Sistemas y Computación

Systems and Computing

Autor: Sofía Naranjo Pino
IS&C, Universidad Tecnológica de Pereira, Pereira, Colombia
Correo-e: sofia.naranjo@utp.edu.co

***Resumen*— Este documento presenta un resumen de los principales contenidos del programa de Ingeniería de Sistemas y Computación. En el documento se explica el sentido de las cuatro grandes temáticas que se abordan en la carrera, y se indican sus principales aplicaciones en el campo industrial e investigativo. Las áreas son: programación, redes y comunicaciones, ingeniería de software e inteligencia artificial. El docente ha realizado la primera parte: programación, dejando para el estudiante la realización de los restantes tres temas: redes, software e inteligencia artificial.**

***Palabras clave*— sistemas, redes, inteligencia artificial, software, computación, investigación, industria.**

***Abstract*— This document presents a summary of the main contents of the Computer and Systems Engineering program. The document explains the meaning of the four major themes that are addressed in the career, and indicates their main applications in the industrial and research field. The areas are: programming, networks and communications, software engineering and artificial intelligence. The teacher has done the first part: programming, leaving the student to carry out the remaining three topics: networks, software and artificial intelligence.**

***Key Word*— systems, networks, artificial intelligence, software, computing, research, industry.**

I. INTRODUCCIÓN

El Programa Ingeniería de Sistemas y Computación estudia varios campos del conocimiento ligados a la teoría de la Informática y los Sistemas en general. Se han identificado varias áreas que representan el sustento teórico y práctico de la carrera, según se ha mencionado en el resumen del documento.

El objetivo del presente documento es describir cada uno de los temas mencionados, buscando con ello brindar una visión integral de la carrera, lo cual le permitirá al estudiante elegir aquellas temáticas que mejor se adapten a sus capacidades académicas.

1.1 PROGRAMACIÓN

En [1] se define la programación de la siguiente manera: “La programación informática es el proceso por medio del cual se diseña, codifica, limpia y protege el código fuente de programas computacionales. A través de la programación se dictan los pasos a seguir para la creación del código fuente de programas informáticos. De acuerdo con ellos el código se escribe, se prueba y se perfecciona.”

Si se analiza la anterior definición, se aprecia que la programación se orienta a la solución de problemas técnicos y cotidianos a través de la escritura de un cierto código fuente, el cual debe respetar cierta estructura y método de trabajo. Para programar se debe conocer, con un buen grado de detalle, un lenguaje que se adapte al problema que se desea resolver.

Por ejemplo, si el problema a resolver es de carácter matemático, lo usual es que se emplee un lenguaje como Python, de gran acogida en los últimos tiempos. Una variante, más antigua pero igualmente importante, es el lenguaje Fortran, con el cual se desarrollaron las primeras soluciones a los problemas de Ingeniería.

Si el problema de tipo comercial, un lenguaje que se utilizó ampliamente es el lenguaje COBOL. Se dice que en la actualidad, y por un factor histórico, el 80% de las soluciones informáticas comerciales están elaboradas con este lenguaje.

Si la idea es resolver un problema de tipo general, se puede recurrir al lenguaje C, el cual se puede considerar como el padre de todos los lenguajes, pues fue utilizado en los orígenes de la computación moderna para el desarrollo del primer sistema operativo importante: UNIX.

Los lenguajes de programación se organizan según su modelo y estructura. A cada una de estas formas de organización se la conoce como: “Paradigma de Programación”.

Según [2] un paradigma de programación es:

“Un paradigma de programación es un marco conceptual, un conjunto de ideas que describe una forma de entender la construcción de programa, como tal define:

- Las herramientas conceptuales que se pueden utilizar para construir un programa (objetos, relaciones, funciones, instrucciones).
- Las formas válidas de combinarlas.

Los distintos lenguajes de programación proveen implantaciones para las herramientas conceptuales descriptas por los paradigmas. Existen lenguajes que se concentran en las ideas de un único paradigma así como hay otros que permiten la combinación de ideas provenientes de distintos paradigmas.”.

