4. MARCO TEÓRICO

4. Marco teórico

4.1 Inteligencia Artificial

En este capítulo se desarrollará el concepto de Inteligencia Artificial (IA) y las diferentes ramas que de esta se desprenden, en particular se dará énfasis al reconocimiento de patrones. En el (Apéndice I) se encuentra de forma sucinta la historia de la IA.

Para comenzar se definirá el término *inteligencia*, según la Real Academia de la Lengua Española (DRAE, 2010, s/p) la define como: conocimiento, comprensión, habilidad, destreza y experiencia. En otras palabras, la inteligencia, es la capacidad de resolver problemas.

Esta capacidad se ha tratado de imitar de manera artificial con la construcción de máquinas, para este propósito, se creó la rama de las Ciencias de la Computación denominada *Inteligencia Artificial (IA)*, la cual "consiste en utilizar métodos basados en el comportamiento inteligente de los seres humanos y otros animales para resolver problemas complejos" (Coppin, 2004, pág.4).

Por tanto se concluye que la IA busca entender el comportamiento de los seres vivos para reproducir de manera artificial las habilidades y destrezas que estos puedan tener para resolver problemas.

4.1.1 Ramas de la Inteligencia Artificial

Las disciplinas que han dado dirección a la IA según (Russell y Norving, 2010, Págs.5-14) son:

- *La filosofía*: la cual trata de resolver preguntas tales como: ¿Qué es la mente y cómo funciona?.
- *Las matemáticas*: donde mediante demostraciones y teoremas sientan las bases científicas de tres áreas básicas para la IA: la lógica, la probabilidad y la computación.
- *La neurociencia*: encargada de estudiar el funcionamiento del sistema nervioso enfocándose especialmente en el cerebro.
- *La psicología*: ocupada de estudiar el comportamiento de animales y humanos.

- *La ingeniería en computación*: la cual enfoca sus esfuerzos a la construcción eficiente de computadoras.

Estas disciplinas han contribuido de manera directa en la IA con ideas, conocimiento y técnicas.

Como se observa la IA es multidisciplinaria, la cual gracias a su amplia diversidad de conocimiento, según (McCarthy, 2007, s/p) y (Bourbakis, 1992, págs. 432-433) puede dividirse en las siguientes ramas:

- *Programación genética*: es un método sistemático que sirve para que las computadoras puedan desarrollar de manera automática sus propios programas para resolver problemas (Koza y cols., 2005, pág. 1).
- *Robótica*: Rama encargada del estudio de los robots, se ocupa de su diseño, manipulación y aplicación (Ruiz-Velasco, 2007, pág. 90).
- *Lógica*: en la IA se utiliza para que un programa pueda deducir que opciones son adecuadas para lograr un objetivo (McCarthy, 2007, s/p).
- *Inferencia*: resuelve problemas a partir de una base de datos y declaraciones establecidas (McCarthy, 2007, s/p).
- *Procesamiento del Lenguaje Natural*: es el encargado del entendimiento entre las computadoras y los seres humanos por medio del lenguaje natural (Dubitzky y Azuaje, 2004, pág. 147).
- *Planeación*: programas que se encargan a partir de la obtención de datos particulares de trazar un camino para llegar a una meta (McCarthy, 2007, s/p).
- *Búsqueda de información*: la IA resuelve problemas a partir de la obtención y tratamiento de datos, para lograr este objetivo se ayuda de la heurística, epistemología y la ontología (McCarthy, 2007, s/p).
- *Redes Neuronales (RN)*: esta rama de la IA trata de obtener máquinas que a partir de la experiencia puedan obtener conocimiento, en otras palabras las RN (Flórez y Fernández, 2008, pág. 11) “tratan de emular el comportamiento del cerebro humano”.
- *Reconocimiento de patrones*: es la disciplina encarga de la clasificación de objetos mediante la obtención de imágenes o señales, la visión por computadora es un área importante dentro del reconocimiento de patrones (Theodoridis y Koutroumbas, 2006, pág. 1), este punto será ampliado en la siguiente sección.

