Raymond Colle

INICIACIÓN A LA INFORMÁTICA Y A LA PROGRAMACIÓN

INCOM CHILE
2018

Título: Iniciación a la informática y a la programación
Autor: Raymond Colle
Ediciones INCOM-Chile, Asociación Chilena de Investigadores en Comunicación, Santiago de Chile, 2018

El autor

Raymond Colle es doctor en Ciencias de la Información y analista de sistemas; profesor jubilado de la Escuela de Periodismo de la Pontificia Universidad Católica de Chile y ex-investigador del Centro de Estudios Mediales de la Universidad Diego Portales (Santiago de Chile).

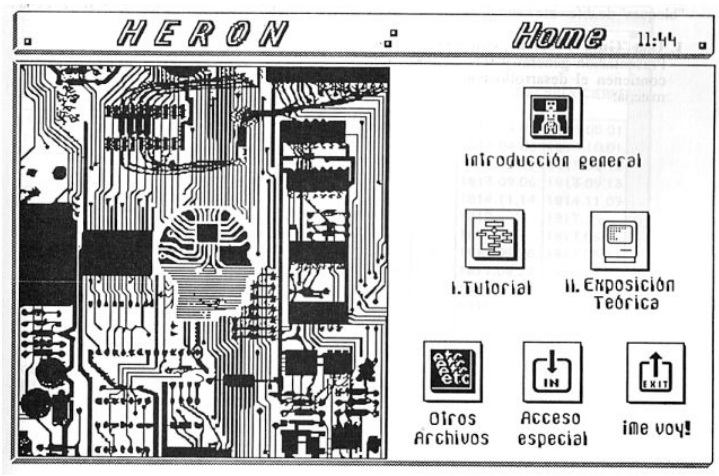
Ha publicado varios libros sobre ciencias de la comunicación, documentación periodística, teoría cognitiva e internet así como numerosos artículos sobre dichos temas y de ética en revistas académicas.

Ha programado en Fortran y Cobol (años '70), Basic y Assembler para Apple II (años '80), Hypertalk y Prolog para Macintosh (años '90), HTML para web, PHP y MySQL (para servidores Linux).

Las imágenes incluidas en el texto provenientes de fuentes diversas se insertaron bajo el principio de *"fair use"*, dado que la presente obra es de tipo académico y no tiene fines comerciales.

Tabla

Introducción



Cuando, en 1989, se instaló en la Escuela de Periodismo de la Pontificia Universidad Católica de Chile su primera sala de computadores, poblada de Macintosh Classic, inicié también la enseñanza de una asignatura titulada "Procesamiento de Información", un tercio de la cual se refería a la estructura y forma de operar de los computadores. Aunque ya enseñaba esta asignatura antes, aprovechando el nuevo sistema de hipertexto desarrollado por Apple (llamado "Hypercard"), inauguré, en 1991, una modalidad de autoinstrucción, los alumnos encontrando en los computadores toda la información teórica, un tutorial que los guiaba (con un contenido lúdico) y un sistema de evaluación de sus avances. Este proyecto tenía por nombre "Herón", que provenía del primer creador de autómatas, de la Grecia clásica. La parte del curso referida a los sistemas de documentación y las bases de datos también se ofreció de modo similar, bajo el nombre de "Dédalo", en alusión al laberinto de Creta y a lo "dedálico" que podía ser recuperar información en esa época.
Cuando, ya en 1996, la universidad hubo lanzado su red basada en la World Wide Web, desarrollé una asignatura optativa llamada "Conceptos Básicos de Computación", que fue la primera en dictarse integralmente por la web.

En 2001, para los alumnos de la carrera de "Comunicación Multimedial" de la Universidad Diego Portales, dicté una asignatura parecida titulada "Iniciación a la Computación", también integralmente en la web de dicha universidad.

Cuando, ya en 1996, la universidad hubo lanzado su red basada en la World Wide Web, desarrollé una asignatura optativa llamada "Conceptos Básicos de Computación", que fue la primera en dictarse integralmente por la web.

En 2001, para los alumnos de la carrera de "Comunicación Multimedial" de la Universidad Diego Portales, dicté una asignatura parecida titulada "Iniciación a la Computación", también integralmente en la web de dicha universidad.

Ahora que se propone en varios países y sistemas educativos empezar a enseñar a programar en la educación media o incluso en primaria, y más forzosamente en la universidad, sobre todo para alumnos de periodismo, me ha parecido conveniente revisar mis archivos de estas asignaturas para escoger los elementos esenciales ("conceptos básicos") y actualizarlos para que los estudiantes cuenten con los antecedentes más importantes del campo de la informática para entender el

"mundo digital" y saber operar en él sin perder de vista los objetivos centrales: facilitar la comunicación entre los seres humanos así como el acceso al conocimiento.

¿Se debe realmente enseñar a programar en todos los niveles?

Steve Jobs, el fundador de Apple, declaró hace años en una entrevista *"Creo que todo el mundo en este país debería aprender a programar"*. Y hace poco, su sucesor al frente de la compañía, Tim Cook, aseguraba que *"programar es más importante que aprender inglés"*. ¿Por qué? Porque enseña a pensar de determinado modo. Por cierto no debe ser la única forma de pensar, pero es importante para entender como funcionan las máquinas que usamos a diario y poder usarlas activamente en múltiples profesiones.

Ya en 1968, el sudafricano Seymour Papert (1928-2016; foto adjunta) había visualizado el rol positivo de la enseñanza del pensamiento lógico en la enseñanza inicial, gracias a sus trabajo junto con el psicólogo francés especialista en desarrollo de la inteligencia, Jean Piaget. Trabajando en Instituto de Tecnología de Massachusetts (MIT). Papert inventó en 1968 el lenguaje de programación Logo, que ayuda al pensamiento lógico-matemático, como herramienta para la enseñanza escolar de la programación de computadores. Los niños podían utilizar Logo mediante un programa de *software* llamado MicroWorlds, moviendo una "tortuga" en la pantalla para hacer dibujos, o bien con un producto robótico de Lego.

Federico Peinado, profesor de la facultad de Informática de la Universidad Complutense de Madrid, dice que *"La programación computacional es una disciplina en la que sí sería interesante que todo el mundo estuviese formado ya que te explica cómo piensa una máquina, cómo se le dan órdenes y cómo se programa para que automatice procesos y nos ayude a hacer tareas."* (Xataka, 13/11/2017).

Pero frente al estudio de lenguajes de programación, cree que sólo corresponde a un público más específico y especializado, y concuerdo con él. No es necesario saber programar en un lenguaje determinado si no se va a trabajar en esta área. Pero sí es sumamente útil y debería formar parte de toda formación culta saber cómo se programa y cómo funcionan los computadores. No hay que olvidar que *"Es muy probable que cuando nuestros alumnos de primaria lleguen a la adultez, los lenguajes de programación en boga ya no sean los mismos que usamos hoy"* como dice Arriel Torres, en La Nación.ar (25/11/2017). ¡Ni tantos años han de pasar!

El informe *"Computing our future - Computer programmingand coding - Priorities, school curricula and initiatives across Europe"* (Computar nuestro futuro – Prioridades, curriculum escolar e iniciativas en Europa), de la red europea Schoolnet, comparte la idea de que no se trata de enseñar a programar, sino de favorecer un nuevo tipo de pensamiento que se caracteriza como la capacidad de codificar información:

> *"La codificación (coding) es cada vez más una competencia clave que tendrá que ser adquirida por todos los jóvenes estudiantes y cada vez más por los trabajadores en una amplia gama de actividades industriales y profesiones. La codificación es parte del razonamiento lógico y representa una de las habilidades clave que forma parte de lo que ahora se llaman «habilidades del siglo 21»."*

Sería una error confundir la programación con la competencia codificadora. Codificar no significa programar. Significa ser capaz de realizar *"la transferencia de acciones e informaciones para que puedan ser*

interpretados por los ordenadores y otros dispositivos de proceso, transporte y almacenamiento de la información" (Miguel Zapata-Ros[1]). Supone el desarrollo de algunas habilidades básicas que siguen a la capacidad natural de clasificación, como es la seriación, el pensamiento lógico y el análisis (descomponer en partes), para luego poder abordar la resolución de problemas mediante la construcción de algoritmos, es decir, definiendo las etapas que pueden llevar a la solución.

¿Qué es realmente lo más esencial, que todos deberían conocer? La lógica que sostiene el sistema, cuyo centro es el manejo de las proposiciones condicionales y cuya mejor representación son los diagramas de flujo correspondientes a los algoritmos. También, por cierto, plantearse claramente el problema y desarmarlo en elementos más pequeños y más fáciles de manejar.

Este es el propósito de este libro: señalar los fundamentos de la computación, tanto a nivel de máquinas como de programación. Apunta a las bases, a los conceptos y a la forma de pensar que se plasma en el diseño de las operaciones.

1 Ph.D. in Computer Engineering, académico de la universidad de Murcia, España.

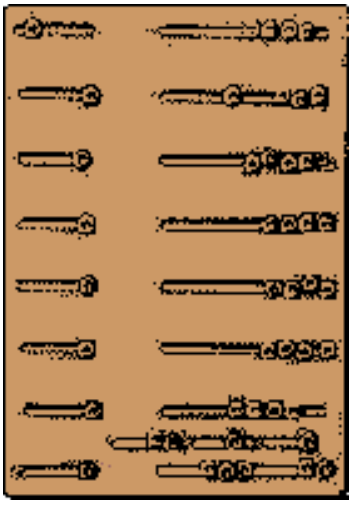
1. Bases de la informática

"Informática" y "computación" son sinónimos aunque, si se quiere hacer una distinción, el primer término apunta más al aspecto teórico mientras el segundo apunta más al manejo de la tecnología. En ambos casos, se trata de una disciplina que estudia métodos, técnicas y procesos, con el fin de almacenar, procesar y transmitir información en formato digital.

En este capítulo consideraremos conceptos básicos que encuentran algunas de sus raíces en un tiempo muy remoto pero cuyo desarrollo útil para llegar a los procedimientos de la informática han sido muy recientes.

1.1. El cálculo

Es evidente que el hombre ya sabía contar mucho antes de recurrir a algún aparato como el ábaco e incluso era capaz de anotar sus cuentas, como lo demuestra un monolito grabado de la era rupestre, encontrado en el sitio Presa de la Mula, en Mina (Nueva León, México) (ver al lado), que podría ser la primera "hoja de cálculo" de la historia, que el investigador William Breen Murray asocia a conocimientos astronómicos[2].



La palabra "cálculo" procede del latín *"calculus"* que significa "piedra". La razón es que los romanos usaban ábacos, en que colocaban piedrecitas (ver imagen al lado). El ábaco romano y el suanpan chino constituyen las primeras "máquinas" destinadas a contar. La mayoría de los modelos chino y romano usaban 4 más 1 cuentas por lugar decimal, por asociación en el cálculo en los dedos de la mano. Pero algunos modelos de suanpan tenían 5 y 2 cuentas por lugar decimal, permitiendo ya el cálculo hexadecimal[3].

Pero ya en el tercer milenio antes de Cristo, babilonios y egipcios sabían calcular la superficie de las principales figuras geométricas planas y calcular el volumen de las tridimensionales, y los escribas calculaban intereses comerciales. No había ningún desarrollo teórico, pero los manuales de esa época describían los procedimientos a seguir, es decir, algoritmos, para cada caso práctico (Lévy, 98).

La Grecia clásica introdujo una ruptura con esta tradición matemática, pasando del pragmatismo a la abstracción. Introdujo la geometría en que los razonamientos se aplican a las figuras independientemente de su valor numérico (Lévy, 99). Con ello introdujo el concepto de demostración, que encontramos, por ejemplo, en los teoremas de Euclides. Y esto comporta un concepto importantísimo para la ciencia: la verdad no se hereda, debe ser fundada – demostrada – aquí y ahora (*ibidem*, 100).

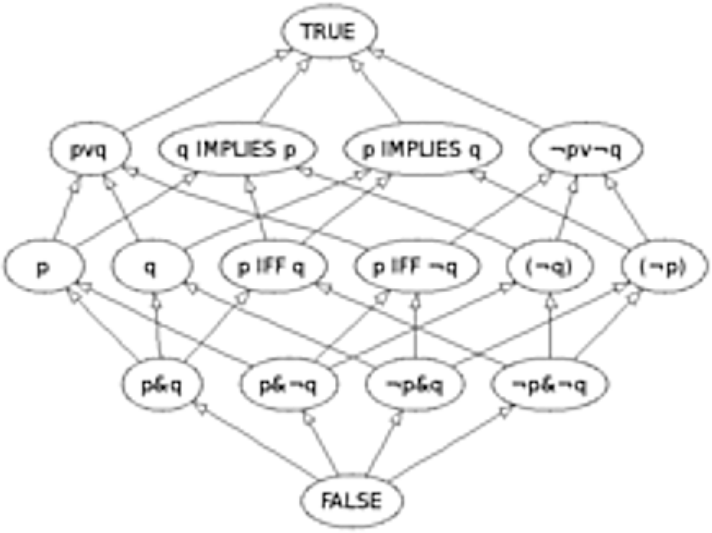
2 Murray Breen: *Arte Rupestre del Noreste.* Fondo Editorial Nuevo León, México 2007.

3 De 16 en 16. (Actualmente se utilizan las cifras 0 a 9 más las letras A a F.)

(De Wikipedia)

En el siglo XVII, Gottfried Leibniz (1646-1716) introdujo principios nuevos que serían fundamentales: la concepción del cálculo como medio de razonamiento, gracias a que opera con símbolos y sigue procedimientos exactamente especificados, lo cual no exige que se mantenga en el dominio de las cantidades. Escribió: *"Un cálculo no es otra cosa que una operación por medio de caracteres, que tiene un lugar no solamente cuando se trata de una cantidad sino también en todo razonamiento"*. (Lévy, 103) Con ello, imagina la eventual construcción de una "máquina de razonar", imposible en práctica en su época. Los lenguajes de programación son los directos herederos de esta visión de Leibniz, que también inventó el sistema binario.

Más de un siglo después, Gottlob Frege (1848-1925) realizó el sueño de Leibniz de crear una escritura formal inventando una "escritura de conceptos", una ideografía monosémica[4] con reglas de inferencia preestablecidas (Lévy, 110). Es el inventor de los cuantificadores («para todo» o «para al menos uno»).



(De Wikipedia)

Un hecho fundamental del siglo XIX corresponde al desarrollo por el británico autodidacta George Boole de una nueva álgebra. En 1847 - a los 32 años - publicó *"El análisis matemático de la lógica"*, seguido en 1854 por su obra magna *"Las leyes del pensamiento"*[5]. Su álgebra consiste en un método para resolver problemas de lógica que recurre solamente a los valores binarios "1" y "0" y a tres operadores: "AND" (y), "OR" (o) y "NOT" (no). A partir de esta "álgebra binaria" se ha desarrollado posteriormente el código binario y las operaciones que realizan todos los computadores actuales.

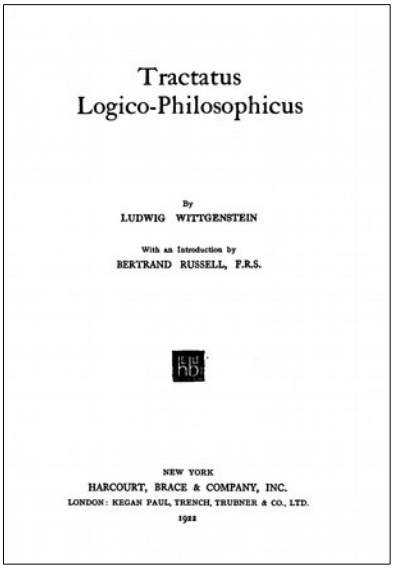
El gráfico adjunto muestra cómo pueden relacionarse dos proposiciones (p y q) de acuerdo a su lógica.

La filosofía de Ludwig Wittgenstein (1889-1951) introdujo, con su *"Tractatus logico-philosophicus"* (1923), una nueva forma de pensar acerca del mundo y de los fenómenos, donde la lógica, el cálculo, la comunicación y la información constituyen las principales categorías (Lévy, 121).

Es una visión del mundo donde las operaciones lógico-matemáticas son la base de la ciencia, que influenció a numerosos investigadores y teóricos. Para Wittgenstein, la lógica es la esencia de la comunicabilidad,

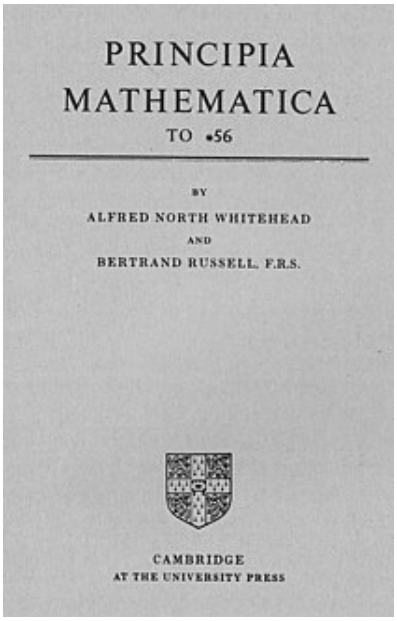
4 Es decir con una sola interpretación posible.

5 *"An Investigation of the Laws of Thought on Which are Founded the Mathematical Theories of Logic and Probabilities"*



porque ella es la base de lo "decible" (*ibidem*, 122). Si el mundo no fuera lógico, no podríamos decir nada al respecto. Todas las traducciones (en diversos lenguajes), los procesos y los cálculos tienen esa misma base.

A principios del siglo XX, la demostración, que estaba basada en el razonamiento, se transforma en cálculo, por la vía de la abstracción. El uso de símbolos desprovistos de significación (ya sugerido por Leibniz) se vuelve la norma. La racionalidad, heredada de los griegos, termina en la lógica formal y las matemáticas modernas.



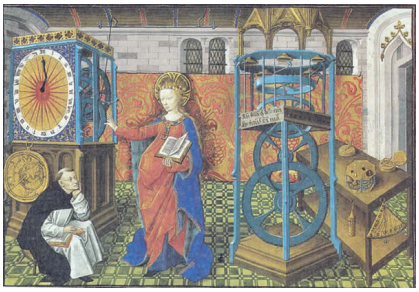
Bertrand Russell (1872-1970) y Alfred Whitehead (1861-1947) publicaron entre 1910 y 1913 los *"Principia mathematica"*, en que completaban un programa de deducción de toda la matemática a partir de la lógica, creando las bases necesarias para el surgimiento de la informática y la cibernética.

Como consecuencia, se ha de entender el cálculo de forma más amplia que su sentido matemático. Se asocia (y deriva), como demostró Russell, de la lógica, y también recubre operaciones como la clasificación, las permutaciones, comparaciones, substituciones, combinaciones e incluso la traducción de un código a otro. Todas estas operaciones, en realidad, pueden ser reducidas a la combinación de dos o tres formas matemáticas fundamentales, que es exactamente lo que permite que un computador las haga todas (Lévy, 72).

1.2. La máquina

El ayudarse a calcular recurriendo al ábaco entraña la aparición y proyección de otro concepto: el de máquina. El término `máquina´ proviene de la palabra griega *"majaná"*, que se deriva del término *"mijós"*, que significa "medio, expediente, recurso". El vocablo *"majaná"* denotaba tanto lo que hoy llamamos máquina como cualquier medio que permitiese alcanzar un fin.

De los griegos heredamos también el concepto de inteligibilidad basada en el cálculo y la geometría (Lévy, 101-102). En la Edad Media, se concibió el cosmos como *"machina mundi"*, con una perfección que sólo podía ser procurada por un "relojero divino" (Prieto, 25).



"Horologium Sapientiae"
(El reloj de la sabiduría)[6]

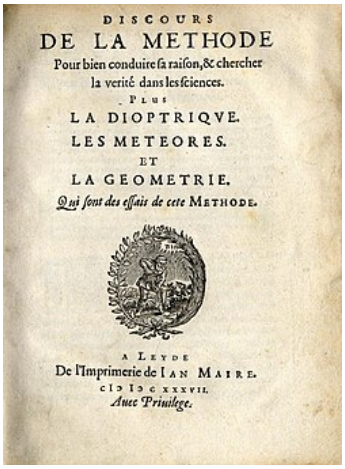
Esta ilustración adjunta apunta a la metáfora del reloj como modelo del conocimiento.

La metáfora del reloj, epítome del modelo mecánico y exitoso debido a su carácter intuitivo, tuvo una larga tradición en Occidente y se haría aún más intensa durante la llustración (Prieto, 24). Se consideró que tanto el universo biológico como la anatomía podían ser interpretados de modo mecánico.

En el siglo XVII, el mecanicismo llegó a transformarse en una suerte de ideología, llegando a influenciar pensadores de la importancia del francés Descartes. Para los racionalistas, que siguieron a Descartes, la máquina *"era el símbolo de los poderes del «hombre del futuro»"*. (Prieto, 13).

El racionalismo posterior ha sido en cierto modo una consecuencia de esta forma de pensar, unida al enfoque griego acerca de la verdad basada en la demostración. Máquina y cálculo al servicio del conocimiento es lo que encontramos en la informática moderna.

1.3. Los sistemas

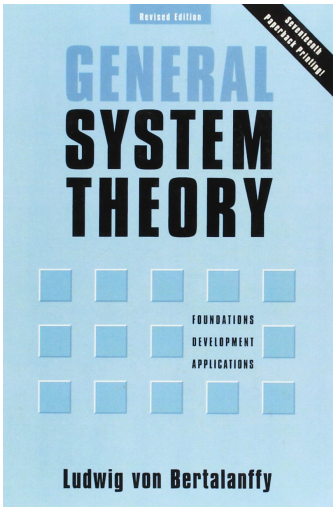

DISCOURS
DE LA METHODE
Pour bien conduire sa raison, & chercher
la verité dans les sciences.
PLUS
LA DIOPTRIQVE.
LES METEORES.
ET
LA GEOMETRIE.
Qui sont des essais de cette METHODE.
A LEYDE
De l'Imprimerie de IAN MAIRE.
cIↃ IↃ c XXXVII.
Avec Privilege.

En su *"Discurso del método"*, René Descartes (1596-1650) tomó como modelo el método matemático para descubrir la verdad por la sucesión de evidencias con certeza siguiendo reglas racionales. Pero es interesante descubrir que introdujo al mismo tiempo la noción de sistema, es decir, *"la idea de una suma de partes organizadas deductivamente de acuerdo con principios ciertos y armonizados entre sí para conseguir un propósito."* (Prieto, 27)

Los ingenieros elaboraron una nueva noción de sistema en las primeras escuelas politécnicas de fines del siglo XVII, asumiendo el mecanicismo y el nuevo concepto de utilidad que se iría confundiendo con el de función (Prieto, 46). La descripción del sistema

> *"era, en puridad, el análisis cartesiano (resolutio), y consistía en hallar las partes fundamentales de un objeto. Estas se concebían como unidades mínimas de sentido cuya razón no podía entenderse fuera del conjunto que explicaba su funcionalidad última: la llamada «estructura»." (Prieto, 47)*[7]

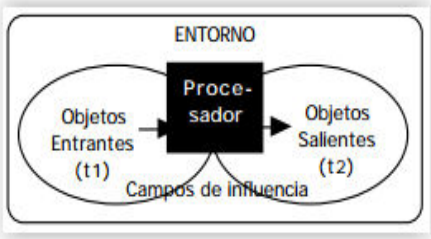
6 Miniatura de Henrich Seuse (ca.1450). Biblioteca Real Albert, Bruselas.
7 Prieto, L. (1972): *Mesages et signaux*, PUF, Paris.



La modelización sistémica emergió con fuerza a mediados de los años 1970, siendo su raíz la publicación de la obra de Ludwig von Bertalanffy, *"General system theory, foundation, development, applications"*, de 1968.

Los teóricos norteamericanos de la sistémica definieron muchas veces los sistemas como "conjuntos de elementos en interacción", lo cual llevó erróneamente a ver en la teoría de conjuntos una herramienta matemática adecuada y que presidió al desarrollo del método de "análisis de sistemas" basado en el reconocimiento y enumeración de los componentes y sus relaciones, una concepción muy cartesiana que ha impregnado fuertemente la difusión de la sistémica. Pero esta perspectiva *"pierde involuntariamente la excepcional fecundidad del concepto de sistema fundado en la dialéctica de lo organizado y lo organizante"* (Mucchielli, 19) y principios tan importantes como la sinergia[8] y la emergencia[9].

Cinco son los conceptos definitorios de un sistema: la actividad, la estabilidad, la finalidad, la evolución y la inserción en un entorno. El sistema u objeto puede ser de cualquier naturaleza: concreto o abstracto, tangible o intangible. La ambigüedad del término "sistema" en el lenguaje ordinario no plantea problema alguno, al contrario: da cuenta de que se perciben - aún confusamente - rasgos comunes en numerosos objetos o fenómenos diversos. Un objeto puede ser definido de tres maneras: recurriendo a su esencia (definición ontológica u orgánica), a lo que hace en su contexto (definición funcional o experimental), o a cómo se ha generado y cómo se desarrolla (definición genética o histórica). El cartesianismo ha privilegiado la primera forma; la sistémica intenta asumir las tres (Le Moigne, 65). Estas tres definiciones juntas cubren los cinco conceptos definitorios.

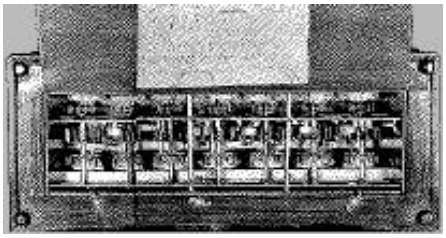


La representación en forma gráfica es parte importante de la sistémica. Ayuda a visualizar tanto la estructura como las relaciones de los componentes. Para facilitar este método de trabajo William Ross Ashby[10] introdujo el concepto de "caja negra": ingenio que procesa una entrada y genera con ello una salida, entrada y salida siendo elementos pertenecientes al entorno (Es el modelo E-P-S: entrada-proceso-salida; gráfico al lado). Es "negra" en el sentido de que, al menos inicialmente, no se conoce lo que hay ni lo que ocurre "adentro": sólo se sabe que "hace algo", que "opera una transformación", es decir, que "procesa" elementos (objetos) provenientes del entorno.

8 Efecto que sólo produce un conjunto organizado, inexistente en sus partes.
9 Aparición de estructuras complejas a partir de reglas simples.
10 Nacido en Londres en 1903, publicó en 1956 "Introducción de la cibernética", considerado un clásico.

2. Antecedentes

2.1. Máquinas lógicas



La calculadora de Pascal
(De mi curso *"Conceptos básicos de computación"*)

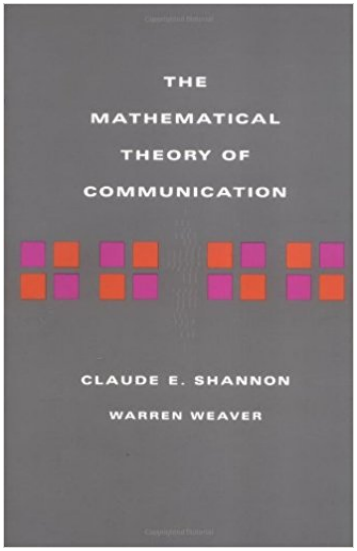
Podríamos citar los esfuerzos realizados desde el Renacimiento para construir máquinas calculadoras, como la primera máquina sumadora mecánica que construyó Blas Pascal en 1642 (al lado), seguida de varios otros modelos que la perfeccionaron.

Algo diferente y más avanzado ha sido la "máquina de diferencias" que mostró en 1821 Charles Babbage (1792-1871), capaz de resolver ecuaciones polinómicas mediante el cálculo de diferencias sucesivas.

En 1833, inició la construcción de una versión mayor y más versatil pero murió sin lograr terminarla, especialmente en razón de la imprecisión en la fabricación de los engranajes, que tendían a bloquearse continuamente. Aparte de su capacidad de calcular, pretendía que fuese capaz de organizar información registrada en tarjetas perforadas, imprimir sus resultados y - sobre todo - evaluar un resultado para determinar ella misma qué cálculos hacer a continuación. En otras palabras, introducía el principio lógico de evaluación (si...entonces...) y un mecanismo de retroalimentación (el dato salido vuelve a entrar), principio que sería medular en la cibernética que nacería un siglo más tarde.

En la línea del desarrollo de "máquinas lógicas", Leonardo Torres y Quevedo creó en España, entre 1893 y 1920, varias máquinas capaces de resolver ecuaciones albegráicas. Más tarde construyó la primera máquina capaz de jugar al ajedrez. En 1914 escribió *"Ensayos sobre la Automática"*, en que describe conceptualmente un aparato capaz de razonar conforme a reglas determinadas por su fabricante. No creía, sin embargo, que fuese posible construirla con los materiales de su tiempo (a diferencia de la opinión errónea de Babbage, cuyos trabajos conocía).

2.2. Teoría de la información



En 1937, Claude Shannon demostró definitivamente que la programación de futuros computadores era un problema de lógica más que de aritmética. Con ello señalaba la importancia del álgebra de Boole, pero -además- sugirió que podían usarse circuitos eléctricos de conmutación como en las centrales telefónicas, idea que sería decisiva para la construcción del primer computador, el que siguió justamente este modelo. Toda la informática descansa en este principio, que permite asimilar procesos físicos con operaciones lógicas.

Con posterioridad y con la colaboración de Warren Weaver, Shannon desarrolló lo que llamó "teoría matemática de la comunicación" - hoy más conocida como "Teoría de la Información" -, estableciendo el concepto de "negentropía" (la información reduce el desorden, es decir, que es lo inverso

de la entropía[11]) y la unidad de medida del "bit" (*binary digit*), universalmente conocida y aplicada tanto en telecomunicaciones (que es el campo a partir del cual trabajaron Shannon y Weaver) como en informática.

"Toda la informática descansa en el descubrimiento de que los procesos físicos pueden ser exactamente isomorfos[12] con las operaciones lógicas." (Lévy, 105) Aunque los circuitos de los computadores realizan las operaciones de la lógica de Boole, la informática recurre a nociones de lógica matemática mucho más complejas (que no es del caso desarrollar aquí).

2.3. Máquina universal


Turing

Claves para el desarrollo posterior de la informática y de los computadores han sido los trabajos de Alan Turing (Reino Unido, 1912-1954). A los 22 años fue nombrado profesor en el King's College de Londres. En 1936, propuso un modelo muy simple de máquina destinada a tratar información. Su tesis era que todos los procesos que pueden descomponerse, con un alfabeto restringido, en una secuencia finita y ordenada de operaciones podían realizarse en una máquina. Esta máquina se compondría de un módulo de entrada de datos, otro de consulta de reglas de comportamiento (seleccionando la que corresponda al dato leído) que, luego, ejecutaría, para comparar luego el producto de nuevo según sus reglas ("instrucciones"), imprimiendo el resultado. Y concluyó que existía una clase de máquina que podía resolver todos los problemas calculables o realizar todos los procedimientos descriptibles. La llamó "la máquina universal".

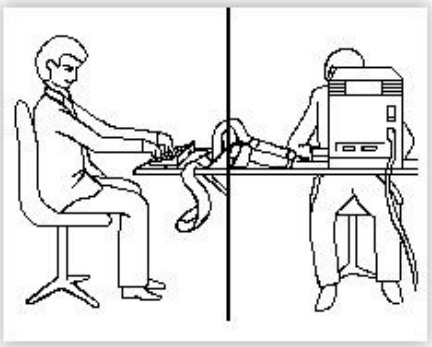
Para que una máquina sea "universal", debe aceptar dos tipos de entradas: los datos del problema y las instrucciones correspondientes a las operaciones a realizar (obviamente debe también poder exhibir el resultado).

Se puede representar gráficamente como se muestra aqui.

En 1937, Turing concibió ya un proyecto (teórico) de cerebro artificial pero se encontró detenido por su desconocimiento de la neurofisiología. En 1947 publicó "Maquinaria inteligente", sobre el tema de la inteligencia artificial, donde comparaba los ordenadores a los cerebros "por programar" de los bebés. Inventó la prueba de diálogo conocida con su nombre: si no podemos distinguir entre un interlocutor vivo y una máquina, ésta puede ser considerada como "inteligente".

11 La entropía es la parte no utilizable de la energía en el funcionamiento de un sistema. Crece constantemente porque no es nunca totalmente recuperable la pérdida de calor que se produce.

12 De forma semejante.



(Forma simplificada de la prueba de Turing)

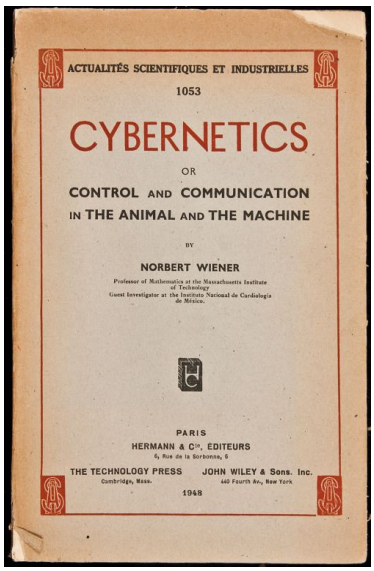
La "Prueba de Turing"

Un interrogador permanece encerrado en una pieza, enviando mensajes por un buzón a dos interlocutores, un hombre y una mujer, siendo su objetivo determinar quién es el hombre y quién la mujer (sin preguntar, obviamente, por el sexo de cada uno). En el modelo inicial de la prueba, se supone que el varón quiere hacerse pasar por mujer. Luego se pregunta: ¿qué ocurrirá si el varón es reemplazado por una máquina? Si el diálogo que ocurra y el número de errores en la solución dada se acerca al número de errores ocurridos en la comunicación con un ser humano, se podrá estimar -según Turing- que estamos ante una máquina "inteligente".

Turing también es autor de la demostración matemática de que sería imposible redactar un programa computacional (serie finita de instrucciones) capaz de analizar otro programa y predecir si este -de tener algún sistema de recursión o autocontrol- provocaría o no una repetición infinita de las operaciones previstas.

2.4. Cibernética

En 1949, Norbert Wiener se enfrentó al problema de dirigir las trayectorias de proyectiles hacia objetos en movimientos, como aviones enemigos. Para acertar, debe predecirse la posición futura del blanco, y corregirse la trayectoria si este cambia de dirección. El equipo de Wiener se dió cuenta de que era un problema semejante al que resuelve el cerebro cuando conduce la mano para recoger un objeto (estático o en movimiento). Así se propusieron crear un aparato que imitaría los procesos de control existentes en el ser humano. Incluyendo fisiólogos en su equipo y recuperando el concepto de retroalimentación (*feed-back*) formulado por Babbage, echaron las bases de la cibernética, disciplina hoy rectora de los procedimientos automáticos. Aunque se aplica en la construcción de robots controlados digitalmente, no debe confundirse con la robótica



El término cibernética viene del griego *"kybernḗtēs"*, que se refiere al timonel, el cual "gobierna" la embarcación. La palabra francesa *"cybernétique"* fue utilizada en 1834 por el físico André-Marie Ampère (1775-1836) para referirse a las ciencias de gobierno en su sistema de clasificación de los conocimientos humanos. La idea (y desarrollo) de máquinas con retroalimentación, sin embargo, ya existía en la Grecia clásica (Wikipedia).

En 1949 fue publicado el resultado teórico de los trabajos del equipo de Wiener bajo el título de *"Cybernetics"*. La naciente cibernética se definió como "teoría de la comunicación y autorregulación en sistemas probabilistas extremadamente complejos". Estudia los sistemas de comunicación y de regulación automática de los seres vivos para aplicalos a sistemas electrónicos. Es, por lo tanto, eminentemente interdisciplinaria,

involucrando tanto a la física como al estudio del cerebro, a pesar de que se considera generalmente como una rama de las matemáticas.

Gregory Bateson y Stafford Beer son otros importantes teóricos de los inicios de la cibernética.

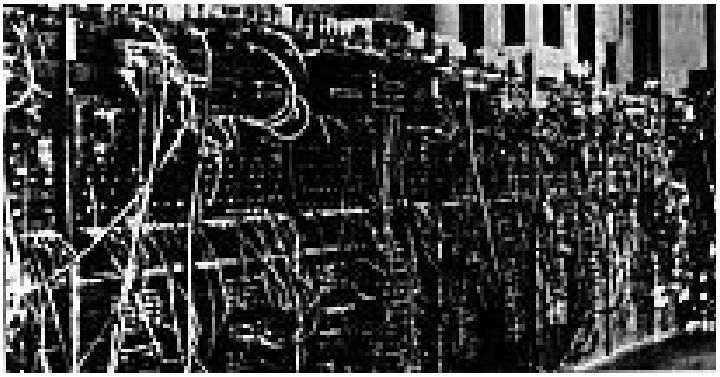
3. Conceptos y componentes

La informática ha heredado el concepto de información de Shannon y Weaver asi como su definición del bit y la codificación binaria. También heredó el concepto de máquina de Turing, junto al concepto de instrucciones para realizar las operaciones. De Babbage heredó el concepto de retroalimentación, perfeccionado por Wiener.

3.1. Arquitectura

El aporte más significativo, que aún hoy sigue determinando la estructura de los computadores de todos los formatos, ha sido el modelo desarrollado por John von Neumann y que lleva desde entonces su nombre.

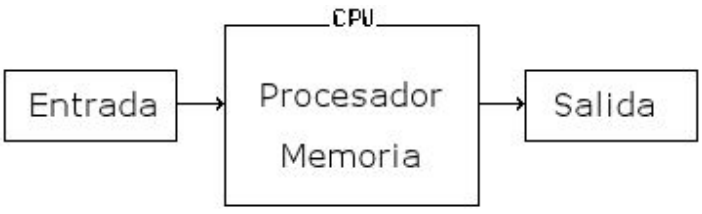
Arquitectura de von Neumann



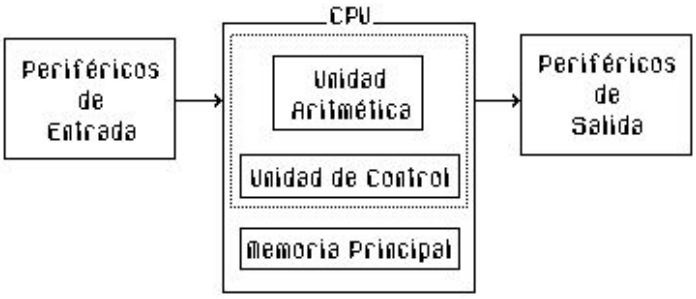
La ENIAC

John Von Neumann era un matemático de origen húngaro que trabajaba en 1947 en el laboratorio atómico de Los Alamos. Tuvo la oportunidad de conocer el ENIAC (*Electronic Numerical Integrator and Calculator*), el primer computador electrónico, compuesto de 17.468 válvulas o "tubos", que pesaba 30 toneladas y en el cual las secuencias de operaciones se definían cambiando los cables de la máquina para cada nueva tarea, una tarea enorme (vea la foto al lado de esta parte de la máquina).

En 1949 logró encontrar y desarrollar la solución a este problema, consistente en poner la información sobre las operaciones a realizar en la misma memoria utilizada para los datos, escribiéndola de la misma forma, es decir, en código binario. Todas las computadoras construidas a continuación siguieron este modelo, aunque propuso también otros.



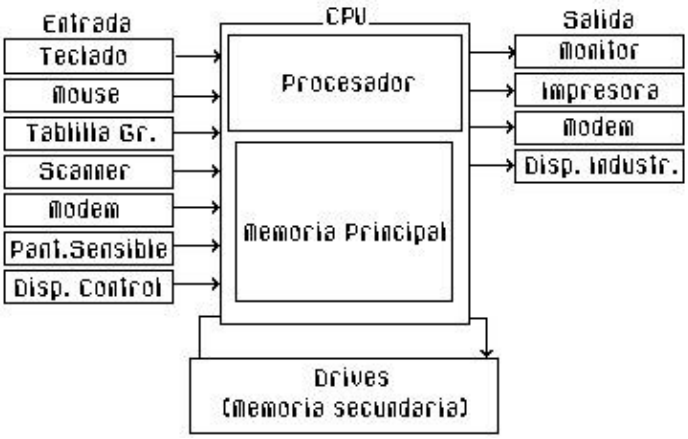
Recordemos que el computador es una máquina destinada a "procesar información". El primer computador recibía datos y órdenes uno a uno a medida que estaba en condiciones de hacer una nueva operación. Desde von Neumann, incluye una memoria capaz de contener los datos y las órdenes, para luego "trabajar sola" hasta lograr el resultado. Es la "arquitectura" graficada al lado.



La Memoria conserva la información indispensable para operar, la Unidad Aritmética efectua las operaciones y la Unidad de Control asegura el correcto flujo de información desde la entrada hacia la memoria, desde ahí hacia la unidad aritmética, luego de nuevo hacia la memoria y, por fin, hacia la salida. (Estudiaremos estos componentes detalladamente más adelante.)

Tanto los computadores con gran volumen de memoria y alta velocidad de proceso (llamados *"main-frames"*) como los computadores personales, los *"smartphones"* y las máquinas de video-juegos (consolas y otras) tienen una CPU con esta misma estructura y que cumple las mismas funciones básicas.

Periféricos



Los "periféricos" son los aparatos que utilizamos para introducir y recibir la información, como el teclado o la cámara para la entrada y la pantalla o impresora para la salida. Los "drives" pueden ser unidades internas, lectores de CD, *pendrives* (USB) o tarjetas, donde se conservan datos o incluso programas.

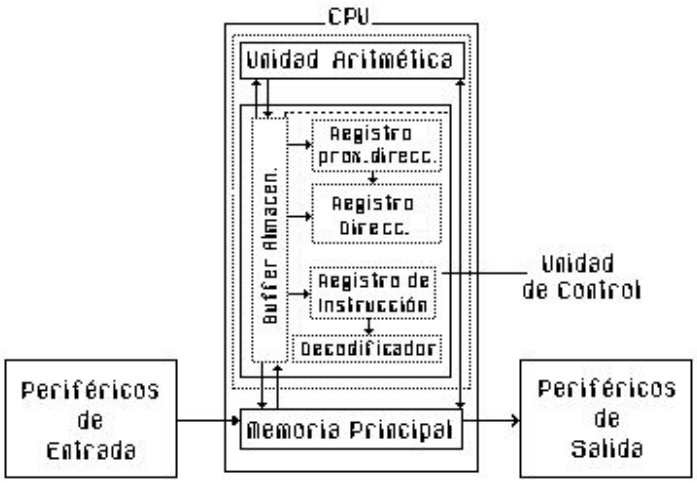
Esto lleva a una primera estructuración: debe haber una unidad de memoria y una unidad que procesa, y esta memoria debe poder contener el programa y los datos asi como resultados parciales de las operaciones en curso. Estas dos unidades básicas conforman la "Unidad Central de Procesos" (CPU o *"Central Process Unit"*, en inglés).

Programa - *Software*

Paralelamente, von Neumann concibió lo que se ha llamado el "programa", o sea el conjunto estructurado de órdenes o instrucciones de trabajo que guían, paso a paso, el funcionamiento de la máquina. Los diversos programas (aplicaciones o *"apps"*) constituyen el llamado *"software"* ("material blando"), en contraposición con el *"hardware"* ("material duro") que es el conjunto de los componentes electrónicos y mecánicos requeridos.

Unidad de Control

La Unidad de Control - encargada de coordinar todos los componentes y los flujos de datos - es muy compleja en computadores grandes. Sin embargo, su estructura en computadores pequeños es una buena ilustración de sus sistemas básicos. Se presenta como en el gráfico adjunto y se compone de:

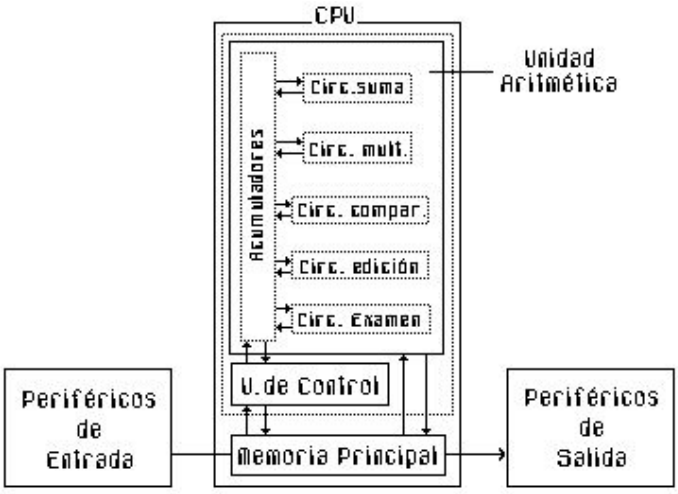


- la memoria tampón o "buffer de almacenamiento", donde se mantienen temporalmente datos que fluyen desde o hacia la memoria principal (la RAM);
- el registro de próxima dirección, que contiene la dirección de la siguiente instrucción a ser ejecutada;
- el registro de dirección, que contiene la dirección de la celda de memoria RAM en la que se está leyendo o escribiendo;
- el registro de instrucciones, que contiene el código de la instrucción en curso de ejecución;
- el decodificador, dispositivo que interpreta la instrucción y dirige los flujos de información de manera que la instrucción sea llevada a cabo adecuadamente.

En máquinas más complejas se agrega, además, un controlador de entradas y salidas, que administra todo el flujo hacia y desde los periféricos.

Unidad Aritmética

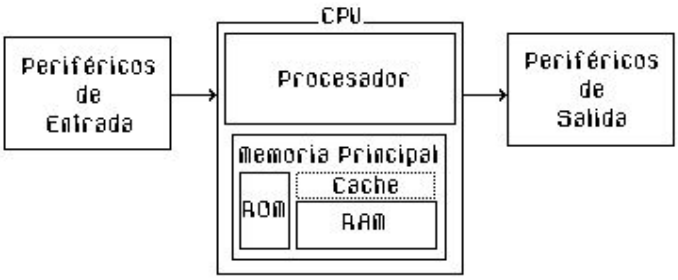
La Unidad Aritmética es la que realiza las operaciones tanto aritméticas como lógicas que se encomiendan al computador, dirigida por la Unidad de Control. Comprende:



- uno o varios registros de memoria llamados "acumuladores", donde se guardan los resultados parciales de las operaciones y el resultado final hasta que sea transmitido al buffer de la Unidad de Control;
- circuitos de suma/resta, multiplicación/división, comparación (mayor/menor), edición (modificar un dato) y examen (saber el estado del acumulador, por ejemplo, si está vacío o hubo un rebalse de bits, lo cual es muy importante para el control interno).

Memorias

Todo computador trae en realidad dos o tres tipos de memoria:



1. Memoria ROM

Para facilitar el trabajo del usuario, trae una especie de "programa maestro" que contiene instrucciones básicas para la Unidad de Control (normalmente qué hacer al momento de encenderse y cómo realizar las operaciones lógicas -base de su poder-), de tal modo que el operador no tenga que ocuparse del funcionamiento electrónico de la máquina. Este programa especial ocupa una parte reservada de la memoria, que puede ser leída pero no alterada: es la ROM (*"read-only memory"*), generalmente contenida en un chip especial. Al apagar el computador, esta información no desaparece, a diferencia de la memoria RAM.

2. Memoria RAM

La memoria disponible para recibir los programas y los datos es la RAM (*"random-access memory"*), o memoria de acceso aleatorio. Se llama así por cuanto es posible acceder directamente a cualquier lugar de ella si se conoce la "dirección" de la celdilla que contiene la información que interesa. Está constituida habitualmente por varios chips.

3. Memoria Caché

Muchos computadores cuentan, además, con otra área de memoria: el "caché" ("escondite"). Es una área reservada de la RAM para conservar los datos de uso más frecuente (requeridos por el procesador) junto con la dirección de los mismos en la RAM normal. Cuando la Unidad de Control determina que necesita el dato de tal celdilla RAM, lo "manda a buscar" en ambos lugares y utiliza el que le llega primero.

Puertos

La CPU se conecta con los periféricos y -eventualmente- con una red a través de un sistema de conexión llamado "puerto". Existen puertos físicos y puertos "virtuales" (definidos por el *software*).

Los puertos físicos pueden ser:

- el puerto serial: envía y recibe datos uno tras otro, lo que lo hace más lento
- el puerto paralelo: envía y recibe datos en forma paralela (simultánea) por varios hilos
- el puerto USB: substituye actualmente los antiguos seriales y paralelos, por ser más eficientes (adecuado para *mouse*, teclado, impresora, cámara, etc.)
- el puerto IEEE 1394 o "FireWire", que se utiliza especialmente para conectar periféricos con alto flujo de datos, como cámaras de video
- otros: la tecnología avanza y otros tipos de puertos aparecen regularmente en nuevos equipos (como los puertos para tarjetas SD).

Los puertos definidos por *software* son más de 65 mil, por lo cual ni se conocen ni se utilizan todos. Están relacionados con el tipo de conexión a la red. Los de uso más frecuente son:

- el puerto 21, para la transmisión de archivos (FTP)
- el puerto 23, para la conexión remota a otro computador
- el puerto 25 y el 110, para el servicio de correo electrónico
- el puerto 80 o http, que da acceso a la web.

Para una mayor protección contra la infiltración de terceros, se "cierran" los puertos que no se utilizan o se define con qué otra máquina se permite la conexión. Hacer estos ajustes es lo que corresponde a la definición de un "corta-fuego" (*Firewall*), para lo cual existen comandos del sistema operativo o, en un nivel de más fácil uso, aplicaciones especializadas. (Del sistema operativo hablaremos más adelante.)

3.2. Lenguajes

Lenguajes de Bajo y Alto Nivel

Lo único que entienden y pueden manipular la Unidad de Control y la Unidad Aritmética son dígitos binarios o sea series de ceros y unos (paso o no paso de corriente eléctrica). Así, mientras el hombre usa un "lenguaje natural" (idioma) muy rico en significados, la máquina usa un sistema en que existiría un sólo "significado": la diferencia entre 0 y 1 (o sea un bit de información). ¿Cómo lograr más? Simplemente concibiendo un nuevo lenguaje constituido de bloques de dígitos binarios (llamados "bytes"). Este es el primer paso o "primer nivel" en la construcción de lenguajes de computación.

1º Nivel

Tecla	Decimal	Hexadecimal
1	49	31
2	50	32
9	57	39
a	97	61
b	98	62
!	33	21
?	63	3F
=	61	3D
n	110	6E
o	111	6F
p	112	70
z	122	7A

Se dice que el código binario es de "bajo nivel" o "primer nivel" (porque al usar pocos signos logra muy difícilmente expresar cosas complicadas), mientras un lenguaje humano es de "muy alto nivel" (con una cantidad mayor de signos y con reglas combinatorias logra expresar con facilidad cosas muy complicadas). Todo el esfuerzo, entonces, para facilitar la comunicación del hombre con el computador, ha de centrarse en el desarrollo de lenguajes de mayor nivel.

El fabricante de un procesador fija los bloques de bits que permitirán a la CPU reconocer y realizar diferentes operaciones. Este es el llamado "lenguaje de máquina", primer lenguaje que la máquina puede interpretar y transformar en acciones. Pero es, evidentemente, muy difícil de usar para un ser humano. Supongamos que quiera hacer imprimir y para ello deba decir "10011101 11100010": ¿cómo recordar órdenes de este tipo y no equivocarse al escribirlas?

Prácticamente nadie trabaja hoy a este nivel, excepto los diseñadores de chips procesadores. Del mismo modo que es posible pasar de un sistema binario a un sistema decimal (más comprensible y más desarrollado en términos de signos legibles) es posible asociar a los bloques de bits no sólo valores decimales sino, también, otros signos. Esto lleva a un segundo nivel de expresión.

Hemos de recordar que el teclado equivale a un conjunto de interruptores: cada tecla que pulsamos equivale a cerrar brevemente uno de éstos, es decir, produciendo un bit de información (no teclear = 0,

teclear =1). Pero dado que hay muchas teclas, hay que identificar cada una, por lo cual pulsar una tecla significa activar un circuito que generará un "bloque" binario (byte) específico que identifica esa tecla.

A cada tecla está asociado un código decimal y un código hexadecimal. El hexadecimal (16 caracteres: de 0 a 9 y de la A a la F) es el que sirve de intermediario a la máquina, para traducir nuestro código natural (alfanumérico) al código binario. Algunos ejemplos de equivalentes decimales y hexadecimales del teclado aparecen a la izquierda.

