidental con los caracteres del alfabeto turco.
- ISO 8859-10 (Latin-6), para la zona de Europa occidental, incluye los caracteres del alfabeto nórdico, lapón y esquimal.
- ISO 8859-11, incorpora caracteres del alfabeto tailandés.
- ISO 8859-13 (Latin-7), incorpora caracteres para los idiomas bálticos y el polaco.
- ISO 8859-14 (Latin-8), incorpora caracteres para los idiomas celtas.
- ISO 8859-15 (Latin-9), añade el símbolo del euro.
- ISO 8859-16, incorpora caracteres para los idiomas polaco, checo, eslovaco, húngaro, albano, rumano, alemán e italiano.

OPERACIONES CON FICHEROS

- Crear nuevo
- Abrir existente
- Cerrar actual
- Leer un fichero
- Escribir en un fichero

OPERACIONES CON REGISTROS

Cuando abrimos un fichero:

- Alta

  Añadir un registro nuevo.

- Baja
  - Lógica: flag, comentario, espacios en blanco.
  - Física: eliminar registro.

- Modificaciones

  Cambiar valor de un registro existente.

- Consultar

  Buscar un determinado registro.

Linux

- File f = new File (“/home/ejercicios/UT1/ejemplo1.txt”);

Windows

- File f = new File (“C:\\Ejercicios\\UT1”, “ejemplo2.txt”);

ESCRITURA DE FICHEROS

FileWriter

- Escribe cada escritura directamente en el fichero (memoria secundaria)
- Versión del constructor para no sobrescribir

BufferedWriter

- Escribe cada escritura en un búfer intermedio
- Cuando el búfer está lleno va a escribir en memoria secundaria
- Al hacer close() o flush() se fuerza la actualización del fichero

PrintWriter

- print(), println(), append()…
- Para textos más cortos
- Más sencillo de usar
- Menos eficiente