Existen muchos paradigmas de programación. Los más importantes se describen a continuación:

PARADIGMA ESTRUCTURADO

El paradigma estructurado se basa en la ejecución secuencial y ordenada de instrucciones sobre un espacio de memoria debidamente organizada. Las estructuras básicas de programación son: secuencia, decisión y ciclo. Un lenguaje clásico de la programación estructurada es el lenguaje C.

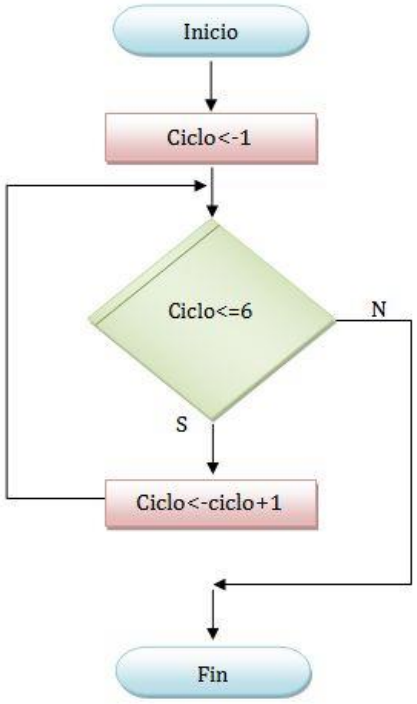

Figura 1. Paradigma estructurado

PARADIGMA DE OBJETOS

El paradigma de objetos es una concepción en la cual de definen entidades, denominadas clases, a partir de las cuales se crean objetos que interactúan entre sí. En cierto sentido, el paradigma de objetos es similar al concepto de objeto que se percibe en el mundo que nos rodea. Un lenguaje orientado a objetos es Smalltalk.

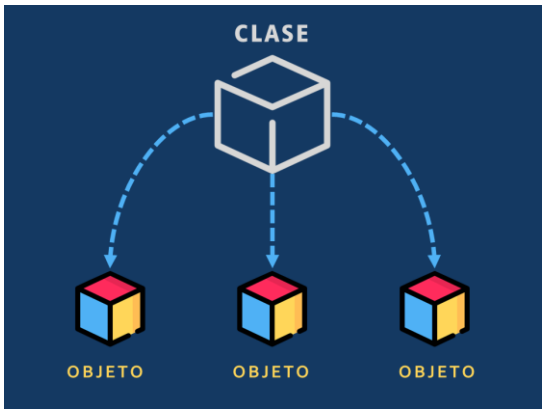

Figura 2. Paradigma orientado a objetos

PARADIGMA LÓGICO

El paradigma lógico está basado en la lógica de predicados de primer orden. Su objetivo es permitir extraer conclusiones a partir de premisas, de acuerdo con un conjunto de reglas y mecanismos de inferencia. Un lenguaje en el campo de la lógica es el PROLOG.

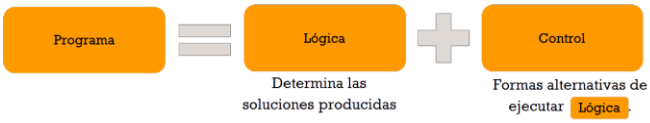

Figura 3. Paradigma lógico

PARADIGMA FUNCIONAL

El paradigma funcional se basa en la utilización de funciones como base de relación entre las partes de un programa. Una función es una porción de código que cumple un objetivo específico, permitiendo con ello simplificar y automatizar las tareas. Un lenguaje funcional es HASKELL.

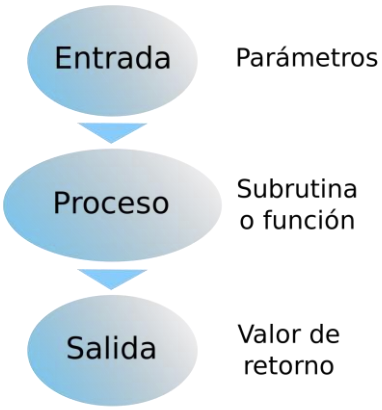

Figura 4. Paradigma funcional.

El paradigma estructurado se conoce, en ciertos entornos, como el paradigma IMPERATIVO. En la siguiente gráfica se aprecia lo visto hasta el momento:

Figura 5. Paradigmas de programación

Los paradigmas de programación, a su vez, se organizan en dos grandes categorías. La primera de ellas se conoce con el nombre de categoría IMPERATIVA. La segunda es la categoría DECLARATIVA.

La diferencia entre las dos categorías es la siguiente: en la categoría IMPERATIVA, los lenguajes de programación requieren que se indique de manera minuciosa cada uno de los pasos de la solución del problema. En este modelo se requiere realizar un seguimiento secuencial de cada paso a resolver en tal modelo.

En la categoría DECLARATIVA los lenguajes de programación no requieren de una descripción detallada y minuciosa de cada paso de la solución. Los lenguajes de tipo declarativo se caracterizan por disponer de un motor interno que les permite simplificar la ejecución de un programa. El motor le permite a los lenguajes encontrar caminos de solución que no están disponibles en el modelo imperativo.

En la siguiente gráfica se aprecia dicha clasificación.

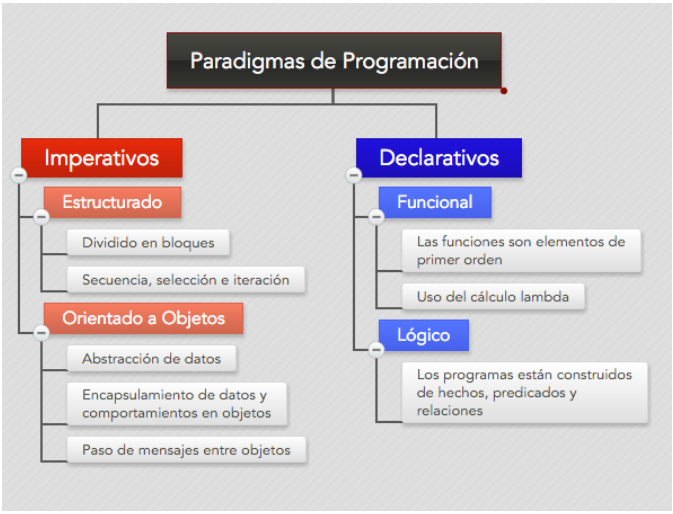

Figura 6. Lenguajes imperativos y declarativos

Por último, se presenta un gráfico que presenta los principales lenguajes de programación.

Figura 7. Lenguajes de programación.

1.2 REDES Y COMUNICACIONES

En [3] se define redes como: un conjunto de ordenadores conectados entre sí, que pueden comunicarse compartiendo datos y recursos sin importar la localización física de los distintos dispositivos. A través de una red se pueden ejecutar procesos en otro ordenador o acceder a sus ficheros, enviar mensajes, compartir programas...

Los ordenadores suelen estar conectados entre sí por cables. Pero si la red abarca una región extensa, las conexiones pueden realizarse a través de líneas telefónicas, microondas, líneas de fibra óptica e incluso satélites.

Cada dispositivo activo conectado a la red se denomina nodo. Un dispositivo activo es aquel que interviene en la comunicación de forma autónoma, sin estar controlado por otro dispositivo. Por ejemplo, determinadas impresoras son autónomas y pueden dar servicio en una red sin conectarse a un ordenador que las maneje; estas impresoras son nodos de la red.

. Dependiendo del territorio que abarca una red se clasifican en:

LAN: Local Area Network. Está constituida por un conjunto de ordenadores independientes interconectados entre sí, pueden comunicarse y compartir recursos. Abarcan una zona no demasiado grande, un edificio o un campus.

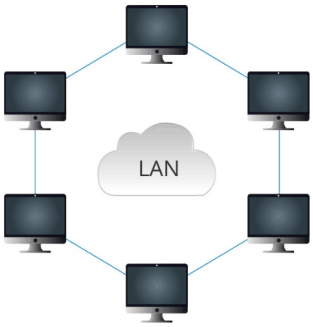

Figura 8. Red LAN

WAN: Wide Area Network, comprenden regiones más extensas que las LAN e incluso pueden abarcar varios países.

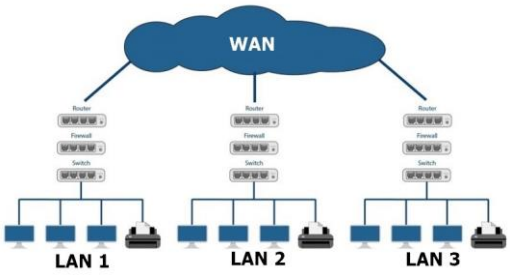

Figura 9. Red WAN

También un conjunto de redes puede conectarse entre sí dando lugar a una red mayor.

PAN: [4] Se trata de una red integrada por todos los dispositivos en el entorno local y cercano de su usuario, es decir que la componen todos los aparatos que están cerca del mismo.

Figura 10. Red PAN

1.2.1 CARACTERISTICAS DE UNA RED

De acuerdo con [5] las características de una buena red son:

1. VELOCIDAD

Es la velocidad a la que se transmiten los datos por segundo a través de la red. Suelen medirse con un test de velocidad. La rapidez de subida y descarga de datos será diferente según los estándares que utilicemos y también según el tipo de red o medio a través del que se transmiten los datos (inalámbrica, fibra óptica, cables de teléfono o coaxial).

Por ejemplo, una red inalámbrica es la mitad de rápida que una cableada (sobre 54 Mbps). Al dividirla entre todos los equipos informáticos conectados, se obtiene una cifra de Megabytes por segundo un poco inferior incluso a lo que cabría esperar debido a los protocolos de comunicación. Hay que mirar si conviene tener un sistema de cableado estructural o incluso si vendría mejor disponer de fibra óptica.

2. SEGURIDAD DE LA RED

Es uno de los aspectos más peligrosos que rodean a las redes inalámbricas, como ya hablamos en otra ocasión. La aparición de intrusos que nos quitan ancho de banda es una de las razones que convierte estas redes en bastante más vulnerables.

Por otro lado, las redes cableadas pueden sufrir interferencias como consecuencia del uso de otros aparatos como el microondas. A diferencia de estas, la fibra óptica es la que ofrece una mayor seguridad.

Figura 11. Seguridad en una red

3. CONFIABILIDAD

Mide el grado de probabilidades que existe de que uno de los nodos de la red se averíe y por tanto se produzcan fallos. En parte dependerá de la topología de la red que hayamos instalado y del lugar que ocupa el componente averiado. Cuando uno de los componentes no funciona, puede afectar al funcionamiento de toda la red o por el contrario constituir un problema local.

Por esta razón resulta determinante contar con un hardware redundante para que, en caso de fallo en uno de los componentes, haya una gran tolerancia a los errores y los demás equipos puedan seguir trabajando.

4. ESCALABILIDAD

Red de área localUna red no puede añadir nuevos componentes de forma continua y esperar que funcione a la misma velocidad. A medida que añadimos nuevos nodos y estos se hallan funcionando a la vez, la conexión a Internet se reduce, la velocidad de transmisión de datos en general es menor y hay más probabilidad de errores.

Es por eso importante ver la facilidad y las posibilidades de añadir o cambiar componentes de hardware y software o nuevos servidores para mejorar el rendimiento de la red.

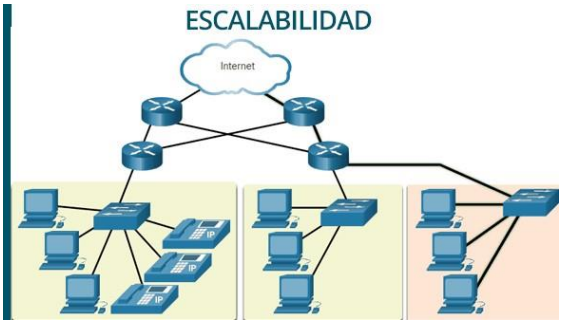

Figura 12. Escalabilidad de una red

5. DISPONIBILIDAD

Es la capacidad que posee una red para hallarse disponible y completamente activa cuando la necesitamos. Hablamos de la cantidad de tiempo posible en que podemos someter los nodos a unas condiciones de rendimiento necesarias en nuestra empresa. El objetivo es conseguir que la red se halle disponible según las necesidades de uso para las que se ha instalado.

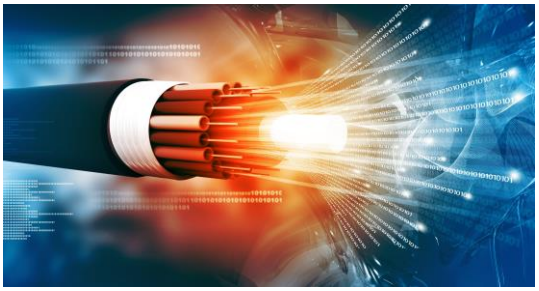

Figura 13. Disponibilidad de una red

1.3 INGENIERÍA DE SOFTWARE

Ingeniería de software es [6] una disciplina que implica el uso de estructuras, herramientas y técnicas para construir programas informáticos.

Así mismo, incluye el análisis previo de la situación, la redacción del proyecto, la creación del software y las pruebas necesarias para garantizar el correcto funcionamiento del software antes de poner el sistema en funcionamiento.

Esta ingeniería aborda todas las fases del ciclo de vida de desarrollo de cualquier tipo de sistema de información y es aplicable a una amplia gama de ámbitos de la informática y la ciencia de los ordenadores, como el diseño de compiladores, sistemas operativos y tecnologías de Intranet/Internet: La empresa, la investigación científica, la medicina, la fabricación, la logística, la banca, el control del tráfico y la meteorología son sólo algunos de los campos en los que puede trabajar.

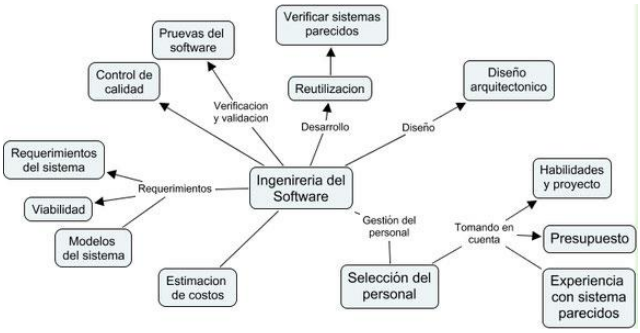

Figura 14. Mapa conceptual ingeniería en software [7]

1.3.1 OBJETIVOS DE LA INGENIERÍA DE SOFTWARE

Los objetivos de la ingeniería de software son muy diversos, pero podemos destacar los siguientes más importantes:

- Crear programas informáticos que satisfagan las necesidades de la sociedad y empresas.
- Guiar y coordinar el desarrollo de una programación difícil.
- Intervenir en el ciclo de vida de un producto.

- Estimar los costos y el plazo de ejecución de un proyecto.
- Actuar como líder del equipo de desarrollo de software.
- Diseño, desarrollo y administración de bases de datos.
- Durante la creación de la aplicación, liderar y dirigir a los programadores.
- Incluir procesos de calidad en las aplicaciones, como la medición de métricas y medidas y la evaluación de la calidad del software.

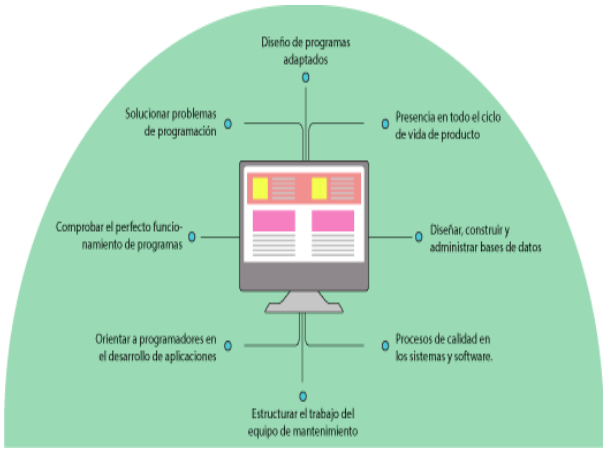

Figura 15. Objetivos de la ingeniería en software

Para nosotros en [8], este servicio trasciende a la programación, que es la base para crear una aplicación. La ingeniería de software engloba toda la gestión de un proyecto. Desde el análisis previo de la situación, el planteamiento del diseño hasta su implementación, pasando por las pruebas recurrentes para su correcto funcionamiento. Podríamos decir que la ingeniería del software es el continente donde se aloja el contenido, que sería el software en sí, resume David Souto, director general de Systems Group.

Figura 16. Programación de software

1.3.2 ETAPAS DE LA INGENIERÍA DE SOFTWARE

Dentro de la ingeniería de software entendemos que también se encuentra todo el proceso de elaboración del software, que se denomina ciclo de vida. Está formado por cuatro etapas:

- Concepción. En esta primera fase se desarrolla el modelo de negocio. Es decir, conocemos las necesidades que debe de tener un software y empezamos a buscar las herramientas para cubrirlas.
- Elaboración. Se detalla las características de la estructura del software.
- Construcción. Tal y como su nombre indica en este paso empezamos a elaborar de forma tangible todo aquello que, de momento, solo hemos plasmado en forma de ideas.
- Transición. Es el momento de la implementación y el desarrollo para los clientes o usuarios. Deben tener tiempo para familiarizarse con el nuevo software.

Una vez se realiza todo este ciclo, entramos en otra fase conocida como mantenimiento. Es una de las etapas más importantes ya que se solucionan los problemas o errores que puedan surgir durante su implementación y también su posterior puesta en marcha. Además, se incorporan actualizaciones teniendo en cuenta los requisitos del cliente con el objetivo de que puedan cumplir la mayor cantidad de tareas.

Relacionado con la ingeniería de software también se encuentra la arquitectura de sistemas. Consiste en la esquematización de la estructura general del proyecto a desarrollar. El objetivo de conocer el esqueleto del software es tener la capacidad de señalar y conocer cuáles son los componentes que son necesarios para llevar a cabo el desarrollo

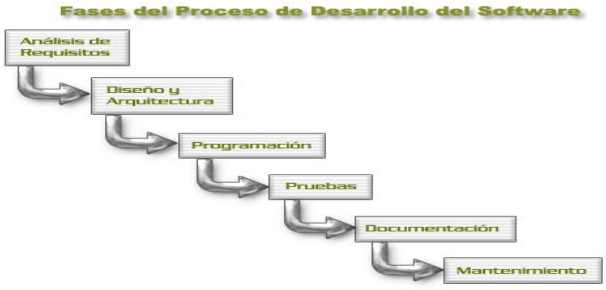

Figura 17. Fases del proceso de desarrollo de software [9]

1.3.2.1 FASES DEL PROCESO DE DESARROLLO DEL SOFTWARE

- ANÁLISIS DE REQUISITOS

  Extraer los requisitos de un producto de software es la primera etapa para crearlo. Mientras que los clientes piensan que ellos saben lo que el software tiene que hacer, se requiere de habilidad y experiencia en la ingeniería de software para reconocer requisitos incompletos, ambiguos o contradictorios. Asimismo, se define un diagrama de Entidad/Relación, en el que se plasman las principales entidades que participarán en el desarrollo del software. 830-1998 normaliza la creación de las Especificaciones de Requisitos Software.

- PROGRAMACIÓN

  Reducir un diseño a código puede ser la parte más obvia del trabajo de ingeniería de software, pero no es necesariamente la porción más larga.

- PRUEBAS

  Consiste en comprobar que el software realice correctamente las tareas indicadas en la especificación. Una técnica de prueba es probar por separado cada módulo del software, y luego probarlo de forma integral, para así llegar al objetivo. En general hay dos grandes formas de organizar un área de pruebas, la primera es que esté compuesta por personal inexperto y que desconozca el tema de pruebas, de esta forma se evalúa que la documentación entregada sea de calidad, que los procesos descritos son tan claros que cualquiera puede entenderlos y el software hace las cosas tal y como están descritas.

- MANTENIMIENTO

Mantener y mejorar el software para enfrentar errores descubiertos y nuevos requisitos. Esto puede llevar más tiempo incluso que el desarrollo inicial del software. Alrededor de 2/3 de toda la ingeniería de software tiene que ver con dar mantenimiento.

1.4 INTELIGENCIA ARTIFICIAL

Según [10] la inteligencia artificial (IA) hace posible que las máquinas aprendan de la experiencia, se ajusten a nuevas aportaciones y realicen tareas como seres humanos. La mayoría de los ejemplos de inteligencia artificial sobre los que oye hablar hoy día – desde computadoras que juegan ajedrez hasta automóviles de conducción autónoma – recurren mayormente al aprendizaje profundo y al procesamiento del lenguaje natural. Empleando estas tecnologías, las computadoras pueden ser entrenadas para realizar tareas específicas procesando grandes cantidades de datos y reconociendo patrones en los datos.

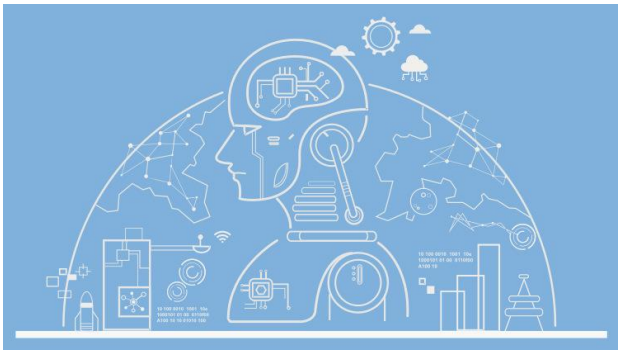

Figura 18. Inteligencia artificial

1.4.1 TIPOS DE INTELIGENCIA ARTIFICIAL

En [11] los expertos en ciencias de la computación Stuart Russell y Peter Norvig diferencian varios tipos de inteligencia artificial:

- Sistemas que piensan como humanos: Automatizan actividades como la toma de decisiones, la resolución de problemas y el aprendizaje. Un ejemplo son las redes neuronales artificiales.

- Sistemas que actúan como humanos: Se trata de computadoras que realizan tareas de forma similar a como lo hacen las personas. Es el caso de los robots.

- Sistemas que piensan racionalmente: Intentan emular el pensamiento lógico racional de los humanos, es decir, se investiga cómo lograr que las máquinas puedan percibir, razonar y actuar en consecuencia. Los sistemas expertos se engloban en este grupo.

- Sistemas que actúan racionalmente: idealmente, son aquellos que tratan de imitar de manera racional el comportamiento humano, como los agentes inteligentes.

1.4.2 APLICACIONES PRÁCTICAS DE LA INTELIGENCIA ARTIFICIAL

Con la IA en [12] se pretende transformar casi todos los aspectos de la vida y la economía. Como ejemplo tenemos:

- SALUD: Los investigadores estudian cómo usar la IA para analizar grandes cantidades de datos sobre la salud para encontrar patrones que podrían llevar a nuevos descubrimientos en la medicina y a otras formas de mejorar los diagnósticos individuales.

Figura 19. IA en la salud [13]

- TRANSPORTE: La inteligencia artificial podría mejorar la seguridad, velocidad y eficiencia del tráfico ferroviario al minimizar la fricción de las ruedas, maximizar la velocidad y permitir la conducción autónoma.

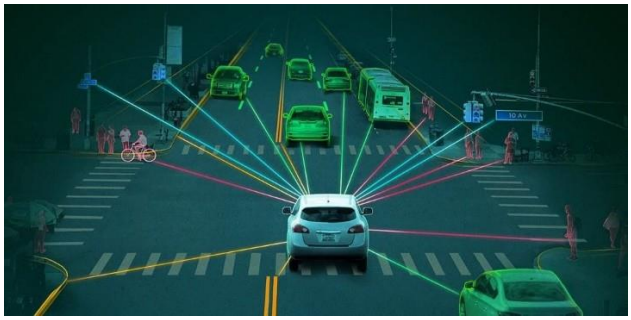

Figura 20. IA en el transporte [14]

- MANUFACTURAS: La inteligencia artificial puede ayudar a que los productores europeos sean más eficientes y potencie de nuevo las fábricas en Europa al usar robots, optimizar los recorridos de ventas o con predicciones puntuales del mantenimiento necesario o de averías en «fábricas inteligentes».

  Desde hace ya unos meses según información de [15], la inteligencia artificial ha venido transformando el sector manufacturero, el cual está liderando el camino en la aplicación de esta

tecnología. Los importantes recortes en el tiempo de inactividad no planificado y los productos mejor diseñados es lo que motiva principalmente a los fabricantes a usar y aplicar analíticas basadas en IA a los datos para mejorar la eficiencia, la calidad del producto y la seguridad de los empleados.

Figura 21. IA en manufactura

- COMIDA Y AGRICULTURA: La IA puede usarse para construir un sistema alimentario sostenible: podría garantizar comida más sana al minimizar el uso de fertilizantes, pesticidas y el riego; mejorar la productividad y reducir el impacto medioambiental.

  Las aplicaciones más relevantes de la IA en la agricultura las podemos clasificar en tres categorías principales en base a [16]:

  Robots: las empresas están desarrollando y programando robots autónomos para ejecutar labores agrícolas básicas como siembra, cosecha, control de malezas y pulverización. (Lea: La inteligencia artificial al servicio de la prevención de hambrunas)

  Monitoreo de cultivos y suelos: mediante la visión por dispositivos electrónicos y algoritmos de aprendizaje para procesar datos capturados por drones y / o tecnología basada en software, es viable monitorear la sanidad de los cultivos y el suelo.

  Análisis predictivo: con modelos de aprendizaje automático para monitorear y predecir impactos de las condiciones ambientales sobre el desempeño y el rendimiento de los cultivos

Figura 22. IA en la agricultura

1.4.3 ORIGEN DE LA INTELIGENCIA ARTIFICIAL

En [17] se expone que desde al menos el siglo I a.C., los humanos se han planteado la posibilidad de crear máquinas que imiten al cerebro humano. Ya en la época moderna, John McCarthy acuñó el término «inteligencia artificial» en 1955. En 1956, McCarthy y algunos otros organizaron una conferencia denominada «Dartmouth Summer Research Project on Artificial Intelligence». Este encuentro dio lugar a la creación del aprendizaje automático, el aprendizaje profundo, el análisis predictivo y, ahora, el análisis prescriptivo. También dio lugar a un campo de estudio totalmente nuevo: la ciencia de los datos.

REFERENCIAS

Referencias en la Web:

[1] https://conceptodefinicion.de/programacion-informatica/

[2] https://wiki.uqbar.org/wiki/articles/paradigma-de-programacion.html#:~:text=Un%20paradigma%20de%20programaci%C3%B3n%20es,relaciones%2C%20funciones%2C%20instrucciones).

[3] https://www.monografias.com/trabajos58/redes-comunicaciones/redes-comunicaciones.shtml

[4] https://redfibra.mx/tipos-de-redes-informaticas-que-es-una-red-lan-wan-man-wlan-wman-wwman-san-pan/

[5] https://www.gadae.com/blog/5-caracteristicas-red-informatica/

[6] https://mexico.unir.net/ingenieria/noticias/ingenieria-de-software-que-es-objetivos/

[7] https://lopezvictor01.jimdofree.com/ingenier%C3%ADa-de-software/foros/

[8] https://systemsgroup.es/tecnologias-de-la-informacion/la-ingenieria-de-software-que-es-y-que-utilidad-tiene/32363/

[9] https://sistemasvd.wordpress.com/2008/07/05/fases-del-proceso-de-desarrollo-del-software/

[10] https://www.sas.com/es_co/insights/analytics/what-is-artificial-intelligence.html

[11] https://www.iberdrola.com/innovacion/que-es-inteligencia-artificial

[12] https://www.europarl.europa.eu/news/es/headlines/society/20200827STO85804/que-es-la-inteligencia-artificial-y-como-se-usa

[13] https://www.iic.uam.es/lasalud/como-acercar-inteligencia-artificial-al-sector-salud/

[14] https://www.techstreet.com/

[15] https://www.telcel.com/empresas/tendencias/notas/inteligencia-artificial-transforma-manufactura

[16] https://www.agronet.gov.co/Noticias/Paginas/La-inteligencia-artificial-al-servicio-de-la-agricultura.aspx

[17] https://www.netapp.com/es/artificial-intelligence/what-is-artificial-intelligence/