2º Nivel

Assembly Language
add $t1, t2, $t3
addi $t2, $t3, 60
and $t3, $t1, $t2
andi $t3, $t1, 5
beq $t1, $t2, 4
bne $t1, $t2, 4

La creación de un lenguaje más comprensible por el hombre consiste por lo tanto en establecer la equivalencia de bloques binarios con signos de nuestro lenguaje habitual. Para permitir la programación (secuencia de comandos), se usan pequeños conjuntos de signos ("palabras") de fácil memorización, con las cuales se redactan programas, por ejemplo "ADC" significará "sumar con reserva" (en inglés: *"ADd with Carry"*). Este tipo de lenguaje se llama "ensamblador", "Assembler" o *"Assembly language"*. La máquina misma hará la tarea de traducirlo en código binario, para seguir las instrucciones, gracias a otro programa cuya función es traducir la expresión humana en "lenguaje de máquina" (binario). Este programa se llama "compilador" (vea el apartado sobre *"Hardsoftware"*).

Aunque el Assembler es un inmenso progreso en relación al código binario, su desventaja reside en que permanece estrechamente ligado a los bloques binarios que reconoce la CPU (es decir, al *hardware*). Para facilitar más la tarea, se han inventado lenguajes de "alto nivel", es decir, más cercanos al modo de expresar del hombre que de operación de la máquina. Los primeros y más comunes son los llamados de "tercera generación", más fáciles de manejar y más independientes de las características técnicas de los procesadores. Ahora, hasta un aficionado puede llegar a redactar un programa, sin tener que preocuparse por el código binario o de ensamble: si un programa traductor podía resolver la transformación de bloques de signos en bloques binarios, era cosa de extender las habilidades del traductor para "enseñar" a la máquina cómo "entender" un lenguaje más complejo y agregar mecanismos automáticos de manejo de la memoria para poder utilizar lenguajes aún más comprensibles.

3º Nivel

El avance en el desarrollo de "compiladores" e "intérpretes" (los dos tipos de programas traductores) ha sido por lo tanto fundamental en el desarrollo de los lenguajes de "3º generación" cuyas ventajas además de la facilidad de aprendizaje y lectura/escritura son las facilidades de corrección, transformación y conversión de un lenguaje a otro.

```
013650 C140-FIND-END-LNAME-EXIT.    EXIT.
013660     SKIP2
013670 C150-EDIT-SUFFIX.
013680     IF SSUB > 3
013690         MOVE 1 TO END-OF-NAME
013700         GO TO C150-EDIT-SUFFIX-EXIT.
013710     IF S-NAME (SSUB) = SPACE
013720         MOVE 1 TO END-OF-NAME
013730         GO TO C150-EDIT-SUFFIX-EXIT.
013740     ADD 1 TO LSUB.
013750     MOVE S-NAME (SSUB) TO NAME-CHARACTER (LSUB).
013760 C150-EDIT-SUFFIX-EXIT.  EXIT.
```

(Ejemplo de Cobol)

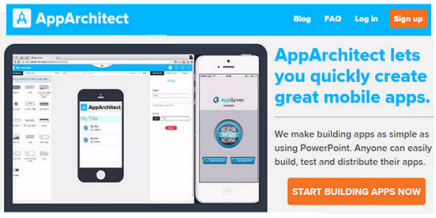
Los más antiguos son el FORTRAN (para aplicaciones matemáticas y científicas) y el COBOL (para aplicaciones de administración y contabilidad), hoy raras veces utilizados. Son numerosos los que los siguieron y no es nuestro objetivo tratarlos aquí.

Es necesario distinguir entre:

- un lenguaje de programación, como C++, PERL o Java
- un lenguaje de manejo de base de datos, como SQL (*"Structured Query Language"*)

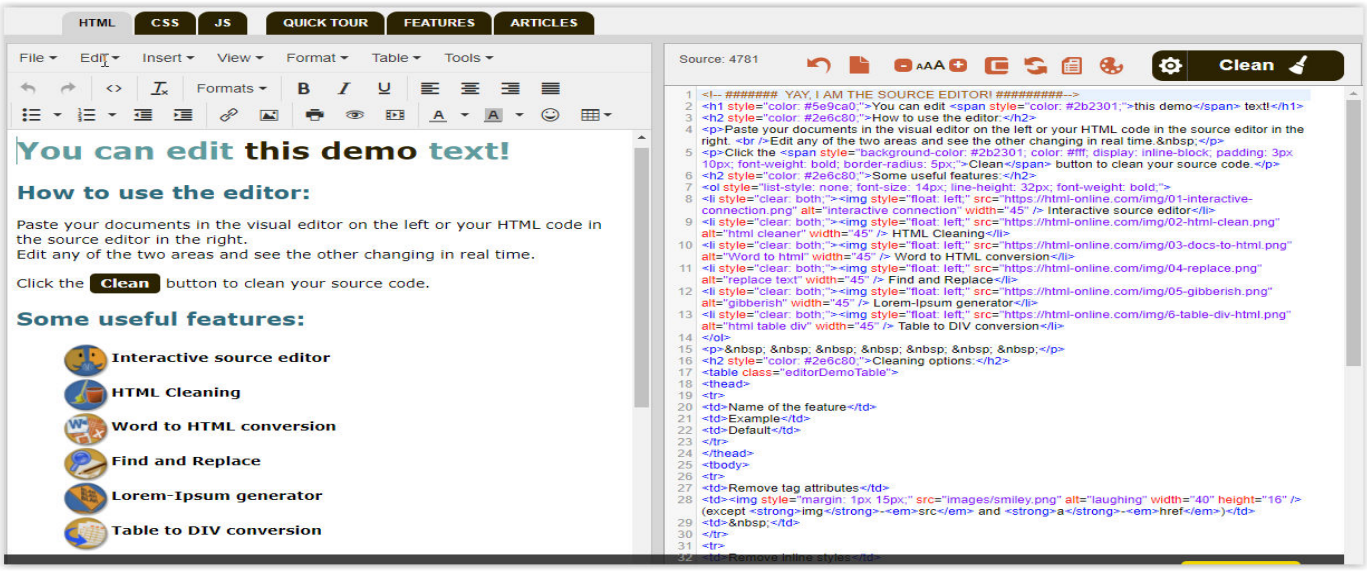
- un lenguaje de despliegue como es el HTML (“*HyperText Marking Language*”) que sirve para definir cómo mostrar el contenido de una página web.

Generadores de aplicaciones



Posteriormente, usando estos lenguajes, se han creado programas destinados a facilitar la creación de *software* sin utilizar lenguajes de programación. Los hay por ejemplo, hoy, para crear con cierta facilidad “*apps*” para teléfonos celulares inteligentes (*smartphones*), como *“AppArchitect”* (al lado). Existen muchas alternativas en este nivel, que nos demuestran que no es necesario aprender un lenguaje de programación y “codificar”, pero es indispensable dominar la lógica computacional y saber “desarmar” un problema para definir los pasos que conducirán a su solución en un computador.

Su modo de trabajo es lo que se llama “WYSIWYG” (*“what you see is what you get”*, es decir, lo que ve en pantalla es lo que obtendrá como producto. Mostramos un ejemplo a continuación donde, en la mitad derecha aparece el código que corresponde a lo que inserta en la ventana de la izquierda, donde se puede trabajar como si fuese un procesador de texto.



Estructura de un Lenguaje

Todo lenguaje, para permitir la programación, ha de contener diversos tipos de instrucciones:

Instrucciones simples:

- de entrada: para buscar y recoger datos en la memoria central o auxiliar, o bien obtenerla por interacción con el usuario (p.ej., mensaje en la pantalla que debe ser contestado en el teclado),
- de salida: datos expuestos en la pantalla o impresos, o transmitidos en una red,

- de asignación: asignar un valor a una variable, sea directamente (p.ej., *Variable1 = 15*) sea por cálculo (p.ej., *Var3 = Var1 + Var2*).

Instrucciones compuestas:

- de secuencia: por principio el orden dado a las instrucciones determina el orden en que se ejecuten, salvo instrucciones especiales de "salto" como las previstas en instrucciones de alternación o iteración.
  Toda instrucción compuesta debe tener una ENTRADA y una SALIDA. Puede siempre reemplazarse una instrucción simple por una compuesta manteniéndose el principio de secuencialidad de las instrucciones.
- de alternación: escoger entre dos alternativas en función del cumplimiento de una condición (p.ej., *if X > 15 then ... else ...*: si X es mayor que 15 haga esto, sino este otro)
- de iteración: ejecutar repetidamente un grupo de instrucciones mientras se cumpla una condición (p.ej. para contar de 1 hasta 10: N=1, while N < 10 repeat {N = N+1, print N}. O sea mientras N sea inferior a 10, agregue 1 al valor anterior de N).
  En este ejemplo, se introduce un concepto muy importante en el desarrollo y uso de lenguajes de programación: la recursividad, factible por el hecho de que lo que se manipula es siempre un valor colocado en alguna celda de memoria. Así, si bien la matemática no puede aceptar una ecuación como N=N+1, aquí estamos ante una instrucción (no una ecuación) que significa *"tomar el valor que está en una celda llamada N, sumarle 1 y volver a colocar el nuevo valor en la celda llamada N"*. Ésta es una "instrucción de asignación".

Modos de programación

Programación no estructurada

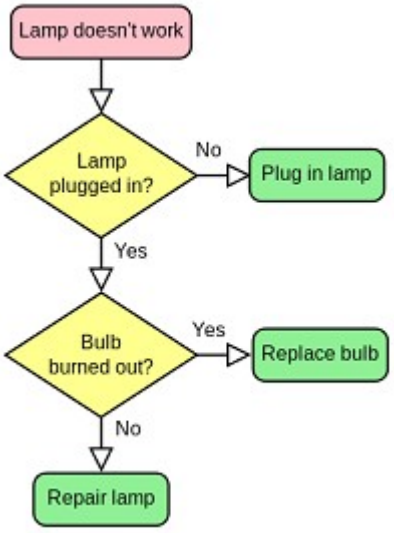
```
open "File1"
read acceso
close "File1"
if acceso <> 1 then
hide fld "Ver"
else show fld "Ver"
end if
put "p." into fld "P"
```

Todo programa se compone de una secuencia de instrucciones que pueden ser simples o compuestas. Podemos simplemente escribir todas las instrucciones una tras otra, lo cual corresponde a la modalidad de programación no estructurada, que es poco eficiente porque es frecuente la presencia de instrucciones que deban repetirse muchas veces. Por ello se prefiere una forma estructurada o modular.

Programación estructurada

La esencia de la programación estructurada corresponde a la idea de descomponer el problema a resolver en sus partes (trataremos este aspecto en detalle en el siguiente capítulo) para simplificarlo, etapa por etapa. Recurre a tres tipos de estructuras básicas:

- Estructuras secuenciales: cada acción sigue ordenadamente a otra acción. La salida de una acción es la entrada de otra.
- Estructuras selectivas: evalúan condiciones y, según el resultado de las mismas, se realizan unas acciones u otras.
- Estructuras repetitivas: instrucciones que se repiten un número determinado de veces.



Además, pone énfasis en la conveniencia de facilitar la lectura de los programas haciendo más visible la dependencia jerárquica de las instrucciones compuestas mediante "indentación", es decir, modificando el ancho del margen izquierdo para cada grupo de instrucciones.

La mejor guía para preparar una programación estructurada es la confección de un diagrama de flujo, como lo veremos más adelante.

Programación modular

```
let Archivo = "File1"
do Procedure "Acceso"
do Procedure "Lectura"
....
Procedure "Lectura"
   open Archivo
      read acceso
   close Archivo
end.
```

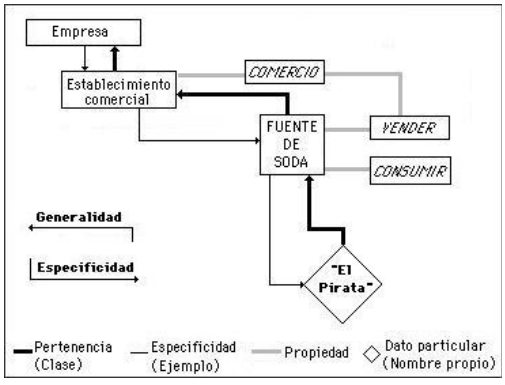
```
Procedure "Acceso"
  if acceso <> 1
     then
        hide fld "Ver"
     else
        show fld "Ver"
  end if
  put "p." into fld "P"
end.
```

Las repeticiones de comandos no sólo se expresan en las iteraciones, donde dicha repetición es inmediata. También existen muchos casos en que la repetición no es un flujo continuo sino dependiente de otras operaciones o condiciones que son muy variables. Es el caso, por ejemplo, de las instrucciones para leer o grabar datos en un disco.

Esta forma de redacción de programas puede ser obligatoria u optativa, según el lenguaje escogido. Se constituyen "paquetes" de instrucciones (llamados "sub-rutinas", "procedimientos" o "funciones"), los que se escriben una sola vez y son "llamados" (utilizados) las veces que se requiera.

Programación orientada a objetos (POO)

Estructura



Es un tipo de programación que usa "objetos" y sus interacciones para diseñar los programas. ¿Que es un "objeto" aquí? Es cualquier entidad (cosa o concepto) que tiene un determinado estado, comportamiento e identidad. El objeto se representa describiéndolo, definiendo la categoría ("clase") a la cual pertenece y eventuales subcategorías, las cuales "heredan" las características de la clase a la cual pertenecen. Esta estructura se asemeja a la de la memoria humana (gráfico al lado).

Ejemplo: Una "fuente de soda" << ES UN >> tipo de establecimiento comercial y un establecimiento comercial << ES UNA >> empresa. [Puse "El Pirata" como nombre propio de un caso particular.]

Los procedimientos asociados serían “vender” y “consumir” (lo cual debe ser definido en términos de programación, mediante reglas y procedimientos). Las variables necesarias para representar el estado de un objeto se declaran dentro de la clase que le corresponde y se llaman propiedades. Las funciones o acciones que pueden interactuar con dichas propiedades son "métodos de la clase" y se colocan dentro de la clase.

La POO permite escribir código en forma más ordenada y crear aplicaciones más fáciles de mantener. Aunque, para los programadores, puede ser fácil y ayuda a evitar importantes errores, es sin duda más compleja para la iniciación a la programación y no profundizaremos aquí.

4. Bases de programación

Con los lenguajes o con los generadores de aplicaciones construimos programas (*software*), que han pasado a llamarse más frecuentemente "aplicaciones" y "*apps*" en el caso de los *smartphones*.

> *"Una aplicación puede ser limitada o amplia, sencilla o compleja; y tanto en un caso como en el otro, está perfectamente controlada por la persona u organización que la haya diseñado. Podemos entenderlas como atajos que nos llevan directamente a donde queremos ir, sin necesidad de hacer búsquedas on line o, si somos de la vieja escuela, en nuestra propia memoria." (Gardner y Davies, "La generación app").*

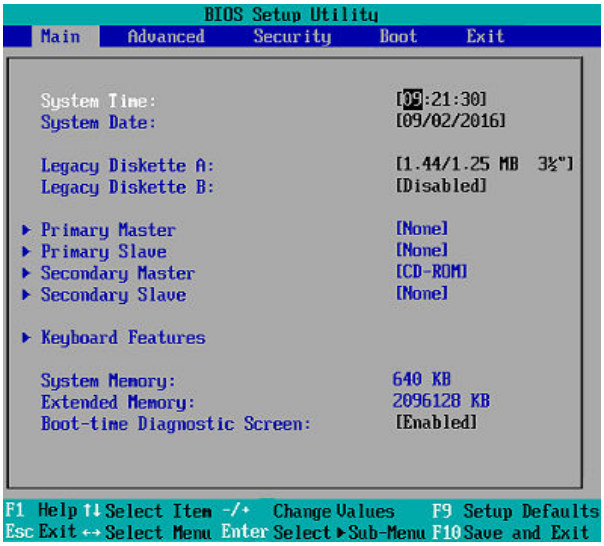
4.1. Sistema operativo

Estas aplicaciones, sin embargo, no pueden funcionar si no las "cargamos" primero en la memoria de computador. Si lo computadores tuviesen solamente su memoria ROM y tuviésemos que cargar cada vez todo lo necesario para poder trabajar, los programas tendrían muchas partes comunes, lo cual sería muy ineficiente. Por esta razón se instala en cada máquina el llamado "sistema operativo" (S.O.), cuyo objetivo es evitar estas duplicaciones y "tender un puente" activo entre el *hardware* y el *software*.

Así, el S.O. es un conjunto de instrucciones que cualquier programa requiere para poder operar, por lo cual era más razonable y económico separarlo (en vez de repetir las mismas instrucciones preliminares en cada programa). Responden tanto a un concepto de modularidad como de eficiencia ya que -al ser independientes- basta que se carguen automáticamente al encender el computador para luego poder ejecutar varios programas sin necesidad de volver a dar estas instrucciones.

El S.O. Linux está en unos 2000 millones de *smartphones* (bajo la forma del sistema Android) y motoriza la informática de colosos como Amazon y la Bolsa de Londres. Domina de forma absoluta en las 500 supercomputadoras más potentes del mundo y también en los servidores de la web. En los últimos años, Linux ha reemplazado UNIX y sus variantes comerciales, que dominaban completamente este mercado. Es una variante *Open Source* (es decir, de libre uso, con la posibilidad de ser modificado por sus usuarios) de los sistemas comerciales UNIX. Las razones de las preferencias son su modularidad (capacidad de personalización y versatilidad), su núcleo multidisciplinar (soporte genérico para todo tipo de tecnologías), su escalabilidad (capacidad de adaptarse a cargas elevadas y eficiencia) y el costo nulo de la licencia para usarlo (Xataka, 15/11/2017).

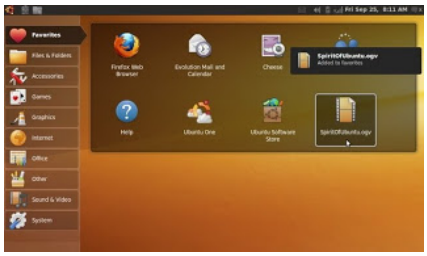
Todo S.O. mantiene una relación estrecha con la arquitectura del computador, por cuanto es el que debe asegurar que todos los componentes funcionen en forma armoniosa. Su primera función, por lo tanto, es entenderse con el "lenguaje de máquina" del procesador. Para esto, el S.O. se encarga de dirigir los comandos contenidos en los programas (aplicaciones) hacia las "puertas" del procesador central.



Pero, aquí es indispensable volver a tener en cuenta la arquitectura del computador: en efecto, el S.O. se encuentra en el disco duro y -por lo tanto- el computador debe "saber" que ha de leer primera cierta sección del disco duro cuando es encendido, para encontrar el S.O. y cargarlo en la memoria RAM, antes de poder hacer otra cosa. Para ello existen instrucciones pregrabadas en la ROM, que contienen lo que conocemos como "BIOS" o bien "UEFI", instrucciones básicas a las que sólo accedemos excepcionalmente (por ejemplo para elegir, si tenemos dos sistemas operativos en el disco duro, o bien para recuperar el sistema operativo en caso de que haya fallado; ejemplo al lado).

La BIOS o UEFI contiene la información que el computador requiere sobre su propia arquitectura, es decir, una definición de sus periféricos y de los accesos al disco duro. Esto viene predefinido y grabado por el fabricante. Guardan, además, algunos parámetros que son definidos cuando se instala el sistema operativo, por ejemplo, en qué orden debe buscar el S.O. en los sistemas de memoria (primero el disco duro, luego -si éste no funciona- en un lector de CD o una memoria USB, etc.).
También determina si se activan y son revisados los diferentes puertos que conectan los periféricos, para lo cual puede ser necesario que el S.O. haya sido complementado por controladores (*"drivers"*) de estos puertos.

Componentes del S.O.



1. Interfaz con el usuario:
La interfaz (o forma de interacción) puede ser de tipo intuitivo, como el escritorio, los íconos y las ventanas del Macintosh o de Windows (llamados por ello "entorno de escritorio"), o bien basarse en "lenguaje de comandos" como en el sistema Unix y otros, que requieren que se escriban órdenes (iguales a instrucciones de un lenguaje de alto nivel).

2. Administración de la CPU y la memoria principal:
Crea subdivisiones en la RAM, indicando dónde deben colocarse las instrucciones, los datos originales y los que generen los programas. En algunos casos, puede hacer que sólo una parte de un programa o de los datos sea "cargada" -y reemplazada en el momento oportuno-, para economizar espacio y trabajar con un conjunto de información mayor que el que cabe en la RAM (caso común de las bases de datos). (En este caso se habla del uso de "memoria virtual").

3. Administración de la memoria auxiliar:
Determina la longitud de los "paquetes" de datos que lee desde o envía a los dispositivos de memoria auxiliar, la forma y el lugar físico en que se guardan, la forma de acceder a ellos (construyen el "directorio" o índice de los discos, por ejemplo).

Para todas estas funciones, el sistema operativo reconoce ciertas instrucciones que forman parte del lenguaje de alto nivel (p.ej., "open", "read", "write", "save", etc.) y genera para cada una múltiples instrucciones en lenguaje de máquina de tal modo que el procesador efectúe todas las operaciones requeridas.

4. Interpretación del teclado:
Los sistemas operativos también determinan la forma en que se codifican las instrucciones y los datos que provienen del teclado. Todos los sistemas de origen americano hoy usan el ASCII (*American Standard Code for Information Interchange*), que tenía originalmente 128 caracteres (suficiente para el inglés). Pero esta cantidad impedía el uso de varios signos, entre ellos los acentos. Por ello se usa hoy el "Ascii extendido", que cuenta con 256 caracteres. Pero en la segunda serie de 128 caracteres, los diferentes sistemas operativos han recurrido a opciones diferentes y no son compatibles entre sí (razón por la cual los textos acentuados se ven con caracteres extraños cuando se pasa de un tipo de computador a otro, como del Macintosh al PC-Windows).

Es también la razón por la cual, en el correo electrónico, conviene algunas veces escribir sin acentos (usando el ASCII reducido), para asegurar un texto más legible, salvo que se pueda definir (en las "Opciones" o "Preferencias" de configuración) el uso del protocolo MIME, que traduce los caracteres a un formato común.

4.2. Base de los programas

Ante todo hay que tener en cuenta que el computador no puede pensar como el ser humano. Si queremos utilizarlo para resolver algún problema, debemos aprender primero a pensar de la manera adecuada. Si queremos sumar a mano y por escrito una columna de cifras, por ejemplo, hemos aprendido a sumar las unidades, escribir la cifra de la unidad, luego sumar las decenas, etc. Con una calculadora, podemos sumar la segunda cifra a la primera, luego la tercera al resultado, etc. Podemos usar el computador como la calculadora, pero no es lo que nos proponemos al hablar de programar.

Programar es decirle claramente cómo hacerlo en cualquier circunstancia. Para hacer una suma, debemos darle la lista completa de cifras a sumar, luego decirle que sume y que, finalmente, nos muestre el resultado. Como la calculadora, sumará la segunda cifra a la primera, luego la tercera al resultado, y así sucesivamente. ¿Pero cómo sabrá que llegó al final y que llegó el momento de mostrar el producto? Nosotros lo vemos, pero él no lo ve: está "ciego". Hay darle una pista para que "se dé cuenta".

¿Qué podemos hacer? Indicarle antes de empezar la cantidad de cifras (cuantas sumas hacer), o bien agregar una indicación al final de la lista, que pueda reconocer como tal (como la "raya para la suma", que usamos frecuentemente). Se prefiere generalmente el segundo método, porque nos evita contar nosotros primero, lo cual podría ser tedioso (y podemos fácilmente equivocarnos si la lista es larga) y no faltan los casos en que no sabemos de antemano cuantas cifras habrá que sumar. El lenguaje de programación nos puede proporcionar para ello un código especial. Pero esto tiene otra consecuencia: la máquina deberá "preguntarse" cada vez, después de agregar una cifra, si llegó al final de la lista. Veremos más adelante cómo representar esto en un programa.

Todo programa consiste en una serie de instrucciones (operaciones a realizar) de acuerdo a ciertas condiciones. Estas condiciones se establecen en términos lógicos.

Desde los griegos, la lógica considera solamente dos valores de verdad - "verdadero" y "falso"- aplicables a proposiciones completas, a conjuntos analizados de términos o a resultados de operaciones anteriores.

Esto es extraordinariamente congruente con el sistema binario ya que se puede establecer la equivalencia 0 = Falso y 1 = Verdadero. Con lo cual es posible "procesar" lógicamente afirmaciones con un procesador binario, transformando el razonamiento en cálculo. Así fue posible llevar la lógica tradicional a un sistema computacional. Boole lo hizo posible cuando desarrolló el sistema de álgebra binaria.

Aquí hay un pequeño ejemplo referido a la conjunción (operador "y", que se representa por "&"), teniendo en cuenta que la conjunción (&) de 2 proposiciones es verdadera sólo si las dos son verdaderas (=1).

Si x="llueve" e y="graniza" y si es verdad que "Llueve y graniza" (a la vez), tenemos: *(x=1) & (y=1) = x&y = 1.*

"No graniza" sería *~y* (pero es falso, por lo que *~y=0*) y "No llueve" sería *~x* (también falso). (La ~ quiere decir "no".)

Entonces: "Lueve y no graniza" (falso) será *(x=1) & ~(y=1) = x&~y = 0* [o también *(x=1) & (y=0) = 0*]

y "No llueve y no graniza" (falso) será *~(x=1) & ~(y=1) = ~x&~y=0* [o también *(x=0) & (y=0) = 0*].

Pero si "No graniza" es verdad, entonces "Llueve y no graniza" *x&~y=1* por cuanto lo verdadero sería que *y=0* (por lo que *~y=1*)

Problemas no lógicos

Para el computador, como máquina lógica, existen múltiples operaciones que no puede realizar, aunque en algunos casos es posible evitar la dificultad realizando una simulación. Un caso común, que ha entrado a formar parte de los lenguajes de programación, ha sido el de la generación de números aleatorios. Para el ordenador, en estricto rigor, es imposible producir números al azar, a pesar de lo cual podemos encontrar múltiples sitios web de casinos y loterías que los utilizan, lo cual ocurre también para seleccionar preguntas en las pruebas. ¿Cómo se logra?

La computación recurre a números pseudoaleatorios, generados por medio de una función (determinista, no aleatoria), que aparentan ser aleatorios. Se parte siempre de un valor inicial ("semilla") - la hay preprogramada - aplicando iterativamente la función generadora y sometiendo la serie a diversos tests que miden hasta qué punto se asemeja a una sucesión aleatoria. El principal defecto es que, si no se toma alguna medida para cambiar la semilla, cuando se lance el programa el primer número será siempre el mismo. Una forma de evitarlo es incluir un comando que ajuste la semilla en función de un número que será diferente cada vez, como transformar la fecha, con hora, minutos y segundos, en un número (que no podrá repetirse).
Así, para obtener una serie de $n números pseudoaleatorios, en PHP podemos usar la función rand (*random*, o sea aleatorio). Si $seg es el número semilla creado, se puede escribir

```
for ($i = 1; $i < $n; $i++) { $alea[$i]=rand(1,$seg);}
```

$i es una variable contadora, que parte de $1=1 y se le suma 1 (es lo que significa `$i++`) mientras su valor sea inferior a la cantidad deseada ($n). La serie estará en las $n instancias de la variable `$alea`.

4.3. Operaciones

Lectura – Escritura
Es la operación más simple, que consiste en colocar los datos recibidos en los acumuladores, enviar una copia del resultado a un lugar de almacenamiento o "vaciar" un acumulador (llenándolo de ceros).

Operaciones aritméticas
Las operaciones lógicas son operaciones que implican comparar uno a uno los bits de dos acumuladores y colocar un resultado en un tercero, de acuerdo a la regla asociada al operador lógico seleccionado ("y", "o", etc.).
La suma es la operación aritmética básica a la cual se reducen todas las otras operaciones matemáticas. Se efectúa solamente sobre dos operandos (Si se desea sumar más números, se suman los dos primeros, luego el resultado con el tercero y así sucesivamente). El procedimiento variará si se suman enteros sin signo, con signo o números reales (con decimales).
La resta se efectúa recurriendo a la suma y a un "truco" que consiste en descartar un bit del acumulador donde llega el resultado. La multiplicación se efectúa mediante sumas y corrimientos (desplazamiento de bits hacia la derecha o hacia la izquierda) y la división se efectúa mediante un proceso repetido de corrimientos y restas.
Así, la única operación matemática que se hace a la manera humana es la suma: de las otras se podría decir que son "sumas haciendo trampa".

Otras operaciones lógicas
Con la operación de comparación se determina si una cantidad es igual, mayor o menor que otra. (En algunos sistemas se efectúa restando una cantidad de otra y comparando el resultado con cero).
La operación de examen informa de ciertas condiciones críticas (como el "rebalse": cuando, fruto de una operación, un bit no puede ser almacenado porque excede la dimensión del acumulador).
La operación de edición es una combinación compleja de reemplazos y comparaciones de bits (especialmente utilizada por los procesadores de palabras).

Operaciones de la unidad de control
La Unidad de Control se encarga esencialmente de leer y ejecutar o hacer ejecutar las instrucciones que conforman el programa, mientras asegura que los datos fluyan entre la memoria RAM y la Unidad Aritmética.

Leer una instrucción implica:

- buscarla en la "dirección" en que se encuentra
- copiar la orden en el espacio de memoria reservado para ello;
- anotar la "dirección" de la siguiente instrucción (para volver al nº1 de este ciclo y encontrar la instrucción correcta).

Ejecutar: Un "decodificador" determina la secuencia de operaciones que debe realizarse para llevar a cabo la instrucción, activando los circuitos adecuados de la Unidad Aritmética y enviando los datos a sus acumuladores.

Corresponde también a la Unidad de Control evaluar las condiciones que determinan cuál instrucción debe venir a continuación en los casos en que la programación incluye instrucciones llamadas de "salto", es decir, que -según los resultados obtenidos- han de seguir con una serie u otra de instrucciones (volviendo

atrás o realizando otra serie de operaciones). Un ejemplo es lo que ocurre con una instrucción condicional con un "if" (si... entonces... o bien). Hablaremos más de ello en el tema de los algoritmos.

Conservar: La Unidad de Control también se encarga de colocar los programas y los datos en sectores determinados de la memoria RAM y de acceder a ellos cada vez que se requieran. Aunque los datos siempre han de colocarse en forma secuencial (uno tras otro), las formas de acceder a ellos son variables (por ejemplo como simple lista o en forma de árbol jerarquizado).

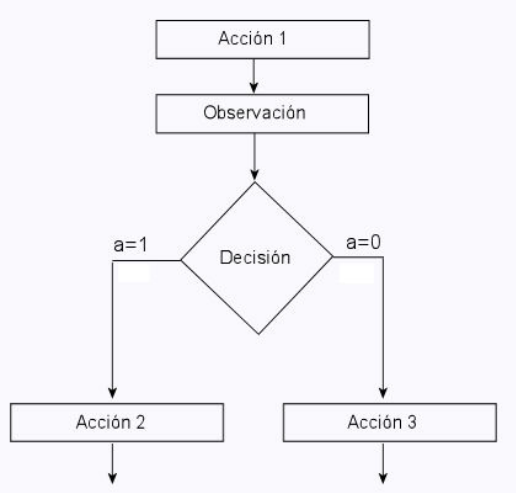
4.4. Construir un programa

No podemos pensar en programar alguna aplicación sin tener una clara visión de la secuencia de operaciones que deberá poder realizar y las condiciones asociadas a cada una, es decir, en la lógica de cada una.

Algoritmo y diagrama de flujo

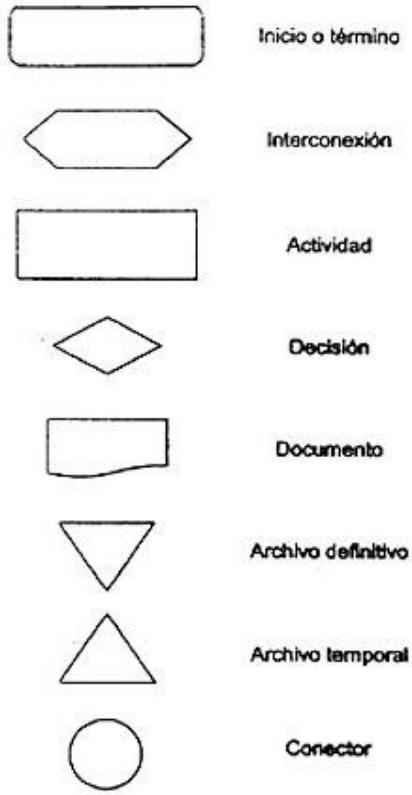
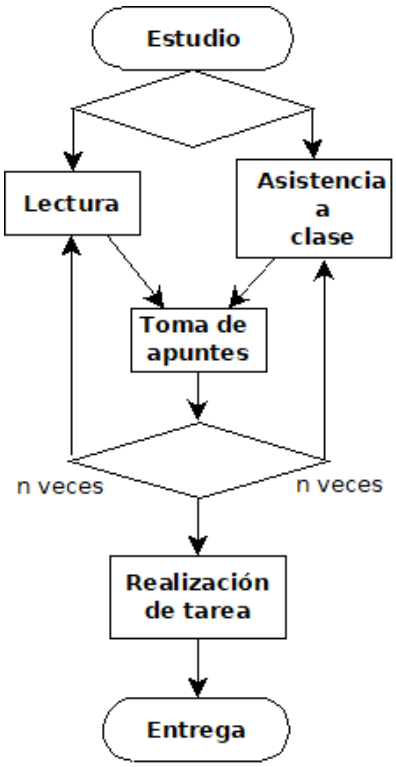
Toda acción que se realice con un computador implica operaciones de selección, partiendo con la selección de un programa ("aplicación") y luego – salvo muy raras excepciones – de una opción en un "menú". Lo anterior conlleva la introducción de condicionantes: "si se elige tal operación, entonces…; si no, entonces…" .

Podríamos inmediatamente utilizar un lenguaje de programación pero no es el método más aconsejable. En efecto, la misma secuencia de operaciones con sus condiciones puede ser escrita en distintos lenguajes pero su esencia permanece igual. Es preferible representar gráficamente esta secuencia, una forma que permite visualizar mucho mejor lo que se pretende hacer y lo que debería ocurrir. Esta forma es la de un diagrama de flujo o "flujograma".



Al lado se puede ver la forma básica del flujograma de todo algoritmo condicional. Es el modelo que debe ser reproducido para cada operación. Se ha convenido representar por un rectángulo las operaciones (debemos obviamente poner un nombre preciso y claro para cada una) y el rombo para representar las dos opciones lógicas: si a es verdadero (a=1) o si a es falso (a=0).

En la página siguiente se pueden ver, a la izquierda, algunos símbolos gráficos comunes para identificar distintas operaciones. Son útiles no sólo para graficar los algoritmos destinados a la programación, sino también para otros usos, como en el análisis de sistemas o la representación de cualquier secuencia de acciones, como en el ejemplo de la derecha en la página siguiente.

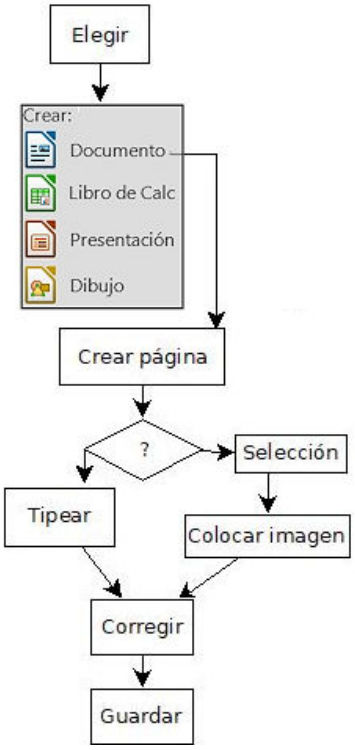
Símbolos comunes	Flujograma de actividad de estudio
	

Definición de propósitos

Obviamente, antes de construir el primer algoritmo, debemos tener claro cual es nuestro objetivo. Es indispensable describir bien el problema antes de pensar en un algoritmo para resolverlo. La selección del algoritmo adecuado (los procedimientos computacionales) es la parte más importante del trabajo y la más difícil, sobre todo si el problema es complejo. Por lo tanto, realizar un buen análisis, descomponiendo el problema tanto como sea posible en un conjunto de problemas menores es clave.

Y hay que recordar, como ya señalado, que la máquina no tiene la capacidad ni opera como nuestra mente: necesita generalmente un "camino" más largo para llegar a una solución, pero lo hará mucho más rápidamente. No podemos confiar por lo tanto en que un algoritmo útil para guiar un humano lo sea el computador. Debemos pensar a partir de la máquina y su forma de procesar.

Y debemos definir nuestro objetivo en términos de la utilidad de la aplicación para el usuario. ¿Le proveeremos datos consultables o le facilitaremos ingresar datos? ¿Será un servicio de compra, de intercambio de mensajes, de bienes? ¡Las posibilidades son infinitas! Debemos tener clara la manera en que el usuario podrá utilizar el programa. Por sobre todo, debemos preguntarnos cómo facilitarle lo más posible su uso.



Debemos, por lo tanto, preguntarnos qué es lo que podrá hacer el y cómo lo hará. Esto ha de llevarnos a anotar las órdenes que podrá dar, cuya lista constituye lo que llamamos habitualmente "menú". Por ejemplo: crear un documento nuevo, crear un área en este documento (página, celdilla, recuadro), admitir tipeo (o gráficos, para los cuales podrá haber un menú de formas prediseñadas), dar forma a lo ingresado, etc.. hasta finalizar con "guardar" lo hecho (en un determinado formato, generalmente electivo).

Así, después de formular un objetivo lo más claro posible, podemos pensar, en el "menú" de nuestra aplicación: una vez abierta la aplicación, el usuario podrá hacer la acción1, o la 2, o la 3… Cada una de estas opciones del menú pasará a encabezar un algoritmo, ya que será el punto de partida de una serie de acciones.

El paso siguiente consistirá en pensar, de forma más precisa, en las acciones a realizar, las cuales pasarán a ser los cuadros principales de los algoritmos.

4.5. Definiciones

Composición

Los programas se componen básicamente de comandos (órdenes), variables y constantes. Los comandos indican lo que debe hacerse (p.ej., *"print"* para imprimir o mostrar en la pantalla); las variables definen espacios de memoria cuyo contenido puede cambiar, y las constantes son espacios con datos invariables.
Los lenguajes de programación exigen, generalmente, que se defina al inicio del programa el tipo de datos aceptable para cada variable. Se le da un nombre y se indica el tipo de dato que le corresponde. Debemos considerar:

- cadenas alfanuméricas (letras y cifras)
- enteros (señalado como *"integer"* o *"int"*)
- de coma flotante (real, con decimales)
- lógico (de valor binario: vercdero o falso).

Palabras reservadas

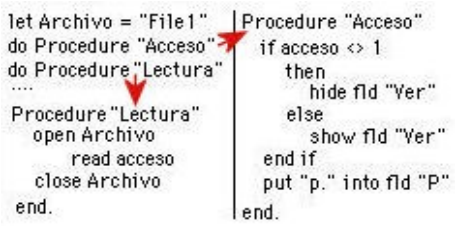
Podemos dar a las variables y constantes o parámetros (que tienen valores fijos) el nombre que deseemos MENOS las llamadas "palabras reservadas" que corresponden a los términos propios del lenguaje de programación. Los más frecuentes:

```
if then else elseif (si entonces o bien o bien si)
for while (para mientras [para operaciones reiteradas])
print (imprimir = mostrar)
query (obtener)  open(abrir)  close (cerrar)
```

Rutinas y sub-rutinas

Como ya señalado a propósito del concepto de programación estructurada, en los programas es habitual que haya series de órdenes (acciones) que pueden repetirse una y otra vez. Sería muy ineficiente repetir cada vez las líneas de programación que les corresponde, razón por las cuales se constituyen en bloques llamados "rutinas" o "procedimientos" (*"Procedure"*). Y dentro de una rutina, puede a su vez existir un bloque menor repetitivo, que será una "sub-rutima".

A cada rutina y sub-rutina se le da un nombre (un título) que permite que sea "llamada" en la parte de programa donde sea necesaria.



Si se ha de aplicar la rutina y seguir luego en el mismo lugar del programa, se escribe algo como `gosub(nombre rutina)` o bien `do Procedure(nombre rutina)`, como en el ejemplo de la izquierda. Si el desvío es absoluto (se continúa en otra parte), podemos decir `goto(nombre rutina)`. Estos términos también son "palabras reservadas".

Las rutinas y subrutinas se escriben generalmente al principio del programa por cuanto es más fácil encontrarlas ahí (podemos imaginar que, así, "ya son conocidas" cuando se llega a un comando que las llama).

Bucles

```
for (int i = 0; i<numEnt; i++)
{
   [operaciones a realizar]
}
```

Se puede hacer que se repita un comando o conjunto de comando hasta que se cumpla cierta condición. En este caso hablamos de "bucle" y no de rutina. Puede ver un ejemplo típico a continuación del lado izquierdo.

El bucle está definido por *for{ }*. (Los corchetes de inicio y de término son generalmente indispensables.)

Entre paréntesis tenemos la condición que, en este caso, es un conteo que indica cuantas veces se debe repetir: *i++* es la forma convencional para indicar que se suma 1 a i en cada "vuelta" y el límite está establecido por *i<numEnt* , donde *numEnt* debe haber sido definido con anterioridad y es la cantidad de repeticiones posibles.

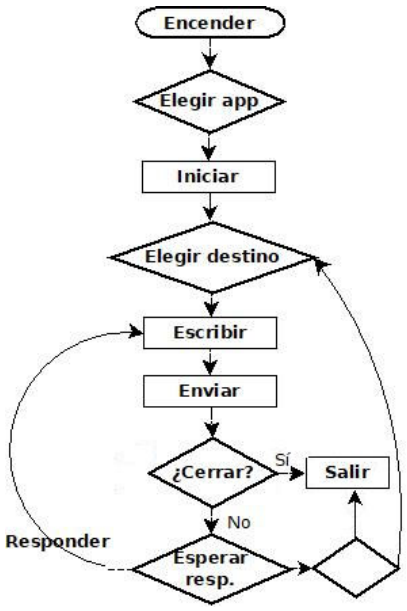
Sangría

```
for ($j = 0; $j < nrows; $j++) {
  $fechabus=data[2];
  if ($fechabus) {
    if ($fechabus==$fechant) {
      $sumfech[$f]=$sumfech[$f]+1;
      } else {
         $f++;
         $sumfech[$f]=1;
         $fech[$f]=$fechabus;
         $fechant=$fechabus;
      }
     $totfechas++;
  }
```

Escribir todo un programa con el mismo margen izquierdo termina siendo confuso y dificulta ver con claridad las distintas partes del mismo. Por esta razón, se acostumbra indentar los distintos segmentos y separarlos claramente. Las aplicaciones para escribir código (como Notepad++), además, marcan con diferentes colores los distintos componentes lo que facilita aún más la supervisión. Ejemplo al lado.

4.6. Entrenamiento

Observación



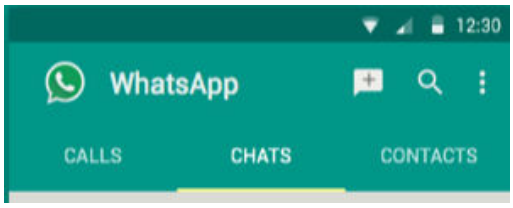
Para entrenarnos, recomendamos tomar nuestro teléfono celular, elegir una aplicación y anotar una por una todas las acciones que realizamos con él. Podríamos obtener:

1. Encender
2. Elegir la app (tocar el icono) [Supongamos que sea de mensajería]
3. Tocar "Iniciar un chat"
4. Elegir (tocar) el "contacto" (destinatario)
5. Escribir el mensaje
6. Enviar

Ahora se nos presenta una decisión:

7. ¿Terminamos?
    1. Cerrar y salir
    2. ¿Esperar una respuesta? (Incluye una decisión de tiempo de espera)
        1. Responder (implica Escribir…: se vuelve arriba)
        2. No esperar más implica una nueva decisión:
            1. Volver a escribir a otro o
            2. Cerrar

Nos queda una tarea importante: revisar la interfaz gráfica de usuario y tomar nota de sus componentes. ¡Cuando planifiquemos una nueva aplicación, deberemos incluir el diseño de este tipo de componentes!



Usando el ejemplo de WhatsApp, veríamos que tenemos un menú de tres palabras (*"Calls", "Chats"* y *"Contacts"*) debajo de tres pequeños iconos: un signo "+", una lupa y tres puntos verticales (que despliegan otro menú). Debemos anotar estos elementos y agregar su significación (la función que cumplen), ya que constituyen el menú de la aplicación y determinan múltiples operaciones.

Revisemos cada opción disponible y anotemos que es lo que permite hacer cada una. Luego tratemos de dibujar un flujograma para cada una. (En PC, recomiendo la aplicación "Dia"[13], de fácil uso, que trae las figuras geométricas adecuadas y que he usado aquí.)

Finalmente, revisemos nuestro flujograma: sin duda habrá sido la guía de quien programó la aplicación. Como ejercicio complementario, podríamos intentar transformarlo en algoritmo usando seudo-código.

13 Disponible en http://live.gnome.org/Dia

Crear una página web

No es necesario saber HTML para crear una página web. Existen varios "generadores" que permiten hacerlo fácilmente. Uno de los más utilizados, que ofrece además mantener el sitio en su servidor, es WordPress.

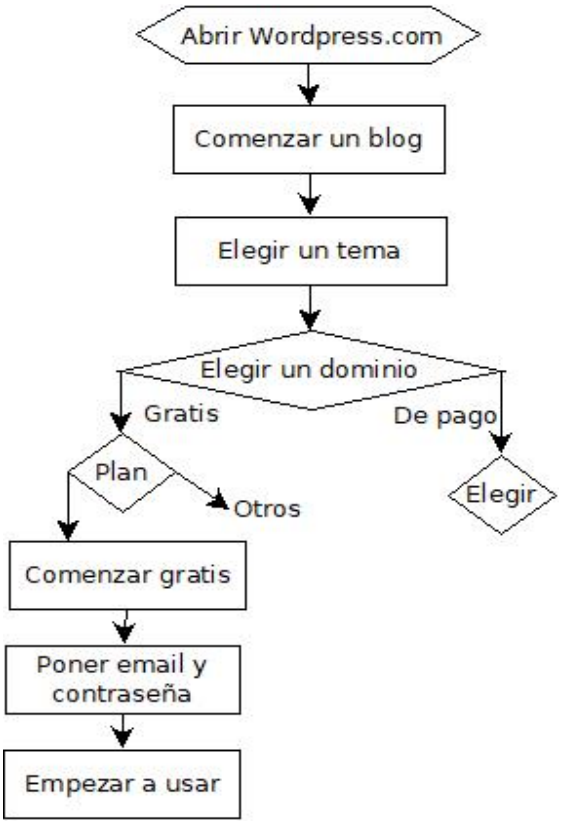
El software de WordPress puede ser descargado gratis para ser instalado en un servidor propio (o espacio arrendado en un servidor comercial) pero también se ofrece la posibilidad de crear un sitio en forma gratuita o pagada en el servidor de la empresa. Los pasos a seguir se indican en el flujograma que sigue.

Los pasos a seguir, después de abrir el sitio web de WordPress son muy claramente señalados:

"Comenzar un blog", luego "Elegir un tema": el "tema" es lo que define la gráfica y la estructura de las páginas. Hay varios temas de uso gratuito y otros que son de pago (generalmente más complejos).

Después se debe elegir un "dominio", que será el nombre de nuestro sitio. El dominio gratuito tendrá una dirección del tipo https://nombre.wordpress.com

A la derecha puede ver el menú que se abrirá después de poner su email y contraseña. Conviene ir al submenú de "Apariencia" para afinar los detalles.

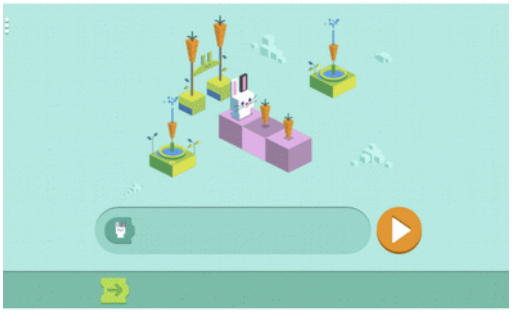


Luego podemos crear las "páginas" y las "entradas". El mismo WordPress explica que *"A diferencia de las entradas, que se muestran en la página inicial de tu blog en el orden en que se publicaron, las páginas se ajustan mejor a un contenido atemporal que quiere ser más accesible, como tus datos de contacto o la información acerca de ti."*

Aprender jugando

"Coding for Carrots"
("Codificando para zanahorias")

Como *"doodle"* del día 4 de diciembre 2017, Google creó un pequeño juego destinado a enseñar a programar, llamado



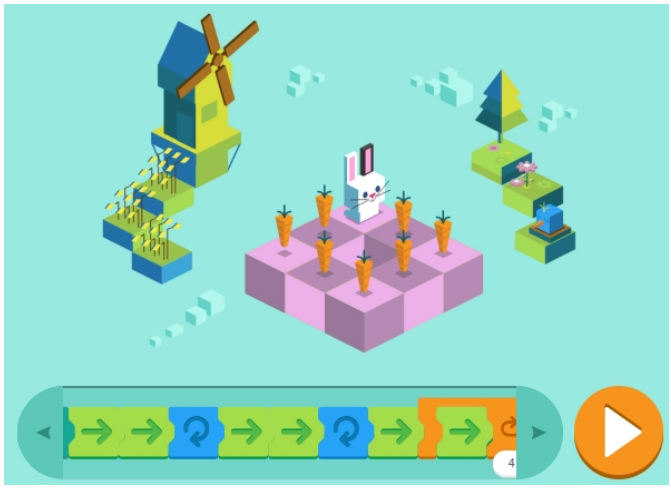
"*Coding for Carrots*"[14], basado en el lenguaje Scratch[15] para niños, desarrollado en el MIT. Se trata de hacer avanzar un conejo para que pueda comer zanahorias y, para hacer que se mueva, hay que introducir iconos de una lista de opciones.

> Google quiso de este modo celebrar el 50° aniversario de la introducción de los lenguajes de programación en la educación infantil: en 1960, el matemático Seymour Popert creó Logo, un sistema de programación adaptado a los más pequeños y que ayudó a que los ordenadores conquistasen las aulas.

Los diseñadores de Google entendieron perfectamente cual es la esencia de este tipo de enseñanza: descubrir y describir las diversas acciones a realizar y ordenar su realización, en este caso, con una secuencia de íconos. Apuntaron a la lógica del proceso y el juego permite ensayo y error, sin apelar a ningún lenguaje de programación.

A continuación se pueden ver algunos pantallazos que muestran el sistema:

14 Se puede ver completo en https://www.google.com/doodles/celebrating-50-years-of-kids-coding
15 http://scratch.mit.edu/ y http://www.programacionscratch.com/

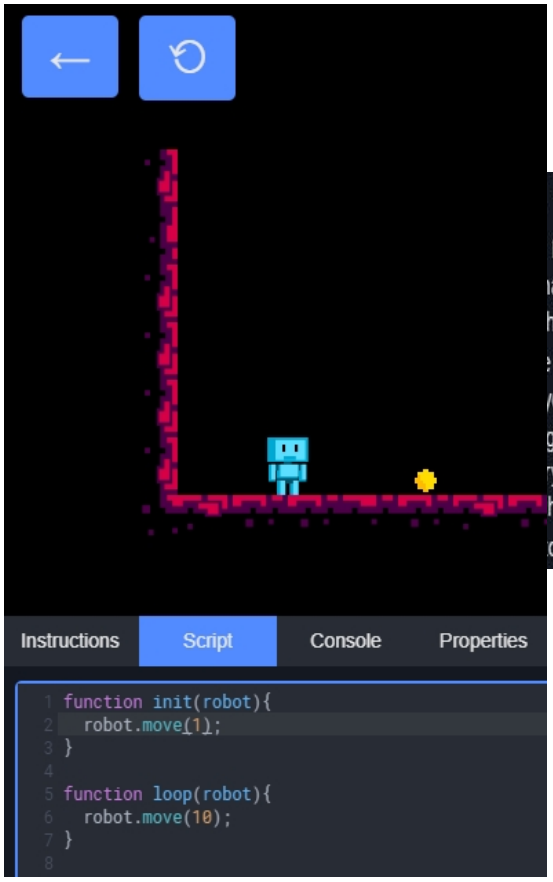


"JSRobot"

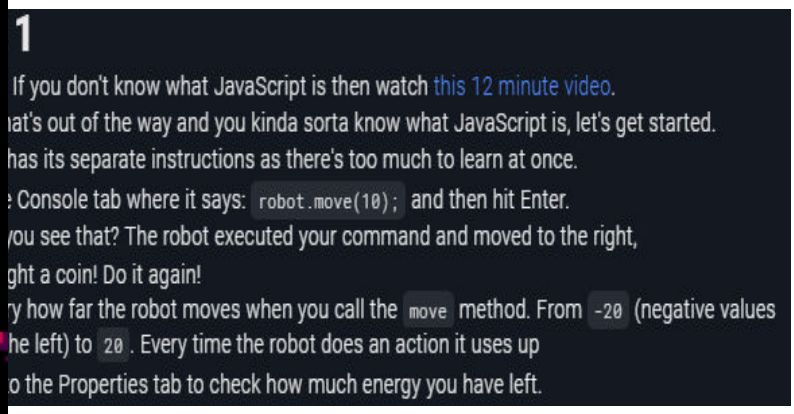
Si, a pesar de nuestra opinión de que es suficiente el aprendizaje con los flujogramas y algoritmos en seudo-código, quiere aprender algún lenguaje, el más empleado del mundo es sin duda el JavaScript, cuyo código se inserta en el de las páginas web. Y un recurso útil para aprenderlo es "JSRobot"[16], un juego (con instrucciones en inglés) en el que se debe guiar a un robot a través de varios niveles de dificultad creciente utilizando únicamente la "consola" (espacio para líneas de comando) y dándole órdenes con el lenguaje de programación[17]. Vea pantallazos a continuación.