LECTURA DE FICHEROS

FileReader

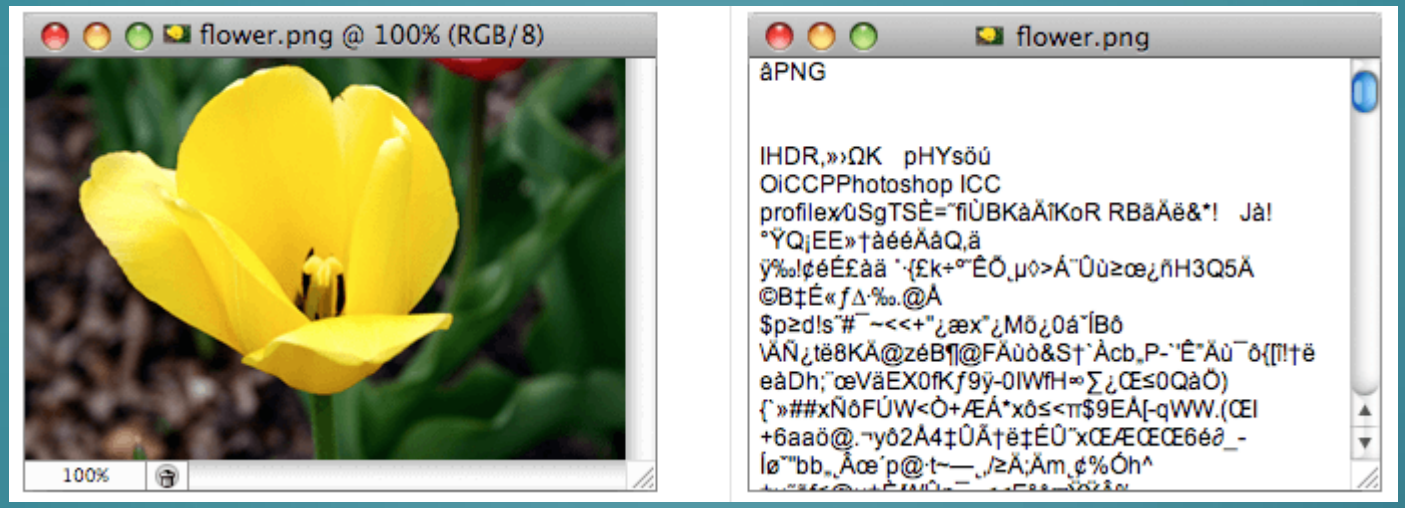
- Lectura carácter a carácter

BufferedReader

- Permite leer líneas

Scanner

- System.in
- nextLine(), nextInt(), nextFloat(), nextBoolean()…



Ficheros binarios

- Flujo de bytes (8 bits)
- InputStream, OutputStream

Ficheros de texto

- Flujo de caracteres (16 bits)
- Reader, Writer

Clases de Flujos de Bytes	Clases de Flujos de Caracteres
InputStream	Reader [InputStreamReader]
OutputStream [OutputStreamWriter]	Writer
FileInputStream	FileReader
FileOutputStream	FileWriter

MÉTODOS IMPORTANTES FILE

Método	Función
`String[] list()`	Devuelve un array de String con los nombres de ficheros y directorios asociados al objeto **File**
`File[] listFiles()`	Devuelve un array de objetos **File** conteniendo los ficheros que estén dentro del directorio representado por el objeto **File**
`String getName()`	Devuelve el nombre del fichero o directorio
`String getPath()`	Devuelve el camino relativo
`String getAbsolutePath()`	Devuelve el camino absoluto del fichero/directorio
`boolean exists()`	Devuelve *true* si el fichero/directorio existe
`boolean canWrite()`	Devuelve *true* si el fichero se puede escribir
`boolean canRead()`	Devuelve *true* si el fichero se puede leer
`boolean isFile()`	Devuelve *true* si el objeto **File** corresponde a un fichero normal
`boolean isDirectory()`	Devuelve *true* si el objeto **File** corresponde a un directorio
`long length()`	Devuelve el tamaño del fichero en bytes
`boolean mkdir()`	Crea un directorio con el nombre indicado en la creación del objeto **File**. Solo se creará si no existe
`boolean renameTo(File nuevonombre);`	Renombra el fichero representado por el objeto **File** asignándole *nuevonombre*
`boolean delete()`	Borra el fichero o directorio asociado al objeto **File**
`boolean createNewFile()`	Crea un nuevo fichero, vacío, asociado a **File** si y solo si no existe un fichero con dicho nombre
`String getParent()`	Devuelve el nombre del directorio padre, o *null* si no existe

ACCESO

Secuencial

- Se recorren todas las posiciones de memoria hasta llegar al destino.
- No es posible hacer inserciones entre datos escritos. Se añade al final.

Aleatorio o directo

- Se accede directamente a la posición deseada.
- Los registros tienen un tamaño conocido.

OPERACIONES FICHEROS SECUENCIALES

- Alta

  Solo permite añadir al final del fichero. Nuevo registro al final del último existente.

- Baja

  Se tendrían que leer todos los registros, menos el que se desea dar de baja, copiarlos en otro fichero auxiliar. Posteriormente eliminar fichero original y renombrar el nuevo.

- Modificaciones

  Igual que la baja, pero realizando las modificaciones deseadas.

- Consultar

  Para consultar un determinado registro n, se tienen que leer los n-1 anteriores.

RANDOMACCESSFILE

new RandomAccessFile (String nombre, String modoAcceso)
new RandomAccessFile (File fichero, String modoAcceso)

modoAcceso = “r” → Lectura
modoAcceso = “rw” → Lectura y escritura

.read()
.write()
.getFilePointer()
.seek(long posición)
.length()
.skipBytes(int desplazamiento)

Para introducir textos de una longitud fija…
Mirar StringBuffer y su método setLength()

FICHEROS XML

GML → SGML → XML

GML fue inventado por IBM en los 70

General **M**arkup **L**anguage

FICHEROS XML

GML → **SGML** → XML

GML gustó a ISO y se estandariza en 1986 (**SGML**)

Standard **G**eneral **M**arkup **L**anguage

FICHEROS XML

GML → SGML → **XML**

En 1999 W3C publica **XML**

e**X**tensible **M**arkup **L**anguage

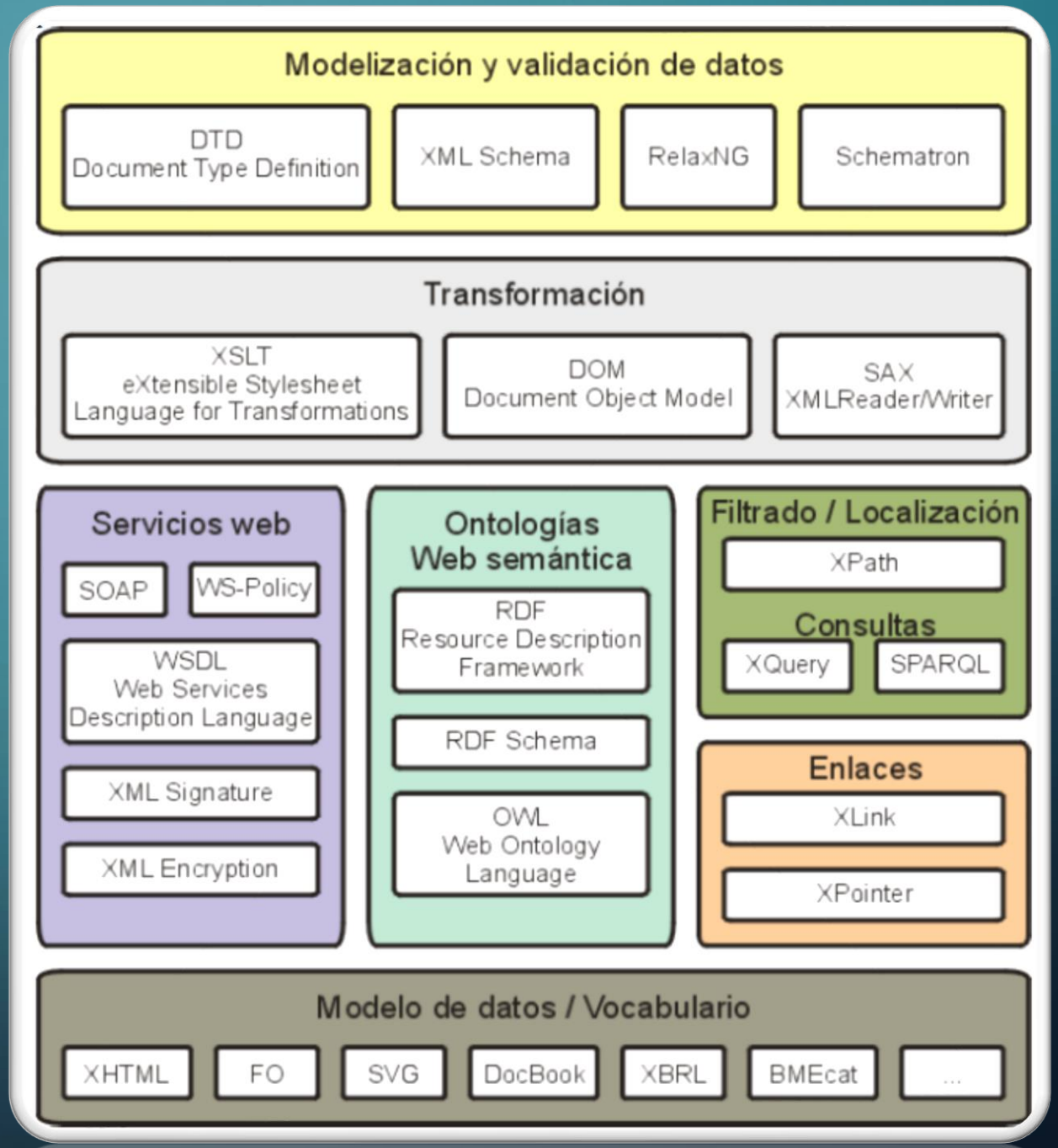
FICHEROS XML

¿Es XML un lenguaje de marcas?

- ✓ Realmente es un metalenguaje.
- ✓ Permite crear otros lenguajes
- ✓ Existen un conjunto de tecnologías que usan XML
- ✓ Por lo tanto, más bien es una familia de tecnologías…

FICHEROS XML

Familia de tecnologías



FICHEROS XML

TEXTO

SENCILLO

POPULAR

XML es:

- ✓ Extensible
  - Puedes definir nuevas etiquetas
- ✓ Estructurado
  - A cualquier nivel de complejidad
- ✓ Validable
  - Puedes validar o no un XML en base a su esquema de validación
- ✓ Independiente
  - Del fabricante y la plataforma. Múltiples formatos

FICHEROS XML

Estructura XML:

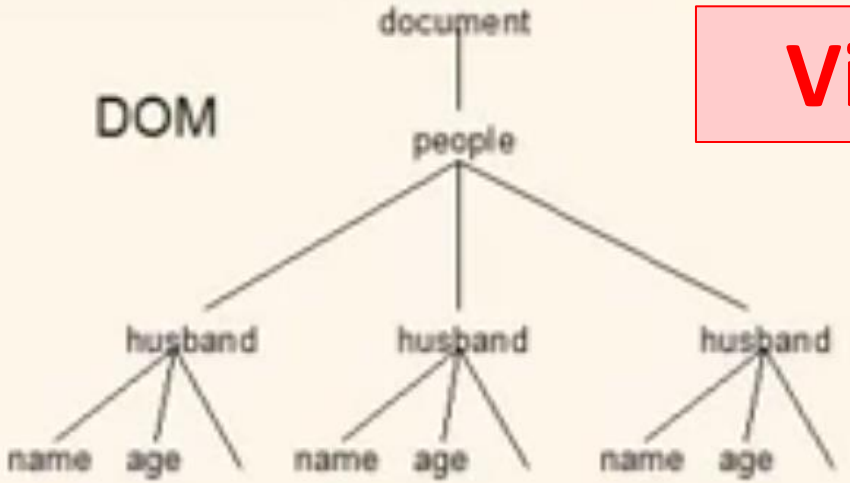
- ✓ Las etiquetas delimitan el texto que desea marcar encerrándolo entre una etiqueta de **inicio** de marcado < > y una etiqueta de **cierre** de marcado </ >
- ✓ Documento XML = Prólogo + Ejemplar
- ✓ **Prólogo** = Declaración XML + Declaración DTD. Opcional, pero recomendable.
- ✓ Elementos ≃ Marcas ≃ Etiquetas
- ✓ Atributo = "valor" → Actúa como modificador o adjetivo. Sencillo y no subdivisible.
- ✓ Comentarios → <!-- Esto es comentario, cualquier carácter (menos '--') -->
- ✓ Secciones de datos → <![CDATA[sección de datos]]>

FICHEROS XML

SAX Or DOM?

```
<people>
  <husband employed="yes">
     <name>Mark</name>
     <age>45</age>
       <wife>
         <wname>Janet</wname>
         <age>29</age>
       </wife>
  </husband>
  <husband employed="no">
     <name>Matt</name>
     <age>43</age>
       <wife>
         <wname>Annie</wname>
         <age>41</age>
       </wife>
  </husband>
<husband employed="no">
     <name>Bob</name>
     <age>46</age>
       <wife>
         <wname>Agnes</wname>
         <age>41</age>
       </wife>
  </husband>
```

DOM



SAX

```
start document
start element: people
start element: husband
start element: name
characters: Mark
end element:name
start element: age
.
.
.
```

Visión global

Más eficiente

FICHEROS XML

DOM

org.w3c.dom

- ✓ **Document**
  Equivale a un ejemplar de XML
- ✓ **Element**
  Propiedades y métodos para los elementos
- ✓ **Node**
  Representa cualquier nodo del documento
- ✓ **NodeList**
  Lista con los nodos-hijo de un nodo
- ✓ **Attr**
  Acceder a los atributos de un nodo
- ✓ **Text**
  Caracteres de un elemento
- ✓ **CharacterData**
  Métodos para manipular caracteres
- ✓ **DocumentType**
  Información del <!DOCTYPE>

FICHEROS XML

DOM

javax.xml.parsers

```
DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();
try {
        DocumentBuilder db = dbf. DocumentBuilder();
…

        DOMImplementation imp = db.getDOMImplementation();
        Document document = imp.createDocument(null, “raíz”, null);
…
```

DOM

FICHEROS XML

```
document.setXMLVersion("1.0");

Element element = document.createElement("elemento");

Text text = document.createTextNode("Texto de ejemplo");

element.appendChild(text);

document.getDocumentElement().appendChild(element);
```

FICHEROS XML

DOM no define mecanismos para generar un XML

```
Source source = new DOMSource(document);

Result result = new StreamResult(new File("ejemplo.xml"));

Transformer t = TransformerFactory.newInstance().newTransformer();

t.transform(source, result);
```

FICHEROS XML

¿Y para leer un XML?

```
document = builder.parse(new File("ejemplo.xml"));

NodeList ejemplo = document.getElementbytagName("elemento");
```

En el modelo DOM para XML, todo considera un NODO:

- Todo el documento en su conjunto es un nodo-documento.
- Cada elemento XML es un nodo-elemento.
- El texto de los elementos XML son nodos-texto.
- Cada atributo es un nodo-atributo.
- Los comentarios son nodos-comentario.

Para manipular los nodos en Java se dispone de la clase Node

FICHEROS XML

¿Y para leer un XML?

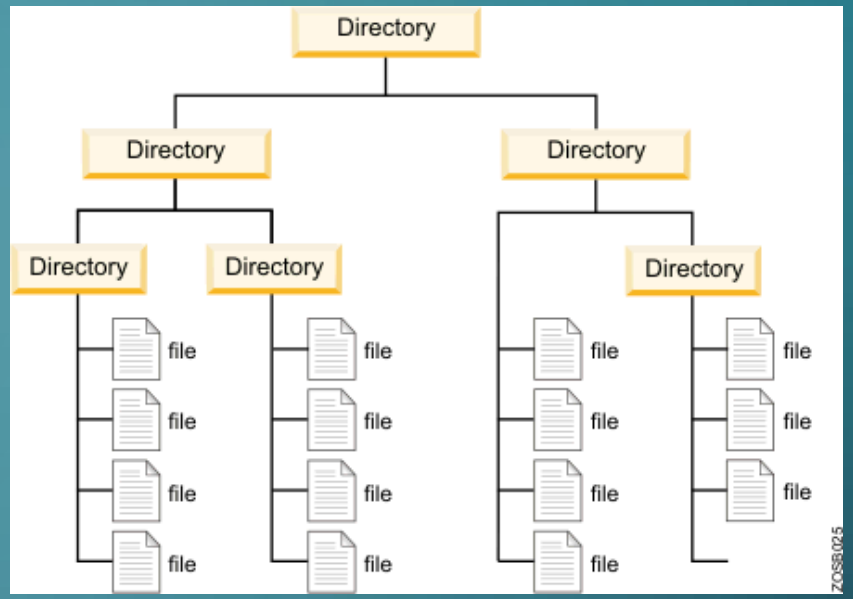
- ✓ Node getFirstChild()
- ✓ Node getNextSibling()
- ✓ Node getParentNode()
- ✓ NodeList getChildNodes()
- ✓ int getLength()
- ✓ String getNodeValue()
- ✓ boolean hasAttributes()
- ✓ void setAttribute(String name, String value)
- ✓ Node appendChild(Node hijoNuevo)
- ✓ Node insertBefore(Node hijoNuevo, Node hijoReferencia)
- ✓ Node removeChild(Node hijoViejo)
- ✓ Node replaceChild(Node hijoNuevo, Node hijoViejo)

https://docs.oracle.com/javase/7/docs/api/org/w3c/dom/package-summary.html

FICHEROS VS BBDD

Hasta la aparición de las bases de datos, las empresas utilizaban ficheros para digitalizar la información. Útiles para poco volumen de datos sin dependencias entre sí.

Sistemas de información **orientados al proceso**.



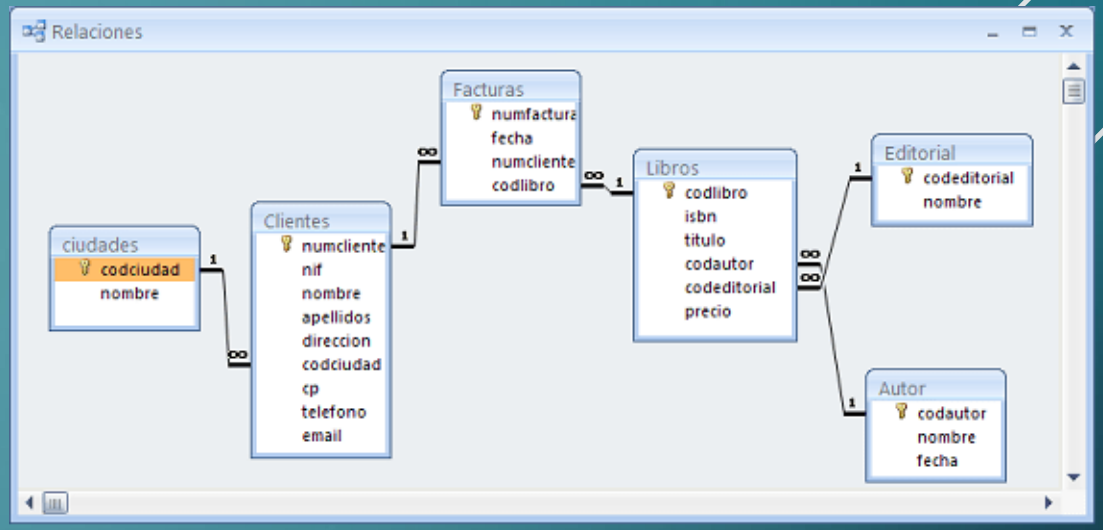
- Redundancia
- Inconsistencia
- Organización
- Escalabilidad
- Rendimiento

FICHEROS VS BBDD

Las bases de datos aparecen en los años 70 ante la necesidad de organizar la cantidad creciente de información por parte de las empresas. Es un sistema mucho más eficaz que el sistema de ficheros y está ampliamente consolidado en la actualidad.

Acceso a los datos mediante SGBD con lenguaje estandarizado SQL.

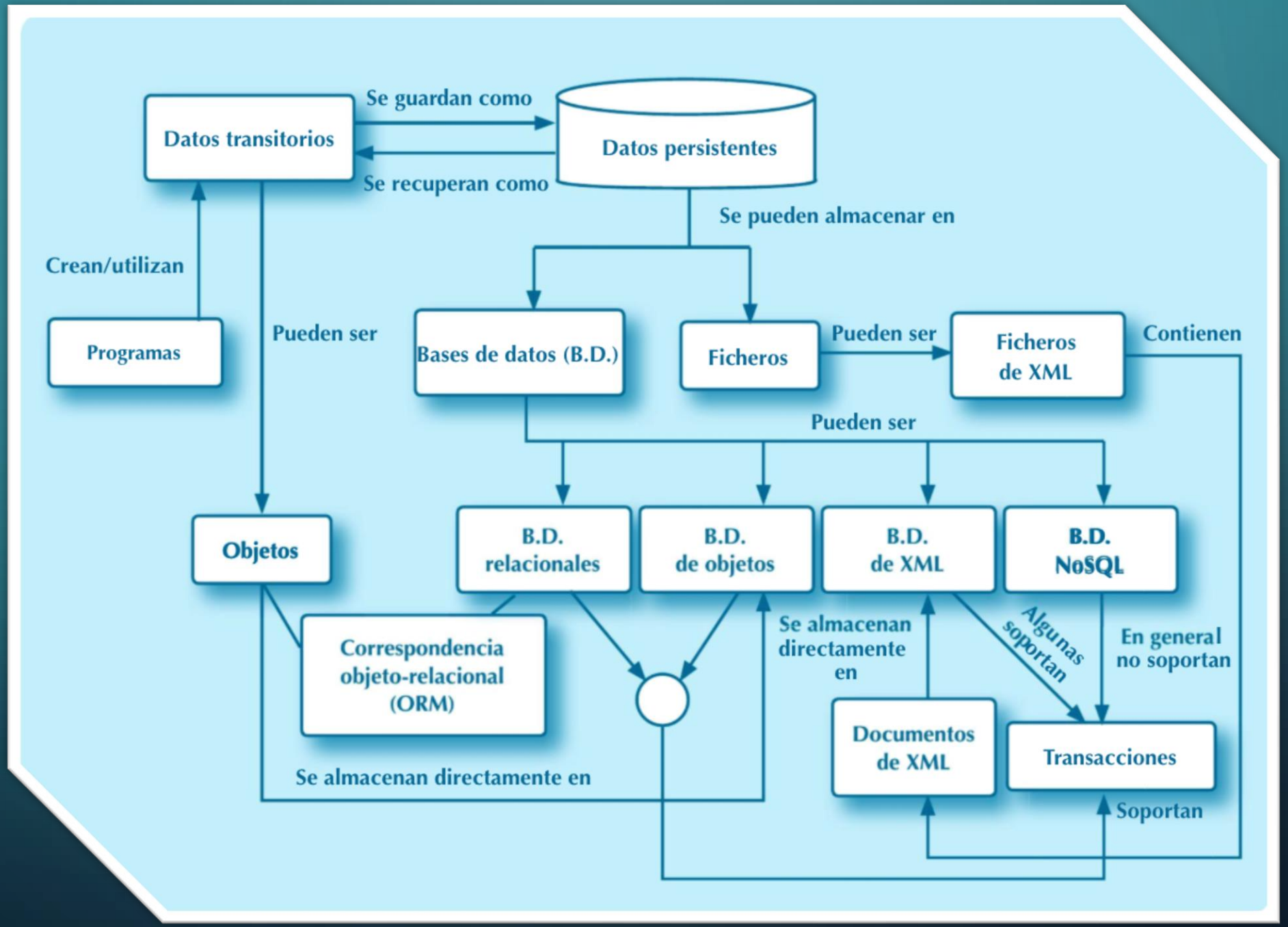
Sistemas de información **orientados los datos.**



- ✓ Consistencia
- ✓ Organización
- ✓ Escalabilidad
- ✓ Integración
- ✓ Seguridad
- ✓ Concurrencia

BASES DE DATOS

SGBD


Datos transitorios
Se guardan como
Datos persistentes
Se recuperan como
Se pueden almacenar en
Crean/utilizan
Programas
Pueden ser
Bases de datos (B.D.)
Ficheros
Pueden ser
Ficheros de XML
Contienen
Pueden ser
Objetos
B.D. relacionales
B.D. de objetos
B.D. de XML
B.D. NoSQL
Correspondencia objeto-relacional (ORM)
Se almacenan directamente en
Algunas soportan
En general no soportan
Documentos de XML
Transacciones
Se almacenan directamente en
Soportan