REPRESENTACIÓN Y COMUNICACIÓN DE LA INFORMACIÓN

1. REPRESENTACIÓN DE LA INFORMACIÓN

1.1. CONCEPTO DE INFORMACIÓN.

La información es un fenómeno que proporciona significado o sentido a las cosas, e indica mediante códigos y conjunto de datos, los modelos del pensamiento humano. La información por tanto, procesa y genera el conocimiento humano. Aunque muchos seres vivos se comunican transmitiendo información para su supervivencia, la superioridad de los seres humanos radica en su capacidad de generar y perfeccionar, tanto códigos como símbolos con significados que conformaron lenguajes comunes útiles para la convivencia en sociedad, a partir del establecimiento de sistemas de señales y lenguajes para la comunicación.

Los datos se perciben mediante los sentidos, estos los integran y generan la información necesaria para el conocimiento quien permite tomar decisiones para realizar las acciones cotidianas que aseguran la existencia social. El ser humano ha logrado simbolizar los datos en forma representativa, para posibilitar el conocimiento de algo concreto y creó las formas de almacenar y utilizar el conocimiento representado.

Existe una relación indisoluble entre la información, el conocimiento, el pensamiento y el lenguaje por lo que comprender la información es la forma de liberar el conocimiento que genera el pensamiento humano. Dicha liberación se produce mediante el lenguaje (oral, escrito, gesticular, etc.), un sistema de señales y símbolos que se comunican de alguna manera.

1.2. TEORÍA DE LA INFORMACIÓN.

La teoría de la información es la rama de la teoría matemática de la probabilidad y estadística que se relaciona con los conceptos de información y entropía de la información, sistemas de comunicación, transmisión de datos, así como la teoría de la distorsión de la transferencia, criptografía, relaciones señal-ruido, compresión de datos y temas relacionados.

Claude E. Shannon es conocido como el padre de la teoría de la información. Su teoría considera la transmisión de la información como un fenómeno estadístico y ofrece a los ingenieros en comunicaciones una forma de determinar la capacidad de un canal de comunicación en términos comunes de medida llamados bits. La parte de la teoría, que hace referencia a la transmisión, no está relacionada con el contenido de información o el mensaje en sí mismo, aún cuando el lado complementario de la teoría de la información se preocupa por el contenido a través de la compresión con pérdida de los mensajes sujetos a un criterio de fidelidad. Estas dos ramas de la teoría de la información están unidas y justificadas mutuamente por los teoremas de transmisión de información, o los teoremas de separación de canal de origen que justifican el uso de bits como el formato universal de información en diversos contextos.

La Teoría de la Información y de la Codificación es uno de los pilares teóricos sobre los que se construyen las computadoras modernas y los sistemas de transmisión de la información.

1.3. LA INFORMACIÓN EN LOS COMPUTADORES

Conviene tener en cuenta que una computadora no es más que un sistema de procesamiento de la información, y que antes de poder iniciar ese procesamiento, hay que representar dicha información de alguna manera para que la máquina sea capaz de reconocerla y utilizarla.

En el momento actual los elementos con que se construyen las computadoras son dispositivos electrónicos capaces de representar una de dos posibilidades, denominadas de modo arbitrario 1 y 0 (lo que de manera coloquial se denomina tecnología digital), y en consecuencia, es necesaria la existencia de una etapa de traducción entre la representación de la información usada en el mundo real, fácil de entender y manipular por los seres humanos, compuesta por estímulos sonoros, luminosos, olfativos, o de cualquier otro tipo, y la representación adecuada para la computadora, que como ya se ha dicho, consiste únicamente en secuencias de 1 y 0.

Cuando la computadora finaliza el tratamiento de los datos, en la mayoría de las ocasiones será necesario realizar la traducción inversa: desde la secuencia de 1 y 0 usada por la máquina hasta un conjunto de símbolos, colores, sonidos, o estímulos a los que el ser humano (o cualquier otro ente, como puede ser otra máquina) pueda ser sensible.

Cuando se habla de computadoras como máquinas capaces de tratar automáticamente la información (realmente lo que manipulan son representaciones simbólicas de dicha información), se está haciendo referencia a multitud de posibles procesamientos, entre los que se pueden indicar:

- Adquisición automática de información, lo que implica su traducción o codificación a fin de lograr una representación de dicha información en el "interior" de la computadora.
- Almacenamiento de la información, ya que quizás la información introducida y/o procesada no es requerida en el mismo instante en el que ya se encuentra disponible después de su tratamiento.
- Transferencia automática de la información, permitiendo así la comunicación de dicha información entre distintos sistemas de computación o seres humanos.
- Representación de la información transferida y/o procesada en un formato inteligible o útil para el ser humano u otra máquina.

1.4. CODIFICACIÓN Y REPRESENTACIÓN DE LA INFORMACIÓN.

Como hemos mencionado anteriormente, los circuitos digitales son capaces de almacenar únicamente dos estados: 0 y 1. Sin embargo, una computadora es capaz manejar y almacenar todo tipo de información: números enteros, reales, imágenes, sonidos, texto, etc. Por tanto, toda esta información debe reducirse a su representación mediante 0 y 1.

En algunos casos (como los números) su representación binaria puede resultar relativamente sencilla, aunque para la mayoría de los casos será necesario realizar algún tipo de codificación que permita su representación en binario.

Como ejemplo, veremos cómo se representan algunos de los tipos más comunes de información.

1.4.1 REPRESENTACIÓN DE NÚMEROS

El sistema decimal común es un sistema de numeración posicional que emplea 10 símbolos y donde la base es 10. En un sistema de numeración posicional, un número se representa por una sucesión dedígitos, en los que a cada dígito se le asocia un peso en función de su posición. Si la base de un sistema de numeración es *b*, el peso que se asocia al dígito i es bi.

Por ejemplo:

$$\mathbf{1458 = 1 \cdot 10^3 + 4 \cdot 10^2 + 5 \cdot 10^1 + 8 \cdot 10^0}$$

Los sistemas digitales pueden representar de manera natural números en base 2, usando los símbolos {0,1}. Por ejemplo el número binario 1010 se puede expresar como:

$$\mathbf{1010 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0,}$$

que equivale al número decimal 10.

Podemos aplicar la misma idea a un número expresado en cualquier base: usando una base *b*, una magnitud *x* de *n* cifras se puede representar como:

$$\mathbf{x = x_{n-1} b^{n-1} + \ldots + x_{n-1} b^{n-1}}$$

Para representar un número expresado en base 10 en cualquier otra base, basta con dividir sucesivamente dicho número por la base, extrayendo los restos. Por ejemplo:

123 $_{(10}$ = 234 $_{(7}$

123 | 7
53 17 | 7
4 3 2

26 $_{(10}$ = 11010 $_{(2}$

26 | 2
0 13 | 2
1 6 | 2
0 3 | 2
1 **1**

1.4.1.1. NÚMEROS ENTEROS POSITIVOS.

- Codificación en binario: es la más usual. El número entero se codifica mediante su representación en base 2.

4310 = 1000011010110

- Codificación BCD: el número se representa en base 10 y cada dígito se codifica en binario usando 4 bits. Esta representación tiene el grave problema de usar más bits de los necesarios.

4310 = 0100001100010000

1.4.1.2. NÚMEROS ENTEROS CON SIGNO.

- Codificación signo-magnitud: El bit más significativo indica el signo (0 positivo y 1 negativo), mientras que los restantes bits indican el valor absoluto en binario. En este formato es preciso conocer con cuantos bits se representa el número. Si utilizamos n bits podremos representar números dentro del rango

  $-(2^{n-1}-1)$ y $(2^{n-1}-1)$.

43 = 00101011

-43 = 10101011

Un posible problema asociado a esta representación es que existe una doble representación para el cero: 000…0 y 100…0. Además, las operaciones aritméticas resultan más complejas que con otras representaciones.

- Complemento a 2: Si el número es positivo se representa por su expresión en binario, y si es negativo se representa por la expresión en binario de 2^n menos su valor absoluto, donde n es el número de bits. El número ha de estar comprendido entre -2^{n-1} y $2^{n-1}-1$.

43 = 00101011

-43 = (256-43) = 11010101

Este sistema de representación es el empleado en todos los procesadores actuales.

- Complemento a 1: Si el número es positivo se representa por su expresión en binario, y si es negativo se representa por la expresión en binario de 2n-1 menos su valor absoluto. El número ha de estar comprendido entre -2n-1+1 y 2n-1-1.

43 = 00101011

-43 = (255-43) = 11010100

El complemento a 1 de un número negativo se puede obtener complementando bit a bit el valor binario de su valor absoluto.

Dado que el complemento a 2 y el complemento a 1 de un número negativo difieren sólo en una unidad, una forma simple de calcular el complemento a 2 de un número negativo es calcular su complemento a 1 y sumarle 1.

43 - 0 0 1 0 1 0 1 1
-43 - 1 1 0 1 0 1 0 0 - Complemento a 1
———————————
-43 - 1 1 0 1 0 1 0 1 - Complemento a 2

La suma y resta de números positivos y negativos en complemento a 2 se puede realizar como si fuesen números positivos en binario. El resultado es correcto si está comprendido entre los valores expresables en este código con el número de dígitos empleado. En caso contrario se dice que existe 'overflow' y el resultado es incorrecto.

1.4.1.3. NÚMEROS REALES.

En un sistema de numeración posicional los números situados a la derecha del punto decimal corresponden a pesos de magnitud exponencial negativa:

$$\mathbf{18.374 = 1\cdot10^{1}+8\cdot10^{0}+3\cdot10^{-1}+7\cdot10^{-2}+4\cdot10^{-3}}$$

Utilizando el mismo método se pueden representar números reales en binario:

43.140625 = 00101011.00100100

$$43 = 00101011;\ 0.140625 = \mathbf{0}\cdot2^{-1}+\mathbf{0}\cdot2^{-2}+\mathbf{1}\cdot2^{-3}+0\cdot2^{-4}+0\cdot2^{-5}+\mathbf{1}\cdot2^{-6}+\mathbf{0}\cdot2^{-7}+\mathbf{0}\cdot2^{-8}$$

Para calcular la parte real en binario se multiplica por 2 la parte real en decimal sucesivamente hasta que ésta tome el valor cero. El dígito correspondiente será 0 o 1 dependiendo del valor de la parte entera en cada paso. Veamos un ejemplo:

0,140625	x 2 =	0,281250	0,0
0,281250	x 2 =	0,5625	0,00

0,5625	x 2 =	1,125	0,001
0,125	x 2 =	0,25	0,0010
0,25	x 2 =	0,50	0,00100
0,50	x 2 =	1,00	0,001001
0,00	x 2 =	0,00	0,0010010

Fundamentalmente, existen dos formas de codificar los números reales en un ordenador:

- Representación en punto fijo: se utiliza una cantidad fina de bits para representar la parte fraccionaria de la magnitud. El principal problema que presenta esta representación es su falta de flexibilidad: no permite representar números muy grandes o números muy pequeños.

- Representación en punto flotante: Para superar las limitaciones de la representación en punto fijo de números muy grandes o muy pequeños se suele utilizar la notación científica, en la que el número se divide en mantisa y exponente:

$$6.023 \cdot 10^{23} \rightarrow \text{Mantisa} = 6.023 \text{ Exponente} = 23$$

Este método se puede utilizar también para la representación de números binarios. Un número real se representa entonces mediante cuatro datos: signo del número, mantisa, signo del exponente y exponente.

S_m	Mantisa	S_e	Exponente

Cada procesador fija la longitud en bits de los diferentes campos, así como el código que se elige para la representación, y la base de la exponencial.

Afortunadamente, existe el estándar de IEEE 754-1985 que normaliza la representación de números en punto flotante. En el estándar se especifican dos niveles de precisión: precisión simple (32 bits) y precisión doble (64 bits). En esta tabla se muestra la asignación de bits para cada una de las partes que se representan.

	Signo	**Exponente**	**mantisa**	**Sesgo (exp.)**
Precisión simple	1 [31]	8[30-23]	23[22-00]	127
Precisión doble	1 [63]	11[62-52]	52[51-00]	1023

El **bit de signo** es tan simple como cabe esperar: si tiene el valor '0' el número será positivo y si vale '1' será negativo.

La parte del **exponente** debe representar tanto los exponentes negativos como los positivos. Para conseguir esto, se utiliza un sesgo que se resta al valor almacenado para obtener el valor real del exponente. Para precisión simple este valor es de 127. Así pues, un exponente de cero significa que 127 es el valor que hay almacenado. Por otro lado, un valor almacenado de 200 implica un exponente de 200-127, es decir 73. Los valores -127 (todo 0) y 128 (todo 1) están reservados para poder representar situaciones especiales, como *infinito*, valores *nan* (not a number), etc. que se pueden obtener en determinadas operaciones.

La **mantisa** se compone de un bit inicial *implícito*, seguido por n bits que representan la magnitud. Para establecer el valor del bit implícito debemos considerar que considerar que cualquier número se puede expresar en notación científica de muchas maneras. Por ejemplo, el número 5 se puede expresar como: 5.00×10^0 ó 0.05×10^2 ó 5000.00×10^{-3}. Con la idea de maximizar la cantidad de números representables, las magnitudes en punto-flotante se almacenan en forma normalizada: básicamente, se coloca el punto decimal detrás del primer dígito distinto de cero. De esta manera, 5 se expresa como 5.0×10^0.

En base 2 el único valor distinto de cero es 1, con lo cual podemos suponer un bit inicial que siempre vale uno y por tanto no necesitamos representarlo explícitamente. Así pues, tendremos un bit más de resolución en la mantisa, es decir, se asume que la mantisa tiene el formato 1.*f*, donde *f* es la parte fraccionaria alamacenada.

1.4.2. CODIFICACIÓN DE TEXTO.

La representación de caracteres alfanuméricos en un computador es muy sencilla, y se realiza generalmente mediante una tabla de códigos, en la que se asigna a cada carácter un código binario de 7 u 8 bits. Además de los símbolos gráficos se incluyen símbolos de control como *nueva línea, fin de fichero, nueva página, escape*…

El código más utilizado es el código ASCII, pensado para el inglés y utiliza 7 bits para codificar cada carácter. En total son 128 símbolos:

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Cha
0	00	Null	32	20	Space	64	40	@	96	60	`
1	01	Start of heading	33	21	!	65	41	A	97	61	a
2	02	Start of text	34	22	"	66	42	B	98	62	b
3	03	End of text	35	23	#	67	43	C	99	63	c
4	04	End of transmit	36	24	$	68	44	D	100	64	d
5	05	Enquiry	37	25	%	69	45	E	101	65	e
6	06	Acknowledge	38	26	&	70	46	F	102	66	f
7	07	Audible bell	39	27	'	71	47	G	103	67	g
8	08	Backspace	40	28	(	72	48	H	104	68	h
9	09	Horizontal tab	41	29	)	73	49	I	105	69	i
10	0A	Line feed	42	2A	*	74	4A	J	106	6A	j
11	0B	Vertical tab	43	2B	+	75	4B	K	107	6B	k
12	0C	Form feed	44	2C	,	76	4C	L	108	6C	l
13	0D	Carriage return	45	2D	-	77	4D	M	109	6D	m
14	0E	Shift out	46	2E	.	78	4E	N	110	6E	n
15	0F	Shift in	47	2F	/	79	4F	O	111	6F	o
16	10	Data link escape	48	30	0	80	50	P	112	70	p
17	11	Device control 1	49	31	1	81	51	Q	113	71	q
18	12	Device control 2	50	32	2	82	52	R	114	72	r
19	13	Device control 3	51	33	3	83	53	S	115	73	s
20	14	Device control 4	52	34	4	84	54	T	116	74	t
21	15	Neg. acknowledge	53	35	5	85	55	U	117	75	u
22	16	Synchronous idle	54	36	6	86	56	V	118	76	v
23	17	End trans. block	55	37	7	87	57	W	119	77	w
24	18	Cancel	56	38	8	88	58	X	120	78	x
25	19	End of medium	57	39	9	89	59	Y	121	79	y
26	1A	Substitution	58	3A	:	90	5A	Z	122	7A	z
27	1B	Escape	59	3B	;	91	5B	[	123	7B	{
28	1C	File separator	60	3C	<	92	5C	\	124	7C	|
29	1D	Group separator	61	3D	=	93	5D	]	125	7D	}
30	1E	Record separator	62	3E	>	94	5E	^	126	7E	~
31	1F	Unit separator	63	3F	?	95	5F	_	127	7F	□

También podemos encontrar otras codificaciones, entre las cuales podemos destacar:

- ISO-8859-1 (Latin 1): Es una extensión del ASCII, usa 8 bits (256 símbolos) e incluye las lenguas de Europa occidental.
- ISO-8859-15 (Latin 9): Modificación de Latin 1, para incorporar el símbolo del Euro, además de otras modificaciones.

- ISO-8859-2 a ISO-8859-14: extensiones del ASCII para diferentes lenguas: cirílico, árabe, griego, hebreo, etc.
- ISO-10646 (Unicode): código 16 bits que engloba a todas las codificaciones e idiomas conocidos.

1.4.3. CODIFICACIÓN DE IMÁGENES.

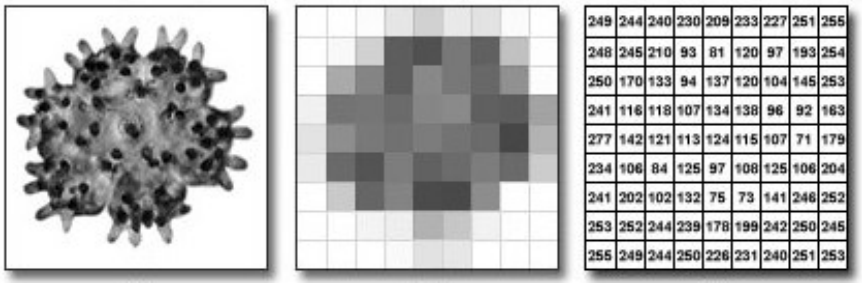
Una imagen natural capturada con una cámara, un telescopio, un microscopio o cualquier otro tipo de instrumento óptico presenta una variación de sombras y tonos continua. Las imágenes de este tipo se llaman **imágenes analógicas**.

Para que una imagen analógica, en blanco y negro, en escala de grises (las llamadas comúnmente, imágenes en blanco y negro), o a color, pueda ser "manipulada" usando un ordenador, primero deben convertirse a un formato adecuado. Este formato es la **imagen digital** correspondiente.

La transformación de una imagen analógica a otra discreta se llama digitalización y es el primer paso en cualquier aplicación de procesamiento de imágenes digitales.
El proceso de digitalización consta de dos partes: **muestreo y cuantificación.**
Un muestreo consiste en una subdivisión de la imagen analógica en porciones cuadradas.

Estos polígonos representan sensores sensibles a la intensidad de luz. La salida de estos sensores es un valor (amplitud) dentro de una escala (color). La salida pueden ser, o bien un único valor (escala de grises) o bien un vector con tres valores por polígono (RGB) que se corresponden con la intensidad de color rojo (R), verde (G) y azul (B). La escala de colores también tiene un rango discreto (por ejemplo, de 8-bits = 256 valores). Las imágenes en escala de grises con sólo 2 colores: blanco y negro (0 y 1, respectivamente), se llaman imágenes binarias.
A este proceso de discretización del color se le llama cuantificación. Un porción de color constante se le llamará píxel.

La siguiente figura representa el proceso de digitalización de una imagen analógica en escala de grises, utilizando una malla de 9x9:



249	244	240	230	209	233	227	251	255
248	245	210	93	81	120	97	193	254
250	170	133	94	137	120	104	145	253
241	116	118	107	134	138	96	92	163
277	142	121	113	124	115	107	71	179
234	106	84	125	97	108	125	106	204
241	202	102	132	75	73	141	246	252
253	252	244	239	178	199	242	250	245
255	249	244	250	226	231	240	251	253

Imagen analógica

Muestreo

Cuantificación de píxeles (imagen digital)

Como podemos apreciar, una imagen digital (sea del tipo que sea) no es más que una matriz numérica.

La resolución (nivel de detalle discernible) de una imagen en escala de grises depende directamente de dos parámetros: del muestreo (tamaño de la malla) y de la cuantificación (número de bits por píxel). Cuanto más se incrementan, más se aproxima la imagen digitalizada a la original, sin embargo el espacio necesario para almacenarla también se incrementará.

Las imágenes digitales a color están gobernadas por los mismos conceptos de muestreo, cuantificación y resolución que las imágenes en escala de grises. Sin embargo, en lugar de un único valor de intensidad que expresa el nivel de gris, los píxeles de las imágenes a color están cuantificados usando tres componentes independientes uno por cada color primario (RGB=rojo, verde y azul). Combinando distintas intensidades de estos tres colores, podemos obtener todos los colores visibles.

1.4.4. CODIFICACIÓN DEL SONIDO.

La grabación digital captura el sonido almacenando los valores de amplitud de una onda a intervalos regulares de tiempo. La amplitud (altura) de una onda de sonido determina su volumen; la frecuencia (medida en Hertzios o Hz) determina su escala (lo grave o aguda que suena).

Las ondas de sonido son continuas en la naturaleza, pero como ya sabemos, un ordenador sólo puede trabajar con información digital (0 / 1). Así que el ordenador almacena la amplitud de una señal a intervalos regulares (muestreo), usando un número determinado de bits (cuantificación). La frecuencia de muestreo indica cuántas muestras del sonido se toman en un segundo. Así una frecuencia de 22 Khz indica 22.000 muestras por segundo.

El número de bits también influye en la calidad de la grabación ya que indica el número niveles de amplitud del sonido: con 8 bits podremos medir 256 niveles en cada muestra y con 16 bits, 65.535. La calidad CD es un estándar que indica que ese sonido está grabado a 16 bits y 44.1 Khz.

2. COMUNICACIÓN DE LA INFORMACIÓN

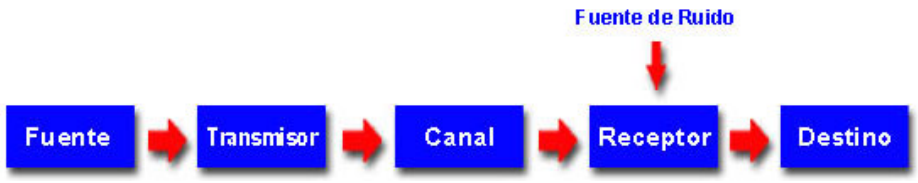
A la hora de entender cómo funciona un sistema informático hay que comprender cómo se representa la información en tal sistema, pero otros procesos que se suelen llevar a cabo sobre la información son almacenarla y transmitirla.

La mayoría de las operaciones que se llevan a cabo en el ordenador requieren del almacenamiento de la información, en algún tipo de medio que tenga un comportamiento binario, ya sea memoria principal o almacenamiento secundario.

También son importantes los aspectos relativos a la comunicación o transmisión de la información.

2.1. ELEMENTOS DE LA COMUNICACIÓN

Un sistema de comunicación es aquel en el que dos entidades intercambian información mediante mecanismos específicos. Este sería el esquema simplificado de un sistema de comunicación:



En la figura podemos distinguir los siguientes elementos fundamentales que intervienen en todo proceso de comunicación:

- Fuente: Componente de naturaleza humana o mecánica que determina el tipo de mensaje que se transmitirá y su grado de complejidad.
- Transmisor: Recurso técnico que transforma el mensaje originado por la fuente de información en señales apropiadas.
- Canal: Medio generalmente físico que transporta las señales en el espacio. Cumple una función simple de mediación y transporte.
- Ruido: Expresión genérica utilizada para referirse a variadas distorsiones originadas en forma externa al proceso de comunicación.
- Receptor: Recurso técnico que transforma las señales recibidas en el mensaje concebido por la fuente de información.
- Destino: Componente terminal del proceso de comunicación, al cual está dirigido el mensaje. Es el elemento decisivo para pronunciarse sobre la fidelidad de la comunicación.

2.2. DETECCIÓN DE ERRORES

Idealmente, el estado físico a la entrada del canal determina una única salida. Sin embargo el canal nunca estará totalmente aislado del exterior, y siempre se introducirá ruido aditivo, que altere de forma aleatoria los datos que viajan por el canal en mayor o menor medida. Debido a ello una misma entrada en el canal puede producir diferentes

salidas, cada una de ellas con una determinada distribución de probabilidad, con lo que entradas diferentes pueden tener la misma salida, lo que las hace indistinguibles.

De esta forma la estimación de la entrada que haga el destino en función de la salida siempre tendrá una probabilidad de error no nula.

Así pues, la comunicación entre dos dispositivos digitales está condicionada por los posibles errores introducidos en el canal, con lo que la información reconstruida por el receptor puede contener errores: uno o más bits han cambiado su valor.

El método para detectar y corregir tales errores es incluir en los bloques de datos transmitidos bits adicionales denominados **bits de redundancia.**

Se han desarrollado diversos códigos para la transmisión de datos que introducen bits de redundancia para detectar e incluso corregir errores en la transmisión.

Algunos de los mecanismos de detección de errores más usuales son los siguientes:

- Bit de paridad: se suele utilizar para detectar errores en transmisión de caracteres. Se trata de añadir a los n bits que forman el carácter original, un bit extra. Este bit se elige de forma que el número total de bits con valor '1' en el carácter sea par (código de paridad par) o impar (código de paridad impar). Este método, aunque resulta satisfactorio en general, sólo es útil si los errores no cambian un número par de bits a la vez, porque un número par de errores no afecta a la paridad de los datos.

- Códigos de bloque: El principio que se utiliza en los códigos de bloque consiste en estructurar los datos en bloques de longitud fija y añadir a cada bloque un cierto número de bits de redundancia. Sólo ciertas combinaciones de bits son aceptables y forman una colección de palabras de código válidas. Cuando los datos se transmiten y llegan al receptor hay dos posibilidades: que la palabra que se recibe sea una palabra de código válido, o bien que no sea, en cuyo caso el receptor puede regenerar el bloque original (código autocorrector) o pedir que se retransmita el bloque (código de autochequeo).

- Códigos de Redundancia Cíclica: Los códigos cíclicos también se llaman CRC (Códigos de Redundancia Cíclica) o códigos polinómicos. Su uso está muy extendido porque pueden implementarse en hardware con mucha facilidad. Estos códigos se basan en el uso de un **polinomio generador** G(X) de grado r, y en el principio de que n bits de datos binarios se pueden considerar como los coeficientes de un polinomio de orden n-1. Por ejemplo, los datos 10111 pueden tratarse como el polinomio $x^4 + x^2 + x^1 + x^0$.

  A estos bits de datos se le añaden r bits de redundancia de forma que el polinomio resultante sea divisible por el polinomio generador, sin generar resto. El receptor verificará si el polinomio recibido es divisible por G(X). Si no lo es, habrá un error en la transmisión.

2.3. COMPRESIÓN DE DATOS.

El término 'compresión de datos' se refiere al proceso de reducción del volumen de datos necesario para poder representar una determinada información. Hay que tener en cuenta que datos e información no son sinónimos, es decir, los datos son el medio a través del cual se conduce la información.

La compresión se aplica con la idea de minimizar dos aspectos:

- El espacio requerido para el almacenamiento en disco.
- El tiempo de transmisión de los datos en un proceso de comunicación.

Existen muchos algoritmos de compresión: LZ77, LZ78, LZW, Huffman, aritméticos, fractales, MPEG, JPEG,… que se utilizan con distintas expectativas: compresión de datos, vídeo y voz, imágenes, compresión en tiempo real, etc. Genéricamente podemos clasificarlos en dos tipos:

- Compresores sin pérdidas, en el sentido de que guarda absolutamente toda la información original (es reversible). Se utilizan para la compresión de datos, en los que no se puede dar pérdida de información. Pueden ser de dos tipos:
  - **Compresores estadísticos**.
  - **Compresores basados en diccionario**.

- Compresores con pérdidas. La compresión hace que se pierda información de la fuente original. Sin embargo, esta pérdida es insignificante en comparación con la ganancia en compresión. Se utiliza sobre todo en imágenes y sonido, donde se puede "engañar" a los sentidos, donde una pérdida de calidad apenas es percibida (pero ocasiona un ratio de compresión mucho mayor).