16 En https://lab.reaal.me/jsrobot/

17 Hay un video introductorio de 12 minutos, en inglés (se pueden activar subtítulos en el mismo idioma), en https://www.youtube.com/watch?v=Ukg_U3CnJWI



En *"Instructions"* se indica qué comando escribir, primero en la *"Console"* para ver su efecto y luego en el *"Script"* (que es el programa que se va completando). Aquí las instrucciones del Nivel 1:



A la izquierda, la vista con el *script*, donde se ordena un movimiento inicial de 1 paso [`robot.move(1)`] y luego la función de repetición [`function loop()`] de 10 pasos.

Incluimos una proposición de ejercicios a realizar con algoritmos y seudo-código en el Anexo 2 de la página 77.

Si de todos modos quiere aprender algún lenguaje de programación, puede consultar Progate, una plataforma actualmente de uso gratuito que ofrece cursos de Java, PHP, HTML, CSS, JavaScript, jQuery, Ruby y Python, o la plataforma Codecademy.

4.7. Bases de datos

Una "base de datos" es un conjunto de datos (informaciones) con una estructura lógico-matemática, gestionado por un tipo de software especial llamado "Sistema de Gestión de Bases de Datos" (SGBD). Todo SGBD debe cumplir como mínimo las siguientes condiciones:

- Las estructuras de datos (espacio informativo) son simples e independientes del programa que genera los datos, lo cual se logra cuando son constituidas en forma de tabla de 2 dimensiones cuyos componentes (valores) son unidades de dato (no divisibles).
- Varios espacios informativos -si los hay- se asocian mediante la presencia de, al menos, un atributo (característica) común.
- Un conjunto de operadores permite la definición, búsqueda y actualización de los datos.

- Un conjunto de requisitos de integridad define el estado coherente de la base de datos. (Esto dice relación con las "llaves" que remiten de un archivo a otro, si hay varios, y la identidad de los valores correspondientes a los mismos atributos).

Aclaremos algunos conceptos y veamos cómo se manejan.

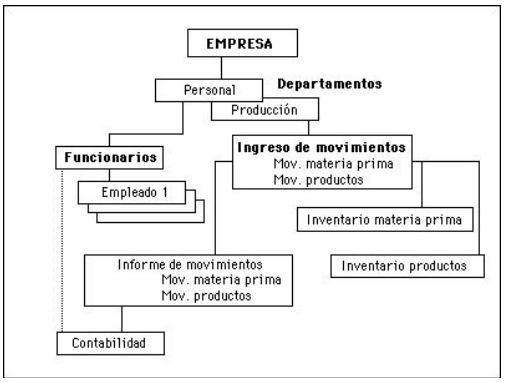
Referente y atributos

Una base de datos puede ser un archivo único, como podría ser el catálogo de una biblioteca. Su estructura depende de las características de los objetos que representa, que llamamos "referentes" (los libros en el caso de una biblioteca). Llamamos "atributos" las características útiles para describir los referentes, por ejemplo, el autor, el título, la editorial, el lugar y la fecha de edición de cada libro. Acabamos de mencionar 5 atributos y, para cada uno, habrá una "unidad de dato". Podemos conservar estos datos registrándolos en el formato que ofrece una hoja de cálculo (aunque no la usemos para funciones de cálculo) porque nos ofrece una estructura de renglones (filas) y columnas (ver ejemplo debajo). A cada referente corresponde una fila, cuyos datos conforman un registro.

Autor	Título	Editorial	Lugar	Fecha
ELLIOTT, D. & R.	El control popular de la tecnología	G.Gili	Barcelona	1980
ESTRELLA, J.	Viejas y nuevas fronteras de la cie	Universitaria	Santiago	
FALLACI, O.	La rabia y el orgullo	El Ateneo	Buenos Aires	2005
FAURE, E. & col.	Aprender a ser	Alianza Unive	Madrid	1975
FITZMYER, J.	Catecismo cristológico	Sígueme	Salamanca	1988
FLAMENT, C.	Redes de comunicación y estructu	Nueva Visión	Buenos Aires	1977

Todas las bases de datos se componen de una o varias tablas con alguna relación entre ellas. Según el tipo de relaciones entre las tablas que las componen, las bases de datos se clasifican en jerárquicas, relacionales u orientadas a objetos.

BD jerárquica



Un ejemplo de BD jerárquica es el que sigue la estructura de una industria: supondremos que ésta se divide en departamentos, cada uno de los cuales desarrolla ciertas funciones. Se ha de considerar información acerca del personal de la empresa y también de las operaciones realizadas, que son documentadas de diversas maneras. Todo ello da origen a un complejo sistema de manejo de información. El ejemplo adjunto presenta un sistema jerárquico simplificado de archivos destinados a documentar las señaladas operaciones y sus consecuencias sobre las existencias de materias primas y productos terminados, y la situación contable de la empresa.

La principal ventaja de esta estructura consiste en que la información que aparece en un registro jerárquicamente superior se aplica igualmente a todos los registros que le son subordinados. Su desventaja es que el arbol jerárquico debe definirse previamente, es difícil de modificar para insertar nuevos registros y es difícil responder con rapidez una consulta sobre un punto específico de un registro subordinado. Por esta razón se usa muy poco este modelo en sistemas digitales.

BD relacional

La estructura relacional de un conjunto de archivos es la que considera y aprovecha las operaciones de la álgebra de conjuntos y la idea de fondo es que todos los archivos relacionados entre sí pueden ser considerados como formando un solo espacio informativo, en el cual pueden efectuarse múltiples operaciones de selección y de reordenamiento sin perder las relaciones entre datos definidas al confeccionar los registros.

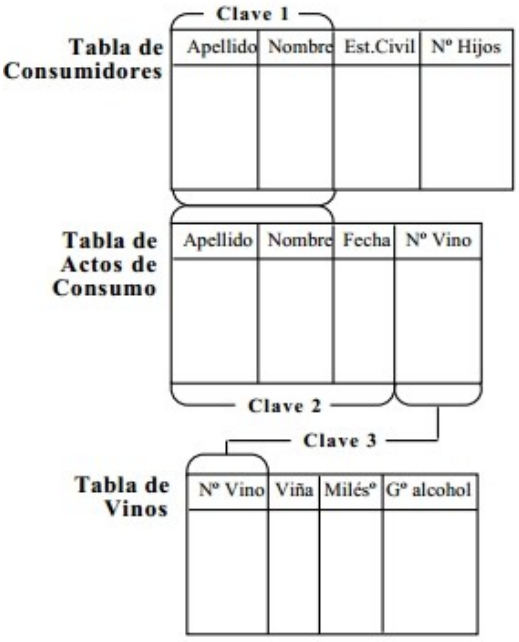
La estructura relacional permite estas operaciones lógicas (unión, intersección, etc.) y otras como el reordenamiento, no estando condicionada por la forma física de los archivos, los cuales muy ventajoso por cuanto permite el uso de una gran variedad de programas para acceder a la información. Sus ventajas están, evidentemente, condicionadas por múltiples reglas que no detallaremos aquí por ser una materia más técnica.

Un sistema de gestión de base de datos (SGBD) relacional exige como mínimo que:

- 1. Toda información de la Base de Datos sea representada por valores en tablas.
- 2. No habrá punteros (direcciones codificadas) visibles para el usuario de tales tablas.
- 3. El sistema debe poder utilizar operadores de restricción, proyección y unión natural sin limitaciones dependientes de condiciones internas ("Operadores relacionales").

Si cumple con otras dos condiciones, podrá llegar a ser "completamente relacional":

- 4. Reconoce y utiliza todos los operadores del álgebra relacional.
- 5. Cumple los requisos de integridad por unicidad de clave y de constricción referencial (limitaciones de libertad).



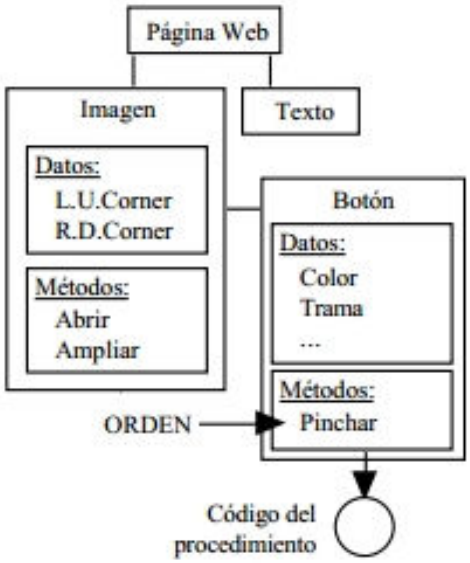
Las constricciones o exigencias básicas de los sistemas relacionales son tres, relacionadas con la "clave":

- 1. Unicidad de clave: Como un conjunto no puede tener dos veces un mismo elemento, no puede existir dos veces el mismo registro en un archivo. Se llama "clave" el conjunto mínimo de atributos cuyos valores permite identificar un registro (fila de datos) único, y también -indirectamente- un referente único.
- 2. Constricción de referente: Ya que debe haber un referente para todo dato ingresado, se prohibe dejar en blanco (sin información o valor nulo) los campos (celdillas) en que debe ser registrada una clave.
- 3. Integridad de referencia: A partir de la relación biunívoca que ha de existir entre un referente y un registro (fila de datos), para ser fiel al principio general de relación se debe asegurar que la clave de un archivo esté presente en cualquier otro archivo con el cual se relacione. En otras palabras, dos archivos se relacionan correctamente cuando remiten a los mismos referentes, individualizados de la misma manera. (Esto no quiere decir que, si pasamos de un 2º a un 3º archivo, los referentes sigan siendo los mismos que para el 1º y el 2º: podrán ser otros, pero descritos con igual clave en el 2º y 3º archivo).

Ver gráfico adjunto.

BD orientada a objetos

Las bases de datos orientadas a objetos (BDOO), a la vez que combinan aspectos propios de los sistemas jerárquicos y de las BD relacionales, introducen importantes diferencias. Un objeto es, aquí, no un mero referente que se describe, sino algo que se describe y, además, incluye procedimientos que desencadenan acciones cuando el objeto es referenciado. Incluirán varios modelos de registros y sólo habrá ocasionalmente alguna relación entre ellos (cuando compartan algún atributo). Se prestan muy bien para construir sistemas de hiperarchivos (hipertextos) y existen "servidores universales" que permiten manejar este tipo de archivos en coordinación con servicios de web.



Un ejemplo muy claro es un "botón" en una página web, que abre una ilustración o produce la navegación hacia otra página cuando es pinchado. Pero el botón específico es un objeto que pertenece a (y sólo aparece en) una página web y comparte características con otros tipos de ilustraciones. Así, el botón pertene a la clase de las imágenes y estas, junto con los textos, a la clase llamada "página web". Así, descubrimos un conjunto de objetos de misma jerarquía que pertenecen a otros objetos de mayor jerarquía. Al igual que en las BD jerárquicas opera el principio de herencia, mediante el cual las características del "padre" se transmiten al "hijo". Pero aquí, como ya señalado, los objetos son generalmente "activos", es decir, asociados a determinadas acciones, como el pinchar el botón ha de desencadenar, por ejemplo, la apertura de otra página. Las acciones asociadas a un objeto son llamadas "métodos". Se activan mediante una orden y desencadenan un procedimiento (Ver Gráfico).

Otras BD

Existen otros tipos de bases de datos. Entre los modelos más significativos podemos mencionar a las Bases de Datos Relacionales Orientadas a Objetos (*"Object Oriented Relational Data Bases"*), que combinan las características de las dos categorías antes citadas; las Bases de Datos Activas o BD con reglas activas (reglas ECA: *"Event - Condition - Action"*) y las Bases de Datos Inteligentes, que son sistemas que incluyen "bases de reglas" y de un "motor de inferencia" (rutina que permite hacer deducciones y proyecciones a partir de los datos.

Manejar una tabla

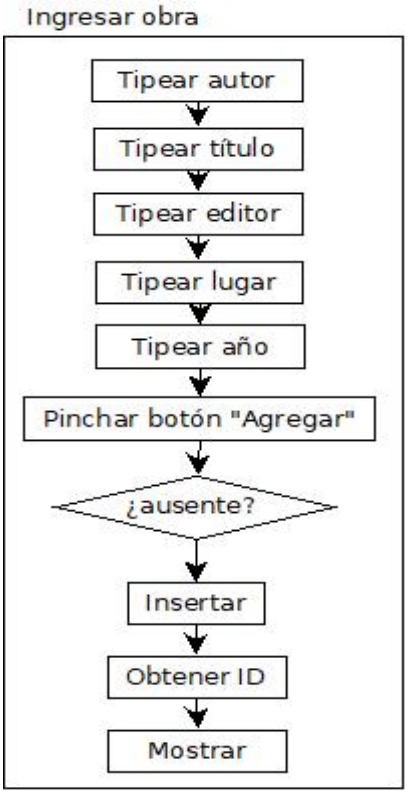
¿Cómo se maneja una base de datos? Para completar este capítulo sobre programación, veremos cómo se maneja una BD de una sola tabla (como nuestro ejemplo de catálogo de biblioteca), que es la forma más sencilla. Consideraremos el ingreso de datos de un referente y luego cómo programar una búsqueda (en una forma simplificada). En verde van los comentarios.

```
<!-- AGREGAR [1° formulario de ingreso]
rows indica el número de líneas para el texto en el formulario
y cols indica su número de caracteres (ancho del espacio) -->
     Ingreso de obra
     Autor <Textarea NAME="autor" rows="1" cols="70"></Textarea>
     Título <Textarea NAME="titulo" rows="2" cols="70"></Textarea>
     Editor <Textarea NAME="editor" rows="1" cols="70"></Textarea>
```

```
Lugar <Textarea NAME="lugar" rows="1" cols="70"></Textarea>
Fecha<INPUT TYPE="text" NAME="fecha" SIZE=4 VALUE="$an">
<input type="submit" value=" AGREGAR "> // (botón)

<!-- La fecha tiene otro formato porque se capturará un valor
numérico de 4 cifras en formato "text"-->
```

Agregar




```
<!-- AGREGAR [1° formulario de ingreso]
     rows indica el número de líneas para el
     texto en el formulario
     y cols indica su número de caracteres
     (ancho del espacio) -->
     Ingreso de obra
     Autor <Textarea NAME="autor" rows="1"
     cols="70"></Textarea>
     Título <Textarea NAME="titulo" rows="2"
     cols="70"></Textarea>
     Editor <Textarea NAME="editor" rows="1"
     cols="70"></Textarea>
     Lugar <Textarea NAME="lugar" rows="1"
     cols="70"></Textarea>
     Fecha<INPUT TYPE="text" NAME="fecha"
     SIZE=4 VALUE="$an">
     <input type="submit" value=" AGREGAR ">
     // (botón)
<!-- La fecha tiene otro formato porque se capturará un valor
numérico de 4 cifras en el área "text"-->

<!-- AGREGAR [2° fase: insertar en la tabla]
     Se debe verificar primero que los datos
     nuevos no estén ya ingresados (no se
     incluye esta rutina aquí), luego se hace
     la inserción: -->
     $sql->Insert("insert into catalogo values
     (0,'$autor','$titulo','$editor','$lugar',
     '$fecha');");
     $sql->QueryRow("select * from catalogo
     where titulo='$titulo'");
      $idno=$sql->data[0];
          echo "Registro creado con ID=
     $idno".
```

"echo" significa mostrar en pantalla. La serie de datos que se insertan empieza por 0 porque este espacio será utilizado automáticamente para dar un número de orden al registro, el que se obtiene luego buscando el registro recién ingresado por comparación del título (sql→QueryRow…) y queda en la variable $idno que se muestra en el "echo". "SQL" es el lenguaje estándar de manejo de bases de datos (*Structured Query Language*).

Para buscar debemos prever un formulario donde seleccionar primero el atributo e ingresar luego el término a buscar. Supondremos que buscamos una palabra (o varias, pero juntas) en el título y la pondremos en la variable $paltit. (Es parecido a lo que mostramos para agregar un registro.)

```
<!-- BUSCAR -->
     $sql->Query("select id, autor, titulo, editor, lugar fecha from
     catalogo where titulo like '$paltit' ");
         $nro=$sql->rows;
         if (!$nro) { echo "NO SE ENCONTRO NADA."; }
         $last = "";
         for ($i = 0; $i < $sql->rows; $i++) {
            $sql->Fetch($i);
            $id = $sql→data[0];
             $autor = $sql→data[1];
             $titulo = $sql→data[2];
             $editor = $sql→data[3];
             $lugar = $sql→data[4];
             $fecha = $sql→data[5];
             if $last != $id { // evita una repetición (!= significa
             "diferente")
                echo "$id, $autor, $titulo, $editor, $lugar, $fecha <br>";
                // <br> significa pasar luego a otra línea en pantalla
                $last = $id;
             }
          }
```

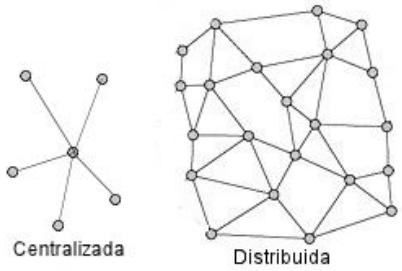
Como se puede ver, verificamos primero la existencia de un registro que contenga la(s) palabra(s) buscada(s). Si existe alguno, se toma nota y luego se muestra. La rutina se repite mientras haya registros (controlado por el avance de $i y su comparación con la cantidad de registros ($sql->rows), anotando el identificador del último registro leído ($id) en la variable $last.

En el Anexo 1 se adjunta una descripción de como concebir y constituir una base de datos periodística, para quienes estudian carreras del área de las comunicaciones sociales.

5. Complementos

5.1. Redes

LAN – WAN



Las redes pueden ser centralizadas (todos pasan por un mismo "nodo") o descentralizadas (también se dice "distribuidas"), en cuyo caso las diferentes máquinas pueden acceder unas a otras.

Existen redes locales o "LAN" (*Local Area Network*) y redes de amplio alcance o "WAN" (*Wide Area Network*). Pueden ser tanto alámbricas como inalámbricas. Una red WiFi casera es LAN inalámbrica. Las redes WiFi[18] públicas son redes WAN inalámbricas en el tramo entre el punto de acceso y los receptores (como los *smartphones*) pero se conectan "del otro lado" con internet de una forma que puede ser a su vez inalámbrica o por cable.

Existen aplicaciones que permiten que los *smartphones* se comuniquen entre sí sin utilizar ni la telefonía ni internet, lo cual permite formar también una red distribuida y se está proponiendo como alternativa a la internet clásica. Es posible, además, usar un celulares para ofrecer conexión WiFi a otros dispositivos, lo que se conoce como *tethering*; cuya función es hacer del teléfono una especie de *router* para compartir datos.

Las comunicaciones internacionales se hacen por redes WAN, sea por cable (fibra óptica) sea por satélite. El 99% de las comunicaciones transcontinentales se realizan por fibra, consiguiendo una velocidad hasta ocho veces superior a la de la red de satélites.

Internet

Internet es una red amplia y distribuida. Es la red mundial que permite a los computadores conectarse entre sí, sobre la base del código binario. La primera conexión de este tipo tuvo lugar en 1969 en los Estados Unidos y la Agencia de Proyectos de Investigación Avanzada de Defensa (DARPA) de ese país organizó la primera red, principalmente para enlazar las universidades, que empezó a funcionar en 1981 (fue BITNET), que pronto se abrió a otros países. Fue en 1983 que quedó definido el modelo universal de conexión (llamado "protocolo TCP/IP") que sirve desde entonces.

> *"La mayoría de nosotros usamos Internet a diario sin darnos cuenta de la enorme cantidad de sistemas, redes y servidores que operan en la sombra para acercarnos los diferentes servicios online directamente hasta nuestras pantallas."* (Xataka, 9/03/2016)

Una característica esencial de internet es que es una red de redes que utilizan todos los tipos de canales disponibles en los sistemas de telecomunicaciones. Es una red topológica, es decir, que puede cambiar de forma a cada instante, alargando o acortando un "camino" para transmitir los datos. Y está diseñada para encontrar siempre la ruta más eficiente. Es desde sus inicios descentralizada, formada por un enorme número de "nodos", que son los computadores que aseguran las transmisiones.

18 Wi-Fi es una marca de la organización Alianza Wi-Fi, que es la que define los estándares de comunicación inalámbrica y certifica los equipos que cumplen la norma. Son comunicaciones inalámbricas de corto alcance.

Cada computador en la red pública de internet tiene una dirección numérica única, llamada "dirección IP", que consiste en una cadena de números que resulta difícil recordar. Por este motivo se creó el Sistema de Nombres de dominio o DNS, que reemplaza estos números por un "nombre de dominio", y se ha creado una autoridad que vigila la atribución (y unicidad) de estos nombres, la IANA (Autoridad de Números Asignados en Internet).

Los números son los que permiten realmente el establecimiento de las conexiones entre los computadores, razón por la cual es necesario que haya máquinas que se encarguen de hacer la "traducción": son los "servidores de nombre", que se encuentran en los nodos (computadores dedicados a establecer las conexiones y enviar los paquetes de datos).

Los nombres de dominio se utilizan tanto para enviar correo electrónico, enviar archivos de computador a computador y acceder a sitios web.

La autoridad superior del ecosistema global de Internet es la ICANN[19] (Corporación Internet para Números y Nombres Asignados), una corporación de beneficio público, sin fines de lucro, con participantes de todo el mundo.

ICANN administra los protocolos y estándares de internet, lo cual implica el mantenimiento de muchos códigos y números utilizados. Los protocolos son conjuntos de tipos, parámetros, variables y valores de comunicaciones acordados. La estandarización de los protocolos de Internet es esencial para garantizar que Internet siga funcionando y permite a aquellos usuarios que utilizan equipo o software de diferentes proveedores comunicarse de forma efectiva.

Sería un grave error confundir internet con los contenidos de la Web o de las apps y sus usos y más aún considerar la estructura actual de internet como un modelo definitivo, aplicable a todas las áreas. Muchas de las apps de nuestros teléfonos celulares utilizan internet, aunque no la web.

La red oscura

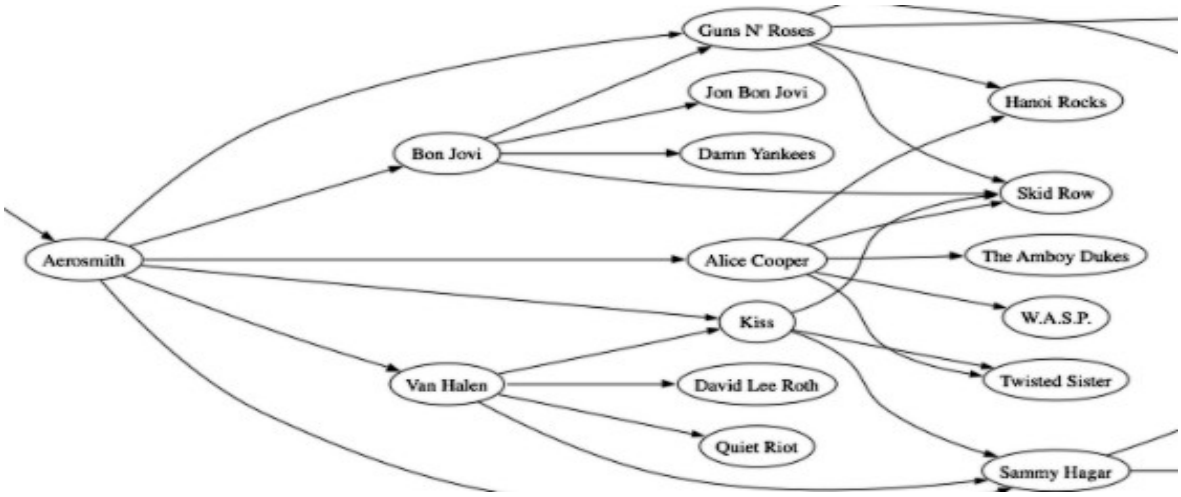
Gran parte de internet no es directamente accesible por cualquiera: es la "web profunda" o *"deep web"* que contiene, por ejemplo, datos de empresas o archivos de particulares (como las "nubes"). En lo profundo de esta gran red se esconde una gran "red oscura", principal herramienta de delincuentes de todo tipo. Venta de drogas o armas y asesinos que se ofrecen para ser contratados son algunos de los temas estrella de esta zona de la red donde florecen sitios anónimos y tecnologías de encriptación de datos, a los que recurren también cada vez más los pederastas. En 2013, el Centro Nacional para Niños Desaparecidos y Explotados de Estados Unidos (US National Center for Missing and Exploited Children), encontró más de 23 millones de imágenes obscenas de niños en internet y a mediados de 2014 había detectado 112 millones de carpetas que contenían imágenes de abuso sexual de menores (BBC Mundo, 19/06/2014). Además de pornografía infantil, la *"deep web"* aloja un supermercado de corrupción, drogas y armas, que se venden como si se tratara de una tienda de zapatillas. Los delincuentes no cejan en sus intentos por evadir la ley y la *"dark net"* es sólo uno de los lugares donde el anonimato saca lo peor del ser humano (ABC.es, 6/04/2014).

19 https://www.icann.org/es

Redes sociales

Uno de los factores que incrementó enormemente el uso de internet ha sido la aparición de las aplicaciones destinadas a facilitar los contactos sociales, especialmente la de Facebook. Ha ampliado, gracias a internet, lo que siempre han sido las redes que formamos en el contexto de la sociedad. Ya el escritor húngaro Frigyes Karinthy había sugerido en 1930 que no requeriríamos contactar con más de seis personas para encontrar a alguien siguiendo sus redes de amigos y conocidos, como se muestra en el gráfico siguiente. Esto fue verificado en la década de 1960 por Stanley Milgram, psicólogo profesor de Harvard, que recurrió para ello al envío de cartas. En 2013, Eman Yasser Daraghmi y Shyan-Ming Yuan,de la Universidad Chiao Tung de Taiwan, mostraron que en las redes sociales sólo son 3,9 los grados que separan a dos personas cualquiera en el mundo (FayerWayer, 31/10/2013).

Los proveedores de redes sociales como Facebook "venden" la idea de que pueden responder mejor a los intereses de sus usuarios utilizando sus perfiles y posts (¡y capturan muchas otras informaciones!), pero los utilizan también para servir a sus anunciantes, siendo esta la fuente de sus beneficios[20].



Fuente: Federatas.com.ar, 6/01/2013

World Wide Web

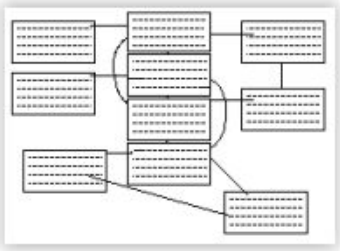
Lo que más a hecho crecer el uso de internet y su difusión fue el invento de la World Wide Web, una modalidad de comunicación que permitía mezclar imágenes y textos, en mensajes de cualquier longitud, con múltiples formas de presentación gráfica en pantalla y con un sistema de enlaces (los "hipervínculos") que permiten que el lector se desplace ("navegue") libremente de una "página" a otra.

Empezó a funcionar en 1991 en algunos centros de investigación y en 1993 vio la luz el "navegador" (*browser*), que permitió que todas las universidades y centros de investigación del mundo se conectaran y lo utilizaran. Finalmente, en 1995, la web se abrió al uso comercial y de cualquier interesado.

20 Instagram, por ejemplo, envía marca y modelo de móvil, país, resolución, versión de Android, nombre de usuario y contraseña, y la contraseña se transmite por medios no cifrados.

La web es administrada por el World Wide Web Consortium (o W3C)[21], una comunidad internacional que desarrolla los estándares que aseguran el crecimiento de la Web a largo plazo.

Hipertexto



La característica fundamental de la web es su construcción como red de contenidos intervinculados, llamado hipertexto.Theodor Nelson fue quien acuñó la expresión "hipertexto" en 1981, pero fue el invento de la Web que le dio su difusión. Él explicaba: *"Con 'hipertexto', me refiero a una escritura no secuencial, a un texto que se bifurca, que permute, que el lector elija y que se lea mejor en una pantalla interactiva. De acuerdo con la noción popular, se trata de una serie de bloques de texto conectados entre sí por nexos, que forman diferentes itinerarios para el usuario."*[22]

El hipertexto se compone de múltiples fragmentos o "páginas-pantalla", llamadas técnicamente "lexias". Una lexia puede ser, dada la capacidad multimedial de las aplicaciones computacionales de hoy, un conjunto compuesto de texto, imagen (fija o video) y sonido. A cualquier componente de la lexia – generalmente a alguna palabra – se puede conectar otra lexia mediante un hipervínculo o enlace (*"link"*): clicando encima, se abre la nueva lexia (generalmente otra página web, pero también puede ser un cuadro encima de la página que está a la vista). De este modo, se permite al lector leer lo que desee en el orden que prefiera. Se forma así una red de contenidos.

El lenguaje HTML es el que se utiliza para dar forma a la presentación de la página y definir los hipervínculos, marcados por un subrayado o un color diferente del texto. El puntero también cambia de forma cuando pasa encima.

La "internet de las cosas"

Se está abriendo una nueva área que extenderá mucho el uso de internet y su presencia en el tráfico de telecomunicaciones: la multiplicación de objetos llamados "inteligentes" porque contendrán procesadores digitales y una conexión a la red para enviar los datos que serán capaces de recoger o de producir. ¿Qué objetos? Conocemos hace tiempo los sistemas de alarma para proteger casas y departamentos, los que han sido mejorados incluyendo cámaras de video y pueden ser monitoreados con un teléfono móvil. Pero ahora, casi todos los aparatos eléctricos que tenemos en la casa podrán ser controlados de este modo e incluirán también otras funciones. El refrigerador, la lavadora e incluso las nuevas ampolletas podrán informar a sus fabricantes del uso que les damos y eventualmente de sus fallas. El refrigerador, incluso, podrá en el futuro avisar al supermercado que elijamos de los alimentos que nos falten, para su reposición (ya que, dentro de algunos años, todos vendrán con una etiqueta electrónica que el refrigerador podrá "leer").

La "internet de las cosas" es referida generalmente como "IoT" por su nombre en inglés: *"Internet of Things"*.

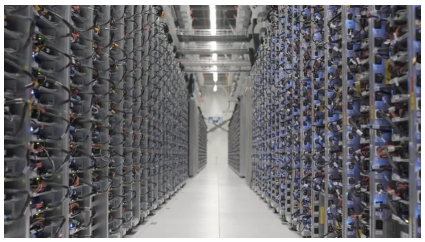
Nubes

La "nube" es un concepto que designa el sistema de conservación de datos en grandes servidores, en este caso computadores con grandes cantidades de discos o chips de memoria, en que tanto empresas como

21 https://www.w3.org/

22 En Landow, G. (1995): *El hipertexto*, Barcelona, Paidos, p.15.

particulares pueden alojar sus datos e incluso programas. En realidad, se debería hablar de las nubes (en plural), porque cada empresa puede crear su propia nube.



Parte de uno de los centros de datos de Google

Nuestros datos de correo electrónico están en la nube, sea de Google, de Microsoft o alguna otra. Y podemos conservar nuestros datos en la nube Google Drive, Dropbox o alguna otra.

Varias de estas empresas ofrecen también otros servicios, desde páginas web hasta aplicaciones de inteligencia artificial destinada a apoyar negocios: son entonces "plataformas", como son el caso de Google, Microsoft y Amazon (que tiene mucho más que su servicio de compras en línea).

El corazón (¡comercial!) de una plataforma es su capacidad para acumular datos acerca de sus usuarios y, de este modo, ofrecerles mejores servicios (más personalizados). Mientras más "posteamos" en Facebook, por ejemplo, más afinará los mensajes y recomendaciones que nos mandará pero, a la vez, gana en valor como empresa. Y mientras más amigos participan, más necesitamos estar presentes también. Esto es el magnetismo de las grandes redes. Investigadores de la Universidad de Stanford (California) y la Universidad de Cambridge (Reino Unido) han demostrado que, de media, "*bastan 10 «me gusta» en Facebook para que sus algoritmos te conozcan mejor que un compañero de trabajo. Con 70 «me gusta» Facebook te conocerá mejor que un amigo, con 150 mejor que tu familia, y con 300 mejor que tu esposa o marido.*" (Gizmodo, 13/01/2015).

5.2. Inteligencia artificial

Origen

John McCarthy
(de Wikipedia)

El concepto y los criterios de desarrollo de la inteligencia artificial se remontan a la intuición del matemático inglés Alan Turing y el apelativo "Inteligencia Artificial" se debe a John McCarthy, uno de los integrantes del "Grupo de Darmouth", un grupo de investigadores que se reunió en 1956 en el Dartmouth College (en Hanover, Nuevo Hampshire, Estados Unidos), para discutir la posibilidad de construir máquinas que no se limitaran a hacer cálculos prefijados, sino operaciones genuinamente "inteligentes".

A partir del grupo inicial de Dartmouth, se formaron dos grandes "escuelas" de I.A. La primera es la de la Universidad Carnegie-Mellon que proponía desarrollar modelos de comportamiento humano con aparatos cuya estructura se pareciese lo más posible a la del cerebro y las redes neuronales artificiales. La otra es el equipo del Instituto Tecnológico de Massachusett (MIT), que se centró más en que los productos del procesamiento tengan el carácter de inteligente, sin preocuparse por que el funcionamiento o la estructura de los componentes sean parecidas a los del ser humano.

El desarrollo teórico de la I.A. es el fruto de numerosos esfuerzos experimentales por desarrollar procesos que sean ejemplos de "comprensión cognoscitiva", en algunos temas acerca de los cuales se entrega información a la máquina. Se verifica cada vez cuanta información se ha de darle y cuantas reglas han de programarse para que el resultado muestre la esperada "comprensión". La explicitación de estas reglas y del modo en que se aplican es el principal fruto de la investigación y el aporte de la I.A. al desarrollo del conocimiento acerca del ser humano y de su inteligencia.

Un computador que opere en el nivel llamado de "comprensión cognoscitiva" - meta de la I.A. -, debiera poder hacer las siguientes cosas:

- Modificar su información como resultado de sus interacciones (equivale a "aprender").
- Relacionar la información presente con las pasadas de una manera inteligente, es decir, haciendo comparaciones útiles y seleccionando las relaciones importantes e interesantes.
- Formular nueva información: deducir conclusiones a partir de la información acumulada.
- Explicarse: mostrar por qué ha establecido las conexiones que ha establecido y qué proceso siguió para llegar a su conclusión.

Estamos avanzando en las tres primeras condiciones, pero aún estamos lejos de cumplir con la cuarta.

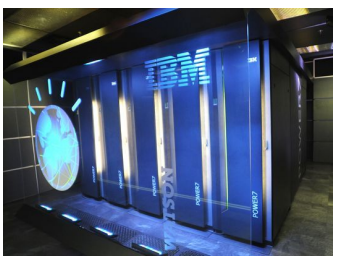
I.A. convencional

La forma convencional (siguiendo la escuela del MIT) se basa en la acumulación de datos y su análisis formal y estadístico para resolver diferentes problemas. Aquí, los investigadores se encontraron con que sus sistemas sucumbían ante la creciente longitud y complejidad de la programación, que debía considerar innumerables reglas. Se desarrollaron múltiples programas para hacer frente a problemas específicos, que son los llamados "sistemas expertos". También se utilizan muchas veces en el actual análisis de datos masivos (*big data*).

Para construir un sistema experto, un programador constituye una "base de conocimientos", la asocia a un "motor de inferencia" (sistema de procesamiento para sacar conclusiones) y le adjunta una interfaz entre la máquina y el usuario (generalmente basada, aún, en un sistema de palabras claves).

Neuronas virtuales (simuladas)

La observación del cerebro, siguiendo la escuela de Carnegie-Mellon, ha ganado más importancia y éxito en los últimos años gracias, sobre todo, al poder creciente de los computadores. En los supercomputadores de hoy, en efecto, es posible simular las redes cerebrales, lo cual ha permitido avances rápidos en el proceso de "aprendizaje" de la máquina. Permitiría resolver los problemas un poco como un cerebro humano, mediante un complejo proceso de reconocimiento de patrones distribuidos a través de muchos nodos o "neuronas" virtuales. El actual poder de cómputo ha permitido a las redes neuronales simuladas reconocer imágenes, palabras y caras, así como pilotar coches de auto-conducción y ganar en competencias de Go y Jeopardy. Han sido progresos notables, pero aún falta mucho para una real imitación del cerebro humano.



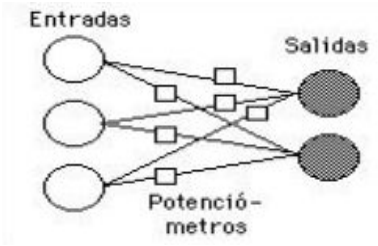
Los supercomputadores más potentes se construyen de acuerdo a la arquitectura de von Neumann pero incluyen una gran cantidad de procesadores que funcionan en forma paralela. En palabras de Raúl Fernández, uno de los investigadores de IBM, su supercomputador Watson [foto al lado] maneja el lenguaje *"como lo haría un niño precoz"* (PC World, 11/11/2016).

El "aprendizaje profundo" (*deep learning*), el mayor avance actual en inteligencia artificial, implica que los datos pasan a través de grandes conjuntos de neuronas simuladas en grandes computadores de arquitectura clásica. Las redes neuronales "profundas" del programa *DeepMind* de Google pueden responder a un 60 por ciento de las consultas formuladas acerca de los artículos que conserva en su nube (MIT Technology Review, 19/06/2015).

Avanzar aquí supone contar con procesadores mucho más potentes.

Neuronas artificiales

La mayoría de los científicos de computación piensan que es sólo el comienzo de lo que será posible. Este tipo de *hardware* se ajusta muy poco en su arquitectura a lo que son las redes neuronales biológicas. Esto significa que puede tomar días o incluso semanas entrenar una red neuronal para resolver un solo problema incluso en un conjunto de computadores y luego requerir una gran cantidad de energía para resolver el problema una vez que están entrenados. Por esta razón se trabaja en la creación de procesadores que imitan las neuronas, o sea en la fabricación de neuronas artificiales.



Para comprender las operaciones básicas del tipo más simple de red neuronal, imagine tres nodos de entrada dispuestos en una columna y una columna de dos nodos de salida a la derecha de los de entrada. Cada uno de los tres de entrada está conectado por cables a cada uno de los dos de salida; pero en la mitad de cada cable hay un interruptor que puede cerrar el paso de la corriente eléctrica a través del cable desde el nodo de entrada al de salida.

Cuando se expone cualquiera de los tipos de nodos a una corriente lo suficientemente grande, se dice que el nodo está «activo»; cuando la corriente es muy pequeña o inexistente, está inactivo. Para resolver un problema con una red de este tipo, se puede asignar a cada nodo de entrada la representación de un rasgo particular del problema (y atribuir un determinado significado a los de salida) [...] Por supuesto, la clave para que la red produzca la respuesta correcta está en ajustar los interruptores entre los nodos de entrada y de salida. En las redes más sencillas, este ajuste puede ser simplemente cosa de abrir o cerrar por completo cada uno de ellos. (Freedman[23], 90-91)

No se usan hoy simples interruptores (sistema binario) sino sistemas que permiten ajustar los valores en una escala continua (potenciómetros). Se habla así del "peso" de una conexión, el que debe ser ajustado para que una determinada entrada produzca la salida correcta, lo cual corresponde a una larga tarea de entrenamiento de esta red.

Actualmente, el más avanzado en el desarrollo de redes neuronales artificiales es IBM, con sus chips neuromórficos TrueNorth y los "resistivos" (*Resistive Processing Units*, RPUs), que imitan una sinapsis cerebral: reciben una multiplicidad de entradas analógicas (voltajes) y utilizan una función de "ponderación" basada en la experiencia pasada para determinar qué debe ser traspasado a la etapa siguiente. trabajando en conjunto. Junto con chips CMOS (como los de las cámaras fotográficas) para las funciones de ingreso y salida de información, forman la llamada DNN: *Deep Neural Network*.. El trabajo reciente (2016) ha permitido su operación con una rejilla de 4.096 x 4.096 RPUs, que permitió realizar 51

23 Freedman, D. (1995): *Los hacedores de cerebros*, Santiago de Chile, Ed. A.Bello.

giga-operaciones por segundo. Esperan construir próximamente un computador con 100 RPUs, que podría operar como una red neuronal de 16.000 millones de sinapsis y sería 30.000 veces más eficiente que los sistemas actuales (Extreme Tech, 1/04/2016).

Dificultades

La mayor dificultad que enfrentan los desarrolladores de la I.A. ha sido la programación del sentido común que poseen los humanos.

> *"Los intentos de programar todas las leyes del sentido común en un único ordenador se han complicado, simplemente porque hay muchas leyes del sentido común. Los humanos aprendemos estas leyes sin esfuerzo porque continuamos tediosamente tropezando con el entorno a lo largo de nuestra vida, asimilando tranquilamente las leyes de la física y la biología, pero los robots no lo hacen.*
>
> *El fundador de Microsoft, Bill Gates, admite: «Ha sido mucho más difícil de lo esperado permitir que ordenadores y robots sientan sus entornos y reaccionen con rapidez y precisión [...] por ejemplo, las capacidades para orientarse con respecto a los objetos en una habitación, para responder a sonidos e interpretar el habla, y para coger objetos de varios tamaños, texturas y fragilidades. Incluso algo tan sencillo como ver la diferencia entre una puerta abierta y una ventana puede ser endiabladamente difícil para un robot». " (M. Kaku[24], 147)*

5.3. Seguridad

Numerosos expertos advierten del peligro del uso de computadores e internet si no se siguen las normas de seguridad básicas en línea y son muchos los ciberdelincuentes que, conscientes de que muchos usuarios no siguen las normas más básicas de seguridad, intentan aprovecharse de la situación (SYNC, 15/11/2017). La seguridad tecnológica es una preocupación creciente en el mundo de las redes digitales. Grandes compañías como Yahoo, Sony y Equifax han perdido datos sensibles de millones de clientes. pero millones de particulares, también, son víctimas de los ataques de los piratas. El impacto nunca ha sido tan grande como en 2017, declaró Tyler Shields, Vicepresidente de Estrategia de la compañía de seguridad Signal Science (Cnet, 9/12/2017).Y se teme que la situación se vuelva peor con los aparatos de la internet de las cosas, que son criticados por su escasa protección.

Debemos distinguir dos grandes tipos de ataques: la penetración directa y la difusión de *software* maligno (*malware*).

Penetración

Se habla de "guerra informática" o *I-War* cuando uno o varios piratas, a título individual o por cuenta de una empresa o gobierno, intentan aprovechar los defectos de las redes, aplicaciones o equipos para causar daños a un "enemigo" que puede ser desde el computador de una pequeña empresa hasta los sistemas militares de un estado. Una inversión de 200 millones de dólares en recursos para la guerra informática serían suficientes para "poner de rodillas" al sistema económico norteamericano, lo cual está al alcance de cualquier país del Tercer Mundo. Si bien internet es resistente a ataques parciales debido a su estructura distribuida, no lo es la red satelital de telecomunicaciones y es especialmente débil la red de posicionamiento global (GPS), donde ligeros errores ya pueden causar desastres.

24 Kaku, M. (2016): *Física de lo imposible*, Santiago de Chile, Debate.

Si bien la idea básica es esencialmente militar, las herramientas de esta guerra están también al alcance de civiles y – lamentablemente - de terroristas. El objetivo es afectar de algún modo el computador o la red del oponente para entorpecer su desempeño o destruir completamente su capacidad operativa, lo cual ya es posible con el mero envío de un virus si el sistema del "enemigo" no está bien protegido.

En un incidente que se dio a conocer el 12 de diciembre de 2017 y que fue calificado como "un punto de inflexión", los atacantes consiguieron desactivar los sistemas de seguridad de una planta industrial. En este caso en particular, mientras sondeaban el *software* del blanco, cometieron un error que hizo saltar las alarmas y la planta detuvo automáticamente su funcionamiento. El *malware* usado durante esta operación ha sido bautizado como Triton y afectó el funcionamiento del *software* de seguridad industrial Triconex, de Schneider Electric. Es la primera vez que los intrusos consiguen inhabilitar el monitoreo de seguridad de una instalación industrial, con excepción del virus de guerra Stuxnet, descubierto en 2010, cuyo objetivo era destruir las centrífugas de la planta de depuración de uranio del proyecto nuclear iraní (La Nación de Argentina, 16/12/2017).

Pero cada vez más frecuentemente se trata de ingresar ilegalmente en el sistema para robar datos, como en los casos señalados de Yahoo y otros, o pervertir las aplicaciones. El robo de datos personales es un negocio muy lucrativo, dado que estos permiten muchas veces acceder a cuentas bancarias o comerciales. Los ladrones generalmente no aprovechan ellos mismos estos datos, sino que ponen las bases de datos a la venta para que otros las exploten. De ahí que los países de la Unión Europea formularan leyes muy exigentes en torno a la protección de los datos personales.

Otra forma de guerra de este tipo es el espionaje comercial o científico, para acceder a los resultados de las investigaciones o de mercadeo de los competidores. Ataques de este tipo son frecuentes pero muy pocos son dados a conocer debido a que multiplicarían los efectos negativos para los afectados.

No olvidemos también que la mayoría de los estados tienen sus unidades de espionaje cibernético. Y las policías tienen sus unidades de investigación y defensa (La policía francesa cuenta desde los años 80 con una sección de informática) y en la mayoría de los países existen ya leyes aplicables a estos delitos.

Malware

El tipo de ataque quizás más perverso y efectivo de los últimos tiempos es el del *"ransomware",* un *software* malicioso que se autoreproduce y encripta todo el contenido de un computador, exigiendo el pago de un rescate para liberarlo. El 12 de mayo de 2017, un ataque de este tipo dejó en dos días a más de 200.000 usuarios afectados en al menos 150 países. Perturbó el funcionamiento de los hospitales británicos, de las plantas de Renault, de la red informativa de los trenes alemanes, de la compañía estadounidense FedEx, del sistema bancario ruso, de universidades de Grecia e Italia, Hitachi y Nissan en Japón, y de empresas e instituciones de enseñanza chinas entre otras instituciones (ABC.es, 13/5/2017). Otros ataques de este tipo se han producido después pero, al parecer, con menos éxito de difusión ya que las empresas e instituciones habían tomado medidas preventivas. Este tipo de infección masiva es posible por la falta de cuidado de los usuarios porque aprovecha fallas de seguridad de versiones pasadas de los sistemas operativos: las víctimas no han actualizado sus sistemas cuando el fabricante corrigió la deficiencia. Aquí, la inteligencia de los piratas se aprovecha de la ignorancia o del descuido de sus víctimas.

El virus es un s*ofware* que se ejecuta en el computador, tableta o teléfono sin control del usuario, con el fin de dañar la información existente o robar datos. Se auto-reproduce al ejecutarse, multiplicándose así con gran facilidad. Los virus se conocen desde 1959, cuando tres jóvenes programadores de los laboratorios de

Bell Computer, subsidiaria de la AT&T, en New Jersey (Estados Unidos), desarrollaron un programa al que llamaron CoreWar capaz de funcionar en forma autónoma. Se difunden a través de sitios web, anexos a mensajes de correo o por conexión con equipos infectados. La "infección" se produce muchas veces por medio de la llamada "ingeniería social" que consiste en engañar al usuario para que él mismo instale en su PC el software malintencionado u obvie los controles técnicos de seguridad.

Otro tipo de ataque de tipo personal, es el *phishing*. El término proviene de la palabra inglesa *fishing*, que en español significa "pescar" y hace alusión a cómo los estafadores, los *phishers*, tratan de que sus víctimas muerdan el anzuelo. Su objetivo: robar los datos del usuario (contraseñas, claves bancarias, tarjeta de crédito) haciéndose pasar por una institución: el banco, la compañía de gas, la empresa para la que trabaja…

El *pharming* redirige a la persona afectada a una página de internet falsa a través de ventanas emergentes usando como excusa un "error en el sistema" o un premio o sorteo.

Lo más desconocido, porque aparentemente inocuo, es el espionaje de e-mail o *mail tracking*: basta la inserción de un (invisible) signo de 1 pixel en un mensaje de correo electrónico para saber si un mensaje ha sido abierto, donde, en qué equipo (y evidentemente cuando). Es una de las formas que tienen los *"spammers"* (creadores de correo basura) para obtener información sobre nosotros si abrimos sus mensajes.

Defensas

La extensión de los ataques como los de *ransomware* – que parecen tener un fin más económico que destructivo – obliga a recordar la necesidad de aumentar en todas partes las medidas de seguridad. También llama a repensar la estructura de internet y aplicar la inteligencia artificial en la detección y prevención de los ataques. Es evidente desde hace tiempo que los piratas son extremadamente inteligentes y se destacan por sus conocimientos, dejando muchas veces a la saga los especialistas legítimos. También ha de alertar acerca de la necesidad de una educación adecuada de los usuarios comunes.

Contra el *malware*, lo primero e indispensable es instalar en cada equipo un antivirus y un cortafuego (*firewall*). Los hay de pago pero también gratuitos bien calificados como el antivirus Avast y el *firewall* ZoneAlarm. Un buen complemento es el Malware Fighter de Iobit. También hay antiransomwares y bloqueadores de espías (*anti.-trackers*) como Ugly Mail, PixelBlock, o Senders.

Contra el *phishing*, la mejor defensa es verificar siempre la URL a la que se envía, comparando con la de la institución aludida. Es frecuente también observar faltas de ortografía, visible mala traducción, y encabezados impersonales (como *"Señor cliente"*).

No debemos olvidar la importancia de utilizar contraseñas (*passwords*) "fuertes", es decir, de 12 caracteres o más, combinando letras, cifras y signos (como h3$mw@f&o#as), que no formen palabras reconocibles o fechas, y una clave diferente para cada servicio. Como no podemos recordar estas contraseñas, podemos

usar administradores de claves como 1Password o LastPass[25], que se pueden agregar como extensión del navegador y ayudan también a crear contraseñas seguras. Un complemento puede ser utilizar la identificación en dos pasos (*two-factor authentication*), optando por recibir una clave de ingreso por SMS en el teléfono celular cuando deseamos abrir una cuenta (de correo u otra).

Otra defensa, para los casos en que el sigilo es importante es el cifrado (encriptación) de documentos. Para el encriptado se usa generalmente el sistema de clave pública y clave privada (criptografía asimétrica). La clave privada es un número aleatorio de 256 bits que cualquier persona puede generar y no contiene ningún dato personal. A partir de esta clave, se genera la clave pública, pero lo inverso es imposible (con la clave pública es imposible descubrir la clave privada). Así, cualquiera puede encriptar un contenido con la clave pública que se le da a conocer pero sólo el destinatario puede decodificar el mensaje con su clave privada.

(De Nuñez, en Preukschat[26], 209)

5.4. Otros conceptos relevantes

Big data

Las nubes son indispensables para albergar las enormes cantidades de datos (datos masivos o *"big data"*) que acumulan las empresas, especialmente los asociados a las redes sociales. Así, por ejemplo, Google procesa 3.500 millones de búsquedas por día y almacena unos 10 exabytes1 de datos. Tiene más de un millón de servidores. Facebook agrega 500 terabytes de datos diariamente. (Infographic Journal, 6/03/2015). Las empresas tratan de aprovechar los contactos logrados a través de la web o de sus apps

25 https://lastpass.com/

26 Preukschat, A. & col. (2017): *Blockchain: La Revolución Industrial de Internet*, Barcelona, Ediciones Gestión 2000

para conocer mejor a sus posibles clientes y rentabilizar mejor la relación. Los gobiernos ofrecen y recopilan información para orientar su gestión y mejorar sus servicios. Las instituciones académicas, además de darse a conocer, ofrecen cada vez más alternativas de enseñanza a distancia y publican los resultados de sus proyectos de investigación.

¿Qué datos?

Aparte de los datos propios de tal o cual empresas, se reúnen muchos datos personales, de todos los que usamos internet. Los que pueden aparecer y son recopilados cuando utilizamos internet son de cuatro tipos:

- privados: son los que generamos y que "viajan" por medio de sistemas directos de persona a persona (como el correo electrónico y el chat); también pueden incluir documentos en la nube (Dropbox, iCloud, etc.) que no compartimos;
- públicos: son nuestras publicaciones en la web: páginas web personales, blogs, artículos de revistas, documentos en servicios como ISSUU, Scribd, etc.
- sociales: corresponden a nuestras actividades en las redes sociales (mensajes de estado, fotos y vídeos, "*likes*", etc.);
- "rastros" (*data trail*): huellas dejadas involuntariamente por las actividades individuales (p.ej., datos del aparato utilizado, ubicación, enlaces seguidos, etc.).

Los datos públicos y sociales son información revelada voluntariamente. Los datos privados pueden ser analizados con o sin nuestro consentimiento si no son encriptados y dejan siempre algunos rastros utilizables por los proveedores de servicios. Constituyen información "extraída" de nuestro comportamiento en las redes.

¿Para qué?

Facebook, Google y otros realizan un seguimiento permanente de sus usuarios y acumulan estas informaciones. Incluso los medios de prensa pueden conocer los intereses de sus lectores. Con los móviles y las redes sociales, las empresas conocen nuestras listas de amigos, nuestros gustos, donde hemos estado, y más. Si bien declaran utilizar los "rastros" de nuestra actividad en sus servicios, gigantes como Google, Facebook y Twitter los combinan con todo lo que les hemos dicho públicamente de nosotros mismos y lo que han observado sin que lo sepamos no sólo para "mejorar su servicio", sino para para obtener ingresos mediante la venta de estos datos.

¿Cuáles son los usos que las empresas dan a las herramientas de análisis de grandes masas de datos? Realizan principalmente (48%) un análisis del comportamiento de los consumidores para poder predecir su comportamiento futuro (Betanews, 21/11/2014). Este es uno de los campos que más les interesan.

¿Quiénes?

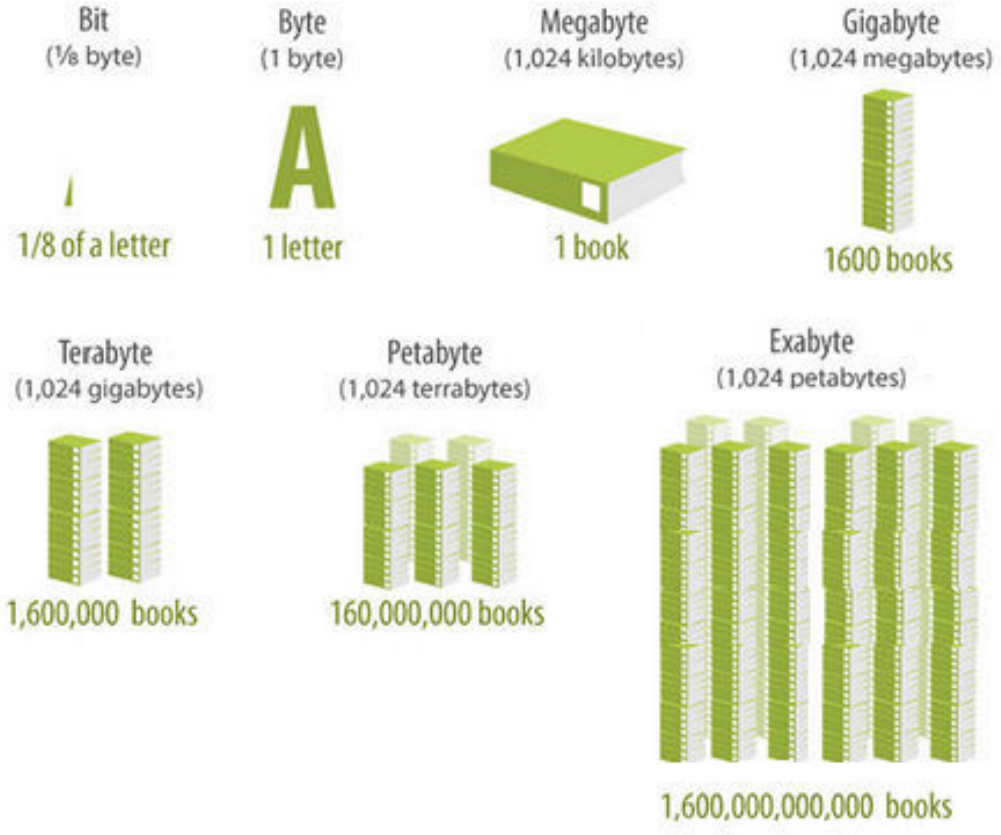
No sólo las empresas de redes sociales acumulan y usan estos datos. Otras, como Google, IBM, Microsoft y Amazon, ofrecen servicios de recopilación y análisis, muy útiles para el marketing de empresas de todo tipo.

¿Cómo?

Las empresas de internet – como muchas otras – acumulan una cantidad enorme de informaciones, superior a lo que es posible humanamente conocer y manejar. Por ello, necesitan sistemas automáticos para extraer los datos que les son realmente útiles. Es la tarea de sus departamentos de inteligencia artificial.

Bits, bytes y más

Google ya almacenaba unos 10 exabytes de datos en 2015 y Facebook agregaba en su nube 500 terabytes de datos diariamente. El siguiente gráfico explica lo que significan estas medidas (Infographic Journal, 6/03/2015).



Blockchain

El *blockchain* o cadena de bloques puede ser entendido como un libro mayor contable ("*ledger*") o como una base de datos en la que se apuntan múltiples transacciones. Es distribuida, es decir, que se encuentra repartida en numerosas máquinas, donde la información se encuentra replicada de forma idéntica. Contiene la información de cada nueva operación que se realice, en un nuevo bloque que se reproduce en todos los computadores que la contienen ("nodos") y en tiempo real. Y cada vez que se pone un bloque nuevo, ese bloque lleva toda la información de todo lo anterior, que no se puede modificar ni borrar (ni duplicar, ya que sería una nueva operación: ¡nada de *spam*!).

Cada bloque contiene un número de bloque, un número arbitrario llamado "nonce" y un campo de datos que contiene los datos de la operación ("transacción") y, además, la fecha y hora (*"timestamp"*), la versión anterior completa de la cadena (*"previous hash"*) y la firma criptográfica del bloque. El *hash* es la "traducción" criptográfica de la cadena al terminar la transacción, que es lo que le da su seguridad. Para crear un bloque se requiere una aplicación específica (*software*) y un procesador adecuado. Quienes participan en la tarea de verificación de una transacción se denominan "mineros".

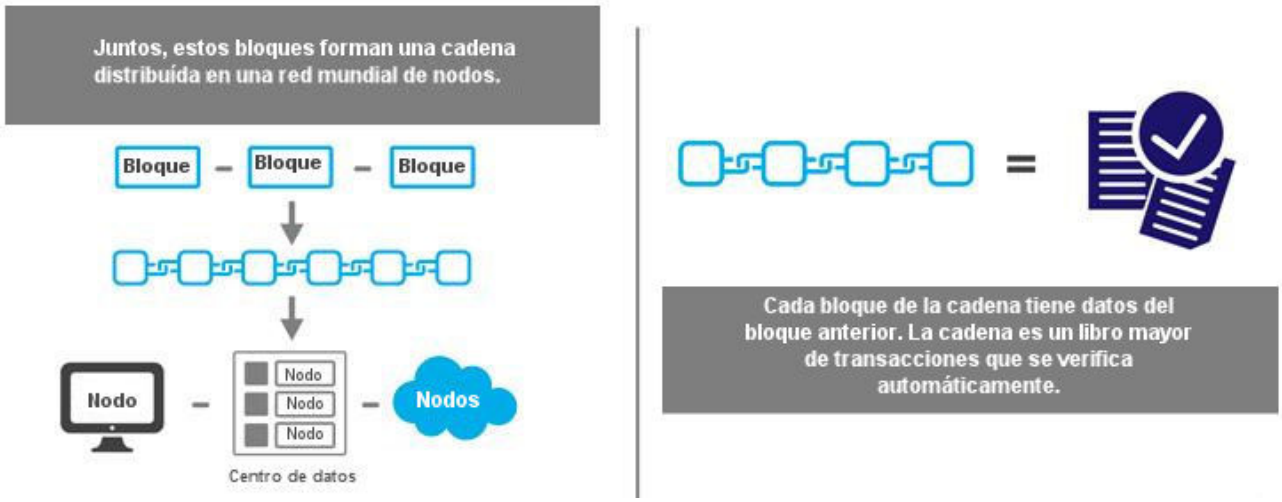


Ilustración adaptado de PC Magazine

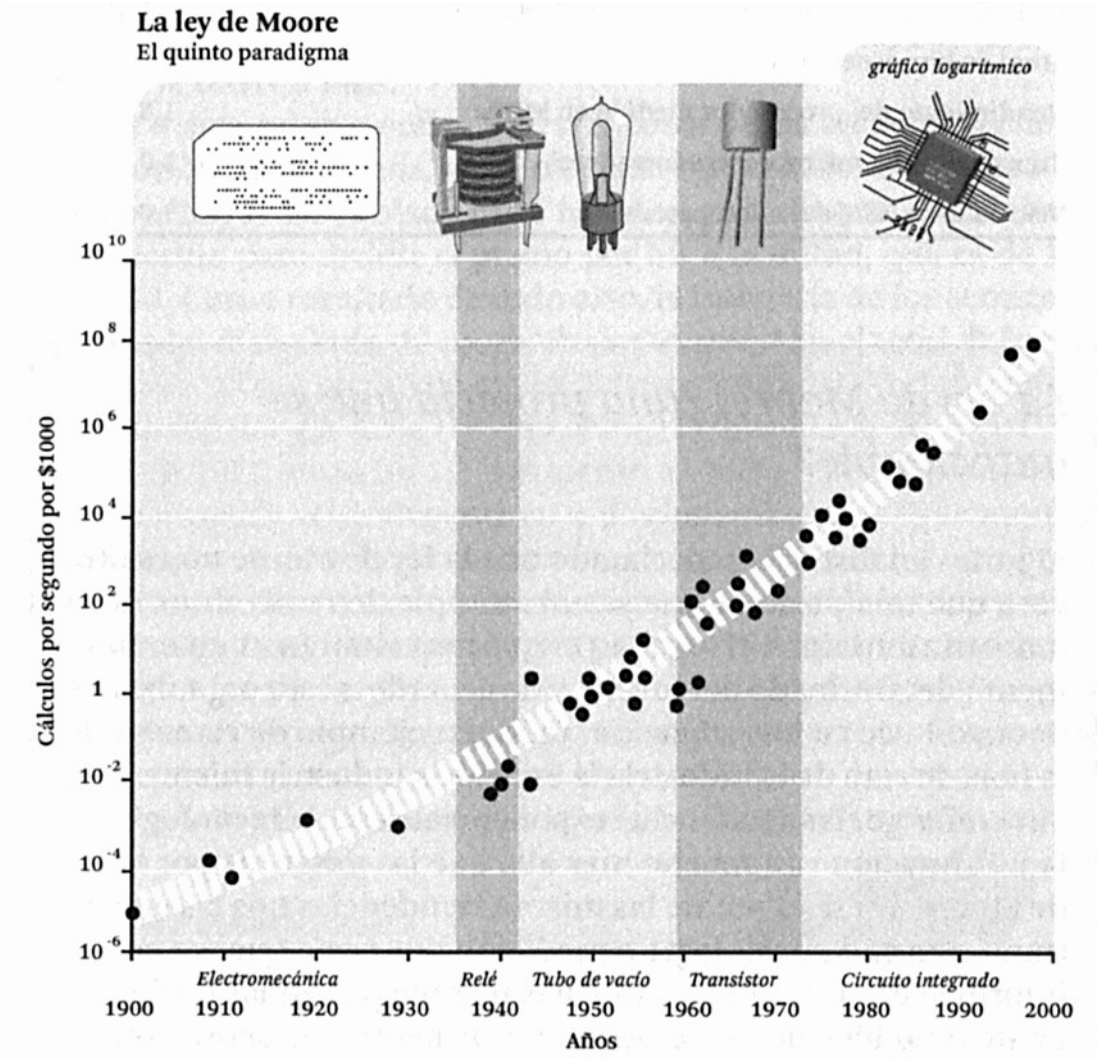
La historia del *blockchain* empezó con el protocolo desarrollado por Satoshi Nakamoto, el misterioso creador del Bitcoin, la criptomoneda que "dio la partida" a esta tecnología en 2008. Varios años después se adoptó para crear otras criptomonedas y, actualmente, se ha empezado a su aplicar en muchos otros campos debido a la seguridad que ofrece. Su potencial es enorme como advierten en la revista PC Magazine: *"Cuando se trata de activos y transacciones digitales, puede poner absolutamente cualquier cosa en una cadena de bloques. En el próximo puñado de años, grandes franjas de su vida digital pueden empezar a correr sobre una base de cadenas de bloques, y es posible que ni siquiera se dé cuenta de ello."* (PC Magazine, 2/8/2017). Para saber más acerca de cómo funciona, consulte mi breve texto *"Blockchain para periodistas y medios de comunicación"*[27].

La ley de Moore

A mediados de la década de 1970, Gordon Moore, un inventor de circuitos integrados que fue más tarde presidente de Intel, observó cómo cada veinticuatro meses era posible doblar el número de transistores que se podían encajar en un circuito integrado. Es lo que se conoce desde entonces como la "ley de Moore", que se ha verificado hasta ahora y podría seguir vigente por algunos años gracias a nuevos descubrimientos en materia de física de semiconductores. Los procesadores también se han vuelto mil veces más rápidos en los últimos treinta años porque los electrones tienen que recorrer menos distancia en los chips (vea el gráfico de la página siguiente) (Kurzweil, 68).

Estamos en una etapa en que los procesadores ya tienen varios núcleos (trabajando en paralelo) y se están empezando a colocar en capas (pasando de lo bi– a lo tridimensional). Con ello, el ritmo de la innovación tecnológica se acelera. Esto, evidentemente, afecta la cantidad de datos que pueden ser recopilados y procesados, explicando lo que ocurre actualmente con internet, como el desarrollo de la "nube" y la capacidad de empresas como Google para conservar y exhibir todo lo publicado.

27 Ediciones INCOM-Chile, 2017. Descargable en PDF de https://goo.gl/AEMD9A



(De Kurzweil, 74)

6. Diccionario complementario

A

ADSL
Conexión a internet por línea telefónica digital de alta velocidad ("banda ancha")

Ancho de banda:
Cantidad de bits que se pueden transmitir por segundo a través de un determinado canal.

Aplicación:
Serie de instrucciones que define las operaciones que ha de realizar el computador. (Sinónimo de "programa").

Aplicaciones interactivas:
Herramientas computacionales en las que el usuario tiene un papel activo, es decir, ingresa datos, toma decisiones o responde consultas.

App:
Aplicación (programa) que se instala en un teléfono móvil o tableta.

ASCII (*American Standard Code for Information Interchange*):
Sistema de codificación americano estándar de los impulsos correspondientes a las teclas (letras, números y comandos), para el intercambio de datos.

B

B2B:
"Business to Business": comercio electrónico entre empresas.

B2C:
"Business to Consumer": comercio electrónico entre empresas distribuidoras y particulares (venta de productos o servicios).

Banda ancha:
Se dice de lías de telecomunicación capaces de transmitir a alta velocidad y en forma simultánea un alto número de "paquetes" de datos.

Base de datos:
Lugar donde se almacena información computacional (Archivo especialmente estructurado para su actualización y consulta electrónica). Permite buscar, ordenar y realizar múltiples operaciones sobre dichos datos.

Baud:
Unidad de medida de la velocidad de transmisión de un bit en un canal de comunicación.

BBS (*Bulletin Board Systems*):
Sistema de boletines, parecido al correo electrónico, donde los mensajes pueden ser leídos por todos los abonados de una red local o amplia. (Hoy día sustituído por los "News".)

Beta testing:
Puesta a prueba de una aplicación para usuarios finales (Última etapa antes de producir la versión final).

Bit:
Unidad de medida binaria de la información (vale 0 o 1).

Bitmap:
Formato basado en "mapa de puntos". Es uno de los formatos posibles para la conservación de imágenes, usado para fotografías y gráfica analógica (como caricaturas y pinturas). Se opone a vectorial o "Bezier", que utiliza coordenadas geométricas y fórmulas trigonométricas.

Bluetooth:
Sistema de comunicación inalámbrica de corto alcance (un par de metros), que requiere un camino sin obstáculos entre los dos dispositivos a comunicar. Se usa generalmente para conectar audífonos inalámbricos al teléfono móvil.

Browser (visualizador o navegador):
Aplicación utilizada para revisar los contenidos de una base de datos o de directorios de hiperarchivos (como es el caso de las páginas de Web).

Bug (mosca o rana):
Error en un programa.

Bus:
Camino (cableado) que recorren los pulsos dentro del computador.

Byte:
Bloque de 8 bits; es lo necesario para codificar un carácter en el código americano usado para los teclados (ASCII).

C

CAD:
"Computer Aided Design": corresponde a un sistema de dibujo tridimensional "asistido por computador", especialmente orientado al diseño arquitectural e industrial. En el área industrial, un detallado proyecto CAD puede transformarse en instrucciones para una máquina-herramienta capaz de modelar la pieza diseñada (proceso "CAD-CAM": "*computer aided design & computer aided manufacturing"*)

Carácter:
Número, letra o símbolo (introducido a través de un teclado).

Chatear:
Conversar mediante mensajes de texto ("chat").

Chip:
Pequeñísima placa de silicona con multitud de circuitos electrónicos.

Circuito integrado:
Circuito eléctrico que contiene decenas o miles de componentes electrónico, agrupados en un bloque indivisible en el proceso de fabricación. Es la parte medular de un chip.

Clave de acceso o contraseña:
Ver *password*.

Compatibilidad:
Son compatibles los computadores capaces de realizar los mismos programas.

Compilador:
Programa que traduce un programa fuente en lenguaje de máquina, para que ésta los ejecute en forma expedita.

Computador cuántico:
A diferencia de los clásicos, los computadores cuánticos no operan con bits (que tienen un valor definido de uno o cero), sino con qubits que pueden estar en una superposición de estados. Durante la medición del estado de un qubit, este pasa a uno de estos estados. Pero llevando a cabo varias mediciones se puede juzgar la distribución estadística entre ellos. De este modo, los computadores cuánticos pueden resolver varios tipos de tareas mucho más rápido pero son mucho más difíciles de construir.

Concentrador:
Elemento que divide la información de un canal de datos en dos o más canales que transportan, cada uno, dicha información.

Conectividad:
Disposición de los equipos computacionales para conectarse entre sí.

Cookie (galleta):
Conjunto de datos asociados a una página web que se guarda en el disco duro del usuario y, mediante intercambio con el servidor de web, determina su navegación posterior.

Correo electrónico (*e-mail*):
Sistema de envío y recepción de correspondencia privada (Todos los usuarios tienen una clave de acceso -*password*- que asegura la confidencialidad). A los mensajes escritos y enviados con la aplicación correspondiente es posible agregar archivos de otros tipos, que son transmitidos en formatos diferentes.

CPU:
Unidad Central de Proceso: circuito que gobierna el funcionamiento del computador.

D

DSL:
"Digital Subscriber Line": línea telefónica digital de alta velocidad ("banda ancha").

Disco duro:
Disco magnético en que se conservan los programas (aplicaciones) y datos.

E

Encapsulado:
Un encapsulado es un documento o una aplicación protegida total o parcialmente contra la lectura o el uso indebido. Para acceder a todo el contenido se requiere algún tipo de autorización (clave de acceso o decriptado). Es muy común en los programas de demostración ("demos"), donde la "cápsula" impide algunas funciones claves (como "Guardar") o bloquea el uso después de cierto tiempo.
Para producir un encapsulado, se utiliza un programa que "envuelve" el documento o la aplicación de tal modo que no sea accesible en su forma original. Es probable que esta técnica se use cada vez más frecuentemente en Internet, para proteger las comunicaciones de la piratería.

Enrutador (*router*):
Elemento de red que realiza la función de asignar direcciones a paquetes de datos entre dos redes o subredes.

Ethernet:
Es un tipo de red de comunicación que transmite a una velocidad estándar de 10 a 1000 Mb/s (miles de bits por segundo).

F

Fibra óptica:
Medio de transmisión de datos que en vez de transmitir una señal eléctrica transmite un haz de luz. Es más rápido que el cable común (cobre) y alcanza mayores distancias sin pérdida de señal.

Foro:
Grupo de discusión en la red ("*News Group*"), típico de los primeros años de internet (antes de la web).

FreeWare:
Aplicaciones (programas) disponibles sin costo para cualquier utilizador potencial.

FTP (*File Transfer Protocol*):
Sistema de transferencia de archivos; permite enviar y recibir archivos de cualquier formato. Existen servidores de FTP que conservan aplicaciones de dominio público ("*FreeWare*") que los interesados pueden traer a su computador.

G

Grupo de interés:
Ver *"News Group"*.

GSM (*"Global System for Mobile communications"*)
Sistema global para las comunicaciones móviles. Fue el segundo sistema estándar (2G) utilizado por los teléfonos celulares (más lento que los actuales).

H

Háptico:
La palabra griega "háptico" significa perteneciente al sentido del tacto. Las interfaces hápticas son las que usan el sentido del tacto; su usuario puede sentir – en los dedos o alguna otra parte del cuerpo – lo que se transmite (con un guante o un traje especial). De este modo se podría acariciar a alguien que esté al otro lado del mundo.

Hardware:
Componente sólido (computador, disco duro, teclado, ratón, etc.).

HDMI (*"High-Definition Multimedia Interface"*):
"Interfaz multimedia de alta definición", norma de transmisión de video sin compresión.

Hexadecimal:
Código basado en 16 dígitos (de 0 a 9 y luego de A a F), por lo cual 2 dígitos hexadecimales forman 1 byte. Se utiliza algunas veces para programar, pero es la forma más difícil de hacerlo (para más rápida para ser utilizada por el computador).

Hiperarchivo:
Documento que mantienen múltiples vínculos con otros documentos (como en la WWW).

Hotspot:
Puesto de conexión inalámbrica a internet por WiFi para múltiples usuarios (ver WiFi y enrutador).

HTML:
Lenguaje utilizado para el diseño de páginas web. No es propiamente un lenguaje de programación sino de "exhibición" (utiliza "etiquetas" para indicar cómo deben ser exhibidos en pantalla los componentes).

Hub:
Ver "Concentrador".

I

ICANN:
Corporación de Internet para la Asignación de Nombres y Números IP (las "direcciones" de los computadores).

Ingeribles (*"digestibles"*):
Dispositivos digitales miniatura encapsulados en pastillas que parecen tradicionales que pueden ser ingeridos sin generar ningún tipo de problema para sus pacientes pero que recolectan y transmiten información (Techcetera, 29/11/2017).

Interfaz:
Elemento mediador, medio de interacción (por ejemplo entre un equipo físico y el usuario del mismo), definido por características físicas del contacto, características de las señales intercambiadas y significado de las mismas. Así, "interfaz gráfica" es la forma en que las aplicaciones son exhibidas en pantalla por el computador.

Internet:
Red de comunicaciones digitales entre computadores que opera con un mismo protocolo de comunicaciones (TCP/IP), lo cual permite transferencias controladas de datos.

ISP:
"Internet Service Provider": proveedor de servicios de Internet; empresa que canaliza las conexiones a Internet.

J

Jailbreak:
Escribir directamente en la memoria del núcleo del sistema.

Java:
Lenguaje de programación que permite utilizar aplicaciones en computadores con diversos sistemas operativos. Utilizado a veces en combinación con servidores y navegadores de World Wide Web.

Javascript:
Lenguaje de programación compatible con el sistema HTML de las páginas web, que permite interacciones dinámicas a partir de estas páginas (las líneas de programación se insertan dentro de la página Html).

K

Kb (kilobyte):
1024 bytes.

Kernel:
Núcleo del sistema operativo

Keylogger:
Software malicioso que captura todo lo que se teclea y lo envía al autor del mismo.

L

Lenguaje de programación:
Lenguaje que se utiliza para redactar programas de computación.

LTE ("*Long Term Evolution*"):
Evolución del estándar de comunicación móvil de tercera generación (3G o UMTS), de mayor velocidad y capacidad, que se considera como cuarta generación (4G) aunque no cumple con todas las especificaciones de la norma para 4G. Provee velocidades de acceso mayores de 100 Mbit/s (megabits por segundo) en movimiento y 1 Gbit/s (gigabits = miles de bits por segundo) en reposo. Actualmente se trabaja en definir la quinta generación (5G), que podría estar disponible en 2020.

M

Mail:
Ver "Correo electrónico".

Mailbombing:
Ataque a un servidor de correo electrónico con miles de mensajes inútiles para llevarlo a colapsar e interrumpir su servicio.

Mirror:
Duplicado: servidor de web que duplica los contenidos de otro (parcial o completamente)

Modem (Modulador-demodulador):
Aparato que transforma la señal digital (binaria) del computador en la señal modulada que requieren las líneas telefónicas convencionales (e inversamente). De este modo, pueden comunicarse dos computadores por medio del teléfono.

Monedero electrónico:
Aplicación que facilita los pagos utilizando el celular, asociado generalmente a las tarjetas bancarias (que se han de registrar inicialmente). El pago se hace acercando el celular al terminal de pago (igual que las tarjetas sin contacto).

MP3:
Formato de compresión que permite escuchar música con calidad de CD, pero en archivos mucho más pequeños que las grabaciones digitales tradicionales. Esto se logra eliminando las frecuencias que producen los instrumentos pero que el oído humano no es capaz de percibir.

MP4 o MPEG4:
Tecnología de compresión de video que permite reducir los archivos digitales a un 10% de su tamaño original, con una pérdida de calidad casi imperceptible. Fue desarrollada por MPEG (*Moving Picture Expertos Group*) para productores profesionales, pero cayó en las manos de piratas que la difundieron por la red (así como copias piratas comprimidas de películas).

N

News Group:
Antiguo sistema parecido al correo electrónico, donde todos los interesados podían leer y contestar los mensajes. Se organizaban por temas, los cuales determinaban "grupos de interés". Un lector elegía los

temas de su agrado (esto se llamaba "suscribirse"), para leer todos los mensajes que el servidor conservaba en relación a estos temas y poder contestar o agregar mensajes si lo deseaba. Existieron miles de grupos.

Nube ("*cloud*"):
La "nube" es el sistema que permite ofrecer servicios de computación a través de una red, que usualmente es Internet. Lo más común para todos es la oferta de "depósitos" donde guardar documentos (como Dropbox, OneDrive y Google Drive, o los archivos de fotos como Flickr y Google Fotos), pero también se ofrecen aplicaciones o *apps* que pueden realizar diversas operaciones con los archivos que se les envía por internet ("*software* como servicio" o "SaaS").

P

Password (Contraseña o clave de acceso):
Palabra o número secreto que permite acceso confidencial a una máquina, una red, un servicio o un conjunto de datos.

PDF ("*Portable Document Format*"):
Formato de conservación y transmisión de textos destinados a impresión de alta calidad, independientemente del sistema operativo del computador del usuario.

Pendrive (lápiz de memoria):
Ver USB

Periférico:
Aparato anexo, que facilita la entrada o salida de datos, o su conservación externa (como el teclado y el ratón).

Phablet:
Tableta con capacidad de teléfono móvil.

Phishing:
Robo de datos para suplantar la identidad.

Pixel:
Punto básico de la pantalla que el computador puede encender o apagar, para formar figuras (letras o dibujos).

Plotter:
Trazador; impresora gráfica que dibuja imágenes con plumas de tinta. Los trazadores requieren datos gráficos en formato vectorial (o sea, basados en coordenadas geométricas).

Plug-in:
Pieza complementaria optativa de una aplicación (software), que le agrega algún tipo de funcionalidad particular.

PostScript:

Lenguaje de descripción de páginas creado por Adobe Systems para controlar las impresoras láser. Los comandos PostScript (ps) se adjuntan al texto que se envía y son traducidos por un intérprete que está en la memoria de la impresora. Su principal función consiste en manipular el tamaño de todo lo que se imprima (espacios, tipos e ilustraciones) para una óptima reproducción. Opera con un sistema de definición de tipografía que evita almacenar modelos de múltiples tamaños.

Privilegios de acceso:
Posibilidad de ver y modificar archivos en un disco compartido o un servidor de archivos en una red. Los privilegios tienen una jerarquía (desde "sólo ver los nombres" y "sólo leer" hasta "modificar" y "borrar") y es el propietario de la unidad compartida que determina quienes tienen acceso y en qué nivel, para lo cual puede exigir a los "clientes" que estén registrados y se identifiquen con una contraseña (clave de acceso).

Procesador:
Chip que contiene todo lo necesario para que el computador pueda realizar sus funciones.

Programa:
Serie de instrucciones que define las operaciones que ha de realizar el computador. (Sinónimo de "aplicación").

Programación de bajo nivel:
Programa redactado en código hexadecimal (el cual se "traduce" en un sólo y rápido paso - automático- al código binario de la máquina. Se opone a la programación de "alto nivel" que utiliza palabras más cercanas al lenguaje ordinario pero requiere etapas más complejas de traducción para que la máquina pueda operar.

Protocolo:
En comunicación de datos, es el estándar por medio del cual se entienden diferentes equipos computacionales.

Puerto:
Salida o entrada (enchufe) a la cual se conecta un cable de periférico o de red.

R

RAM ("*Random Access Memory*"):
Memoria de acceso directo, donde es almacenada la información que se introduce en el computador; se borra cada vez que se apaga el computador.

RealAudio:
Sistema de transmisión de sonido en tiempo real acoplado a páginas web. Requiere que el navegador de web sea complementado con un "plug-in" que remita la recepción del flujo de datos.

Realidad aumentada:
Sistema visual que permite ver datos proyectados por sobre lo observado en la realidad, gracias a lentes especiales.

Realidad virtual:
Imagen tridimensional creada en computador, que se puede ver con lentes especiales, los que dan la impresión de que su usuario puede desplazarse en un mundo artificial.

Red LAN ("*Local Area Network*"):
Red entre oficinas o departamentos unidos en un radio no superior a unos 2 km.

Red WAN ("*Wide Area Network* "):
Red entre instituciones u oficinas establecidas a gran distancia unas de otras.

Repetidor:
Elemento que se utiliza para extender segmentos de una red mas allá del largo permitido por la pérdida de potencia en el cable.

Resolución:
Número de pixels que el computador controla simultáneamente en el pantalla (Mientras mayor es la resolución de pantalla, más detalles puede tener un gráfico y más trabajo se requiere del procesador).

ROM (*"Read Only Memory"*):
Memoria sólo de lectura; contiene las instrucciones básicas que permiten al computador funcionar (las que se imprimen en la fábrica y son permanentes).

Rootkit:
Conjunto de herramientas usadas por intrusos informáticos para acceder en forma ilícita a un sistema informático .

Router
Ver "Enrutador".

S

Sensor:
Elemento que sirve para captar señales (p.ej. los rayos de luz en un sensor fotográfico o las pulsaciones cardíacas en un sensor de presión arterial).

Servidor:
Computador de gran capacidad que contiene el material al que podemos acceder por internet (p.ej. las páginas web).

Sexting:
Contracción de *sex* y *texting*. "fenómeno de fotografiarse en actitud provocativa para enviar las imágenes a alguien de confianza", una práctica de alto riesgo como ha sido demostrado por el pirateo de fotografías íntimas de celebridades.

Shareware:
Aplicaciones (programas) disponibles a bajo costo para cualquier utilizador potencial (Se debe enviar el pago al autor cuando se decide usar en forma regular).

Sistema operativo:
Software básico necesario para el funcionamiento de un computador.

Software:
Programas que utilizan los computadores.

Software libre:
Programas que -gratuitos o pagados- cuyos usuarios pueden conocer y modificar el "código fuente" (es decir la programación original, cosa que no permiten las aplicaciones comerciales más comunes). También es libre su reproducción.

Software "propietario":
Aplicaciones cuyo código fuente (programación) queda escondida para el usuario y es inmodificable. Su reproducción queda habitualmente prohibida.

Spam:
Correo basura.

Subrutina:
Parte de un programa que realiza una función específica; puede ser "activada" desde distintos puntos de un programa, en repetidas oportunidades.

T

TCP/IP:
Protocolo de control de transferencia de datos establecido en 1982 como estándar de Internet.

Terminal:
Aparato que permite comunicarse con la unidad central de un computador que puede estar a gran distancia (Se trata generalmente de un teclado unido a un monitor -pantalla- más el sistema de transmisión).

Tiempo real:
Tiempo de respuesta en la interacción equivalente a lo normal en un diálogo (se considera habitualmente aceptable una respuesta que demore menos de 4 segundos).

Transistor:
Componente electrónico básico que detiene o deja pasar las señales eléctricas dependiendo de las pulsaciones anteriores. Un chip o circuito integrado contiene un gran números de transistores.

Troll:
Término coloquial para denominar a un "alborotador o polemista que participa en foros cibernéticos".

U

UMTS ("*Universal Mobile Telecommunications System*"):
Sistema universal de telecomunicaciones móviles usado por los móviles de tercera generación (3G), sucesora del GSM.

USB (*"Universal Serial Bus"*):
Tipo de dispositivo de almacenamiento de datos que utiliza memoria flash para guardar datos e información.

V

Virus:
Software que se ejecuta en un computador, tableta o teléfono sin control del usuario, con el fin de dañar la información existente o robar datos. Se auto-reproduce al ejecutarse, multiplicándose así con gran facilidad.

W

Wallet:
Monedero. Ver "Monedero electrónico".

WAP (*Wireless Application Protocol*):
Sistema que permite la transmisión de datos y servicios -como el correo electrónico y algunas formas limitadas de páginas web- a través de Internet entre teléfonos móviles.

Wearable:
A veces traducido como "vestible": *hardware* que se inserta en la ropa (sensores y procesadores).

WiFi:
Sistema de comunicación inalámbrica (ondas de radio) de mediano alcance (normalmente hasta 50 metros), usado por las tabletas y los teléfonos móviles para conectarse a internet a través de un puesto de transmisión (*hotspot*) o enrutador.

Wikipedia:
La mayor enciclopedia disponible en la web, confeccionada por voluntarios que aportan sus conocimientos especializados. Disponible en español en https://es.wikipedia.org/wiki/Wikipedia:Portada

World Wide Web (WWW):
Red mundial de computadores por la cual se transmiten principalmente archivos hiper y multimediales. La forman los computadores conectados a Internet que usan los protocolos de comunicación adecuados.

X

XML ("*Extensible Markup Language*"):
Lenguaje orientado a definir el despliegue de información en pantalla. Es la base del HTML y otros derivados utilizados para la construcción y exhibición de páginas web.

Más definiciones
Si sabe inglés, puede encontrar muchas otras definiciones en el sitio https://www.wordnik.com/ (contiene ocho millones de términos del ámbito de la cultura digital que no están en los diccionarios normales).

Anexo 1: Base de datos periodística

Los medios de prensa tienen crecientes problemas para mantenerse económicamente en el ambiente digital actual. Para muchos, la único posibilidad de sobrevivir será utilizando "muros de pago", es decir, un sistema de suscripción, y esto implica poder responder a los intereses de cada lector en particular, lo cual implica conocer tales intereses y ser capaces de seleccionar y desplegar las noticias y notas en función de ellos.

En consecuencia, se deben combinar y asocial dos tipos de bases de datos: una que contenga la información que se publica y otra que contenga datos relativos a los lectores, formando un sistema integrado. Sólo se puede hablar de sistema documental si el conjunto de archivos se estructura en forma de «espacio de información», permitiendo diversas maneras de lectura de la información que contiene y contemplando la existencia de relaciones entre los distintos archivos que lo conforman.

Un verdadero sistema documental periodístico debe ser analítico, y no simplemente un depósito amorfo de datos inconexos. El diseño del sistema debe partir del análisis de las características propias e intrínsecas de los referentes (objetos y eventos) representados, además de tener en cuenta las necesidades y los hábitos de los usuarios.

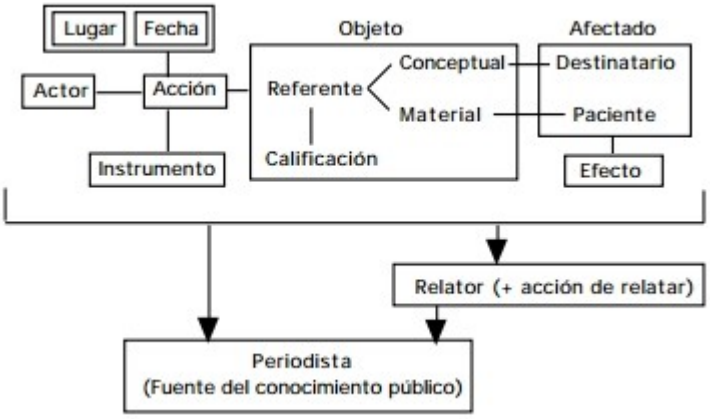
El registro de las noticias ha de ser entendido como la médula de una conversación acerca del referente. Ha de contener tanto lo que el periodista pueda decir del mismo como prever y poder contestar a lo que el lector podrá preguntar. Por ello, debería parecer obvio que la mera transcripción - factible en el caso de textos - no asegura, generalmente, una adecuada respuesta. Si se conservan reseñas periodísticas de hechos noticiosos como textos completos, sin más análisis ni formas diversas de acceso, será imposible satisfacer al lector. Mientras más analítico es el archivo diseñado, más fácil y más precisamente se podrá recuperar la información buscada a posteriori.

Es indispensable configurar un verdadero sistema documental, y no solamente construir diversos tipos de archivos, de acuerdo a la intuición o las necesidades del momento. Por ello, es indispensable basarse en un análisis de los referentes que considere la significación de éstos para las personas que se interesen por ellos. Esto implica contar con un sistema constante de variables en función de las cuales se describen estos referentes (son los "atributos" de los mismos). Luego deben definirse los valores que pueden tomar los atributos. En algunos casos, estos valores se obtienen transcribiendo datos que aparecen en el referente (por ejemplo el nombre del autor y el título de un libro). Estos son datos "intrínsecos". Pero la mayoría de las veces los datos intrínsecos no son los adecuados.

Si analizamos[28] la estructura (lógica y semántica) de una noticia y su relato, como podemos ver en el gráfico siguiente, obtenemos 5 atributos constitutivos, a los cuales se debe poder acceder con facilidad para encontrar respuesta a una pregunta típica. Pero el detalle de la información buscada, a su vez, no estará en el campo de la base de datos que corresponda a alguno de estos atributos - ya que cada uno registra sólo un componente informativo - y deberá aparecer en otra parte. Se puede pensar en un breve resumen que vaya también en la BD, pero lo normal sería incluir ahí una referencia (hiperenlace) al texto completo.

28 No entro aquí en la justificación técnica detallada. Se podrá encontrar en mi libro "Explotar la información noticiosa" y mi artículo "El análisis lógico de hechos noticiosos" (ver Bibliografía).

Estructura de un registro noticioso



Un registro confeccionado de acuerdo a esta pauta podría tener la siguiente apariencia:

Fecha: 2001-04-11
Lugar: Estados Unidos
Título: Yahoo! registró una pérdida neta de 11.49 millones de dólares
Implicado: Yahoo!
Descriptores: Economía, Ética, Portal, Trabajo
Referencia: 20010411Yahoo.html
Autor: J.Pérez [Periodista responsable]

"Implicado" reúne "Actores", "Afectados" y "Relatores" (que son los eventuales testigos entrevistados, que relatan lo ocurrido al periodista): la experiencia a mostrado que no es necesario crear dos campos de datos diferentes. Acción, instrumento, objeto y efecto no requieren ser tratados como atributos independientes: es suficiente su presencia en el relato completo y se incluirán en el resumen si se crea este.

Aquí, en SQL, la definición de formato de una tabla de noticias adecuada. El "id" es el número de la fila de datos, que permite referirse a ella en otras tablas y se ajusta en forma automática (*auto_increment*), será un entero de máximo 11 caracteres (`int(11)`); "varchar" marca la longitud permitida de algunos campos de datos; "default" indica que por defecto se pone un espacio en blanco):

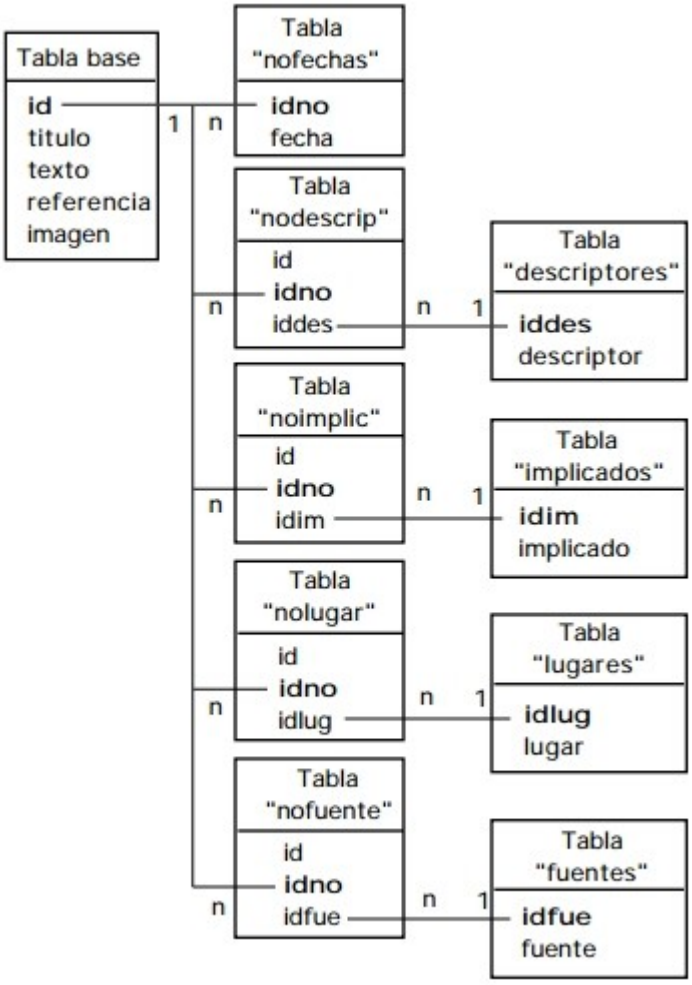
```
CREATE TABLE IF NOT EXISTS `noticias` (
  `id` int(11) unsigned NOT NULL auto_increment,
  `fecha` date NOT NULL default '0000-00-00',
  `lugar` varchar(35) NOT NULL default '',
  `titulo` tinytext NOT NULL,
  `resumen` text NOT NULL,
  `fuente` varchar(70) NOT NULL default '',
  `referencia` tinytext NOT NULL,
  `respingreso` (12) NOT NULL default '',
 PRIMARY KEY  (`id`),
   KEY `fecha` (`fecha`)
)
```

Luego tendríamos, para cada noticias, el comando

```
INSERT INTO `noticias` (`id`, `fecha`, `lugar`, `titulo`, `resumen`,
`fuente`, `referencia`, `respingreso`) VALUES(…);
```

Es indispensable que esta tabla, que es como la "espina dorsal" de la BD noticiosa, sea acompañada de otras, que son las que permitirán distintos tipos de consultas:

Modelo normalizado completo de BD de noticias



La "Tabla base" podrá ser la descrita anteriormente. En este modelo, se consideró solamente una "base" mínima. "Texto" se refiere ahí a un resumen e "Imagen" a un enlace a un archivo gráfico anexo. Se puede ver con facilidad como las tablas se conectan unas con otras ("idno", por ejemplo, es el número identificador de la noticia, "id" en la tabla base). Faltaría un par de tablas en que se registre la identificación de los periodistas responsables de la obtención de cada noticia (que serían una tabla "norespingreso" y tabla "responsables", por ejemplo).

La base de datos de diseminación deberá tener una estructura parecida, con una tabla base de suscriptores y una que vincule los suscriptores y las tablas de la tercera columna del modelo anterior (descriptores, etc.), según los intereses declarados o descubiertos mediante análisis de *"big data"*.

Un sistema documental periodístico, además, es mucho más que una base de datos de noticias o "archivo de crónica". Los archivos que lo componen son, además, típicamente los siguientes:

El "archivo de diseminación" es la base de datos donde se concentran los datos de los lectores, con su identificación y la lista de sus intereses (basados en el los "descriptores" que se usan para clasificar las noticias).

Anexo 2: Ejercicios de programación

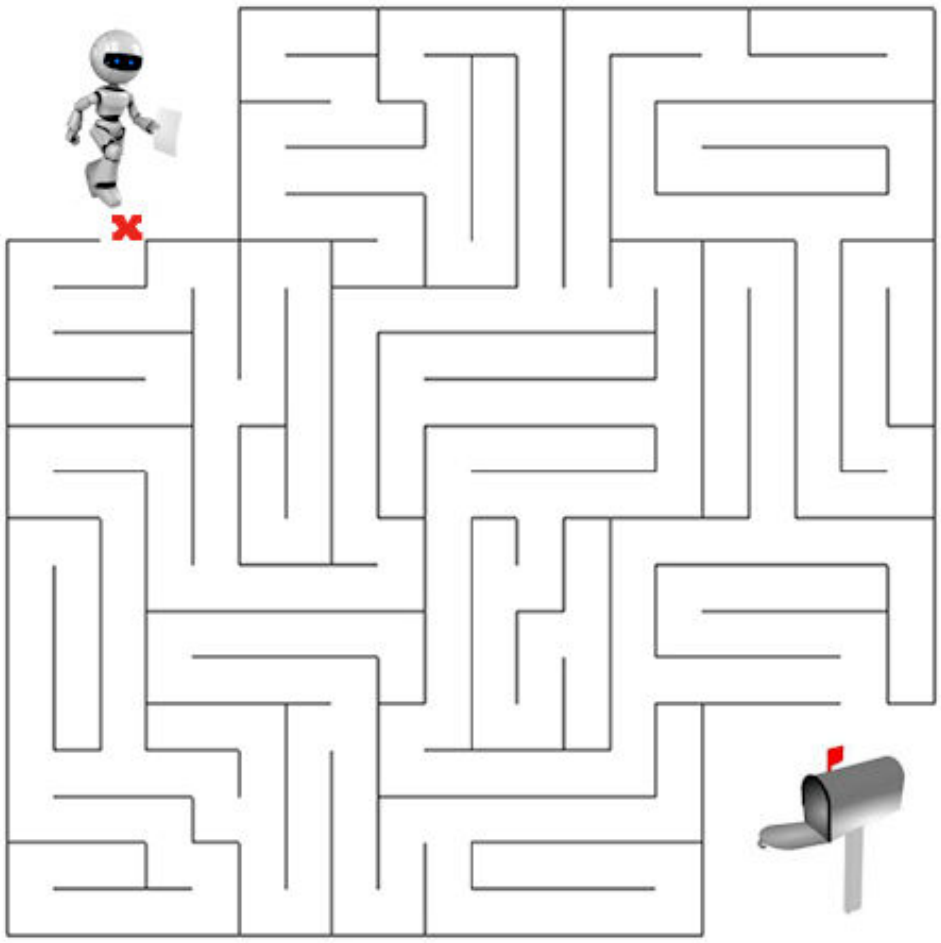
Para todos los ejercicios consideramos suficiente escribir los programas en lenguaje natural sintético o en seudo-código.

1. El robot-cartero

Puede ver a continuación el mapa de un laberinto. El robot debe pasar por él para echar la carta al buzón. Con él, podemos realizar dos ejercicios.

1a. Consideremos que el robot tiene un sensor de contacto, aparte del detector de buzones. Deberá programarlo para que avance por el camino correcto hasta llegar a su meta.
1b. ¿Cómo lo haría ahora con un robot sin sensor de ningún tipo (salvo el detector de buzón)? (Sí puede contar sus pasos. El espacio -ancho- entre dos muros equivale a 2 pasos.)

El robot parte de la posición marcada por la x roja (justo frente a la entrada). Se supone que el buzón está justo frente a la salida.



2. Hacer suma y resta

2a. Escriba un programa que solicite 2 números y luego los sume y muestre el resultado.
2b. Agregue ahora una petición de un número a restar y luego realice la resta y ponga en pantalla el resultado si es positivo o una advertencia si el resultado es negativo.

3. Tabla de multiplicar

Escriba un programa que muestre una tabla de multiplicar como la siguiente:

```
 1   2   3   4   5   6   7   8   9  10
 2   4   6   8  10  12  14  16  18  20
 3   6   9  12  15  18  21  24  27  30
 4   8  12  16  20  24  28  32  36  40
 5  10  15  20  25  30  35  40  45  50
 6  12  18  24  30  36  42  48  54  60
 7  14  21  28  35  42  49  56  63  70
 8  16  24  32  40  48  56  64  72  80
 9  18  27  36  45  54  63  72  81  90
10  20  30  40  50  60  70  80  90 100
```

4. Calcular un promedio

Escriba un programa que solicite 20 números y luego calculo el promedio y muestre los que superan el promedio. Supone decidir cómo conservar la lista y administrar la memoria.

5. Torneo deportivo[29]

Escriba un programa para simular un campeonato de tenis.
Primero, debe pedir al usuario que ingrese los nombres de ocho tenistas. A continuación, debe pedir los resultados de los partidos juntando los jugadores de dos en dos. El ganador de cada partido avanza a la ronda siguiente.
El programa debe continuar preguntando ganadores de partidos hasta que quede un único jugador, que es el campeón del torneo.

29 Ejercicio reproducido de una lista de ejercicios de la Universidad Técnica Federico Santa María (Chile).

Agradecimientos

A Álvaro Patricio Elgueta por su amable revisión del texto y comentarios.

A Claudio Salinas, presidente de INCOM, por su interés en difuncir esta obra.

Bibliografía

Ciancaglini, V. & col. (2015): *Exploring the Deep Web*, Trend Micro, https://www.trendmicro.com/cloud–content/us/pdfs/security–intelligence/white–papers/wp_below_the_surface.pdf

Colle, R. (1992): Documentación periodística, Santiago, Pontificia Universidad Católica de Chile.
- (1994): *El archivo inteligente*, Cuadernos de Información PUC n°9, http://www.cuadernos.info/index.php/CDI/article/view/299
- (1997): (e-Curso) Conceptos básicos de computación, en CD-ROM "Software educativo e Información institucional", Santiago, Pontificia Universidad Católica de Chile.
- (2000): *El análisis lógico de hechos noticiosos*, Revista Latina de Comunicación Social, nº27. http://www.revistalatinacs.org/aa2000tma/126colle.html
- (2014): Internet ayer, hoy y mañana, auto-edición, en http://issuu.com/raymondcolle/docs/universointernet
- (2017): Algoritmos, grandes datos e inteligencia en la red, Libros de Revista Mediterránea, https://www.mediterranea-comunicacion.org/article/view/cmd9-algoritmos-grandes-datos-e-inteligencia-en-la-red-Una-vision-critica/pdf
- (2017): Redes inteligentes: El poder de la comunicación, de las células a la sociedad global, Ediciones INCOM-Chile, https://goo.gl/o6kLL5
- (2017): Blockchain para periodistas y medios de comunicación, Ediciones INCOM-Chile, https://goo.gl/AEMD9A

Kaku, M. (2015): La física del futuro. Cómo la ciencia determinará el destino de la humanidad y nuestra vida cotidiana en el siglo XXII, Barcelona, Debate.

Kende, M. (2014): Global Internet Reports, Internet Society, http://www.internetsociety.org/sites/default/files/Global_Internet_Report_2014_0.pdf

Kurzweil, R. (2012): La singularidad está cerca, Lola Books (Original: "The Singularity is Near", Viking Press, 2005)

Lévy, P. (1987): La machine univers, Paris, La Découverte.

Meeker, M. (2016): *Internet Trends 2016,* Code Conference, Menlo Park (Ca), KPCB. http://www.kpcb.com/internet–trends

Sadowski, J. (2016): *Companies are making money from our personal data – but at what cost?*, The Guardian, 31/08/2016. https://goo.gl/dnbAMF

Sánchez-Migallón, S. (2015): El gran debate sobre si será posible o no una inteligencia artificial, Xataka, 18/08/2015.

Schmidt, E. y Cohen, J. (2014): El futuro digital, Madrid, Anaya (Original: "The New Digital Age", 2013).

Watts, D.J. (2006): Seis grados de separación: la ciencia de las redes en la era del acceso, Barcelona, Paidos.