- *Sistemas expertos:* son programas de computó que almacenan todos los datos que pueden existir sobre un tema tal y como lo haría un experto humano en un área, con el objetivo de tomar decisiones con base a información detallada sobre un problema para obtener una solución (Hellriegel y Jackson, 2006, pág. 241).

En la (Fig. 1) se resumen las diferentes ramas que surgen de la IA (Bourbakis, 1992, pág. 432):

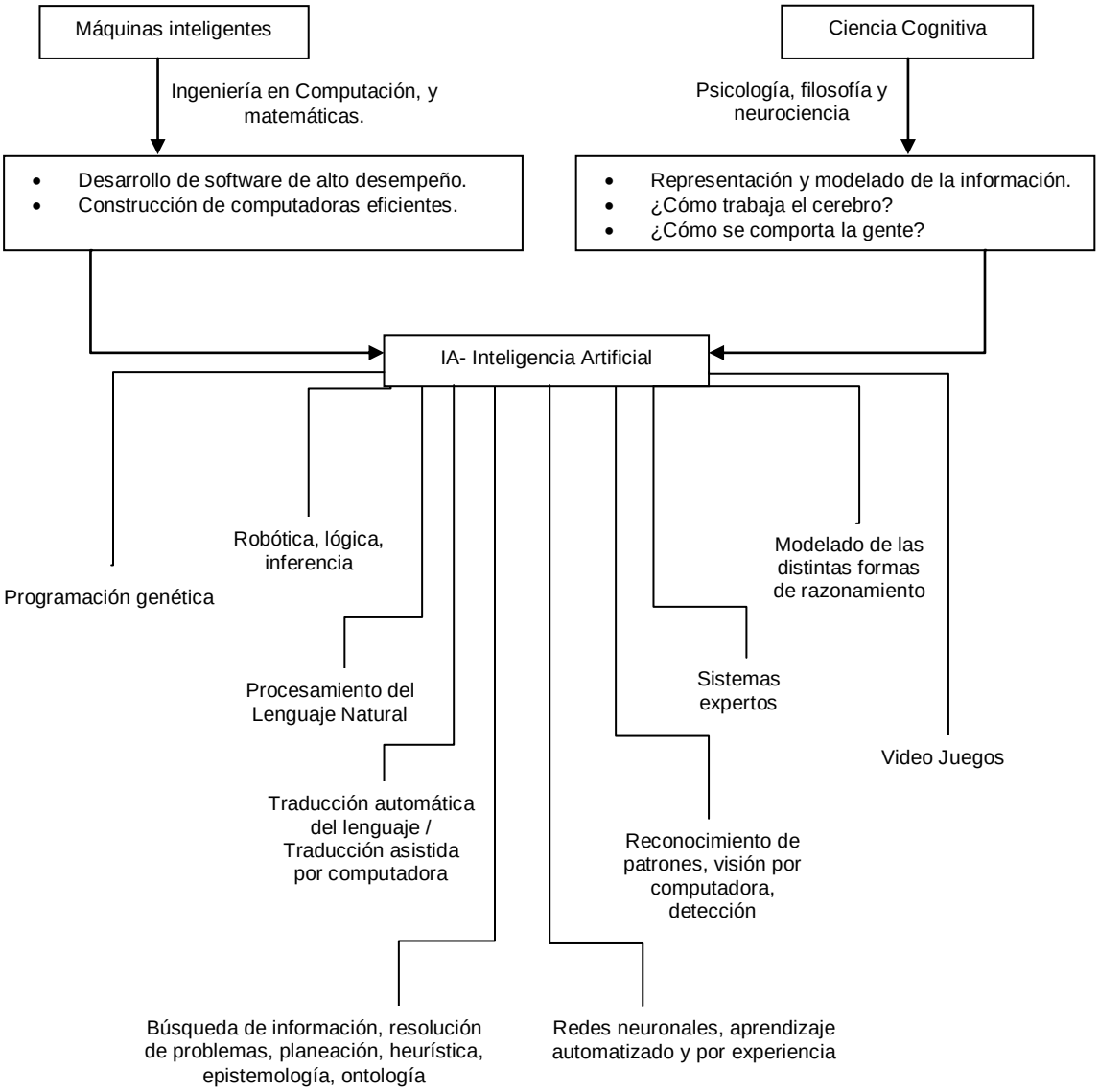


Fig. 1. Ramas de la Inteligencia Artificial

4.1.2 Reconocimiento de Patrones

El *Reconocimiento de Patrones (RP)* es una rama esencial de la IA, como se mencionó anteriormente, el RP se define como la disciplina encargada de la *clasificación* de objetos en *categorías* o *clases*, donde dependiendo de la aplicación utilizada pueden obtenerse imágenes o señales, de las cuales se *extraen características* específicas (*patrones)* para identificar a los objetos (Theodoridis y Koutroumbas, 2006, pág. 1).
Enseguida se listan los conceptos básicos del RP:

- *Patrón*: es la descripción estructural o cualitativa de un objeto que está formado por uno más descriptores (Romero, 2008, pág. 9).
- *Clase*: es un conjunto de entidades que comparten algunas características que las diferencia de otras (Alba y Cid, 2006, pág. 4).
- *Extracción de características*: subsistema que extrae información relevante para la clasificación (Alba y Cid, 2006, pág. 4).

4.1.2.1 Enfoques del Reconocimiento de Patrones

Según (Carrasco, 2003, págs. 2-3) el RP presenta los siguientes enfoques:

- *Reconocimiento Estadístico de Patrones*: enfoque basado en la probabilidad y la estadística en donde a partir de mediciones numéricas y a partir de distribuciones probabilísticas se realiza el reconocimiento.
- *Reconocimiento Sintáctico de Patrones*: se basa en la materia que estudia lenguajes formales, su objetivo es crear una gramática que describan la estructura básica de un objeto.
- *Redes Neuronales*: se concentran en la construcción de neuronas interconectadas entre sí que pueden ser entrenadas para que por medio de estímulos se obtenga una respuesta.
- *Reconocimiento Lógico Combinatorio de Patrones*: este enfoque trata de definir modelos lo más cercano posibles a la realidad, no se basa en suposiciones.

4.1.2.2 Categorías del Reconocimiento de Patrones

El RP se divide en: *RP supervisado*, utilizado cuando se tiene un conjunto de clases previamente analizadas y *RP no supervisado* cuando por el contrario no se conocen las clases a las que pertenece el objeto que quiere ser identificado (Sankar y Pal, 2001, pág. 3-4).

El presente trabajo utilizará el RP supervisado, en seguida se muestra un ejemplo:

Uno de los objetivos de la IA es crear máquinas automatizadas con algoritmos que permitan realizar tareas repetitivas que se basan en la toma de decisiones, por ejemplo, la toma de asistencia del personal en una empresa por medio de su huella dactilar.

El proceso que deben seguir estas máquinas es el siguiente (Romero, 2008, 9):

1. *Se define el universo de trabajo:* se realiza una segmentación del objeto que se desea identificar.
2. *Se realiza la extracción de características (patrones):* se obtienen una serie de características del objeto los cuales son transformadas en un vector numérico.
3. *Se obtiene el vector de características*: el vector es comparado con vectores prototipo almacenados en una base de datos preestablecida.
4. *Clasificación del objeto*: según la semejanza hallada con los vectores prototipo de la base de datos con los que fue comparado se determina si existe o no dentro de la base de datos.

De esta manera en general trabajan los dispositivos que realizan reconocimiento de patrones supervisado.

4.1.2.3 Tipos de Patrones

Enseguida se describirán los distintos tipos de patrones que pueden estar presentes en un objeto:

- *Patrones vectoriales:* este tipo de patrones se caracterizan por reconocer objetos por medio de la obtención de sus características más importantes para luego compararlos con una serie de grupos que contienen descriptores específicos. Por ejemplo, se tiene una planta y se desea saber de qué tipo es, para conocer el tipo

de planta, primero se deben extraer sus características más importantes tales como: altura, color, olor, tipo de hoja, etc. Posteriormente estas características son buscadas dentro de una base de datos predefinida la cual contiene descriptores específicos, en este caso puede tener tablas como las siguientes: tabla_altura (chica , mediana, grande, etc.), tabla_clase_a_la_que_pertenece(tóxica, medicinal comestible, etc.), tabla_color(rojo, verde, azul, etc.), tipo de olor (suave, dulce, etc.), conforme se realiza la búsqueda se irá discriminando y descartando las opciones que no se han correctas, al finalizar la comparación con los descriptores específicos del objeto se llegará a una conclusión y se podrá determinar de qué tipo de planta se está hablando (Alba y Cid, 2006, pág. 4).

- *Patrones estructurados (cadenas):* un ejemplo para este tipo de patrones es el reconocimiento de huellas dactilares, los descriptores son codificados mediante relaciones espaciales entre los componentes del objeto, los patrones que hacen que una huella dactilar sea única no son las crestas o los valles, sino las *minucias* que son algunos puntos anormales en las crestas de una huella (Fig. 2) (Flores y Vargas-Machuca, 2006, s/p).

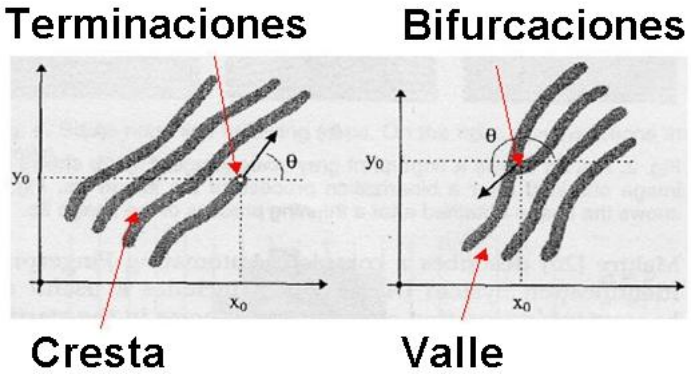


Fig. 2. Minucias

Existen principalmente 8 puntos característicos dentro de una huella digital, estos se pueden repetir de manera combinada entre 60 y 120 veces (Fig. 3).

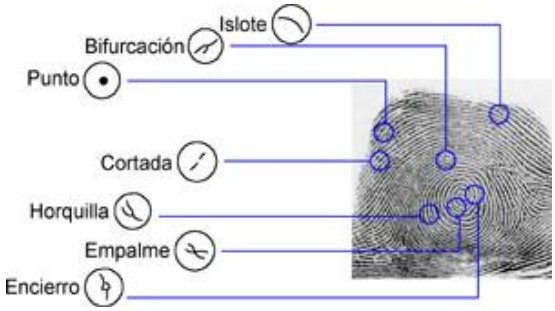


Fig. 3 Principales minucias en una huella dactilar

Para obtener los patrones de una huella dactilar puede ser utilizado un AFRS (Automatic Fingerprint Recognition System), como el que se utilizará en el presente trabajo para el desarrollo de la aplicación. El procedimiento que utiliza es el siguiente (INEGI, 2002, s/p):

- *Paso1:* El dedo es leído por un escáner de huellas. El software del biométrico crea un modelo de la huella en dos dimensiones (Fig. 4).

Fig. 4 Huella dactilar leída por el escáner

- *Paso 2***:** La huella es codificada por el escáner (Fig. 5). Son detectadas las minucias (*patrones)*.

Fig. 5 Identificación de minucias

- *Paso 3:* Es creada una plantilla donde la ubicación de cada punto característico se representa por una combinación de puntos (x, y) dentro de un plano cartesiano, los cuales son utilizados para crear un conjunto de *cadenas* (vectores) que se obtienen al unir las minucias entre sí mediante rectas cuyo ángulo y dirección generan un trazo de configuración única e irrepetible (Fig. 6).

Fig. 6 Plantilla de la huella dactilar

- *Paso 4:* El software guarda y reconoce un conjunto de números que solo podrán ser reconocidos como una plantilla (Fig. 7).

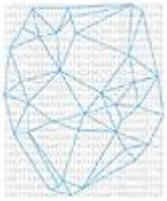


Fig. 7 Conjunto de números únicos que identifican a una huella dactilar

- *Patrones en árbol*: en este tipo de patrones es común aplicar los árboles de búsqueda, por ejemplo en una fotografía digital se pueden ver los distintos tipos de edificios los cuales se pueden agrupar de la siguiente manera (Fig. 8):

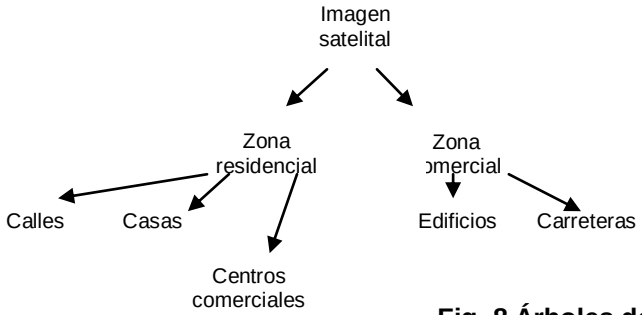


Fig. 8 Árboles de búsqueda

4.1.2.4 Identificación de patrones

Cuando se desea utilizar el reconocimiento de patrones para identificar objetos primero se debe identificar el tipo de patrones que se obtendrán, para identificarlos y clasificarlos de manera correcta existen dos estrategias:

1. W*rapper:* en donde las características son elegidas por el mecanismo clasificador, un *wrapper* puede ser un procedimiento para la extracción de información (patrones) de un objeto, este tipo de estrategias es frecuentemente utilizada en internet, por ejemplo, dentro de una página web en donde se venden diversos productos, este tipo de procedimientos se utilizan para saber cuáles son las preferencias del cliente, siguen sus *patrones de compra*, para que cuando el usuario regrese, la página de manera automática pueda ofrecer las novedades que al consumidor le interesan.
2. *Filter:* elige los descriptores de manera independiente al clasificador, esta estrategia se puede aplicar por medio de:

    - *Tablas de decisión:* las cuales se utilizan para crear subconjuntos mínimos de variables que permitan eliminar confusión entre las clases.
    - *ID3*: es creado un árbol de decisión y se selecciona un conjunto de variables que permitan discriminar entre clases.
    - *Teoría de testore:* se buscan todos los subconjuntos de variables discriminantes, con esto se evalúa cada variable y se seleccionan aquellas con mayor relevancia.

De esta manera trabaja el reconocimiento de patrones en la IA para la identificación de objetos dentro de un universo establecido.

4.2. Interfaces humano-máquina (Biometría)

Una actividad común dentro de las empresas e instituciones es la identificación de personal, ya sea para registrar su asistencia o para que el usuario pueda ingresar a la información que le pertenece en su estación de trabajo, estas actividades se llevan a cabo

con el fin tener un control en los horarios de entrada y salida que debe cumplir el personal o para llevar un registro de quien puede o no consultar cierta información.
Los métodos más utilizados para identificar a una persona son:

- El uso de credenciales.
- La firma del individuo en una tarjeta.
- Por el registro con un nombre de usuario y una contraseña en un sistema computacional.

Los anteriores son métodos tradicionales, pero ¿qué pasa si un usuario pierde su credencial o si alguien más falsifica la firma de un empleado para registrar su acceso o peor aun si al personal le es robado su nombre de usuario y contraseña?, esto causa un gran problema para el control de la empresa o institución, sobre todo si se requiere llevar un registro estricto sobre las actividades del personal y sobre la información que se maneja en la organización, por tanto el inconveniente de los métodos mencionados es que no se puede discriminar entre los individuos legítimos y los individuos impostores.
Para solucionar este problema existe una ciencia denominada *Biometría* de la cual se hablará en esta sección, se explicará para que sirve, como ha evolucionado dentro de la sociedad (ver Apéndice II), que tipo de biométricos existen y como funciona dentro del procesamiento digital de imágenes.

4.2.1 Definición de Biometría

Biometría, la palabra viene del griego *bios* vida y *metría* medida, "es la ciencia que estudia las características físicas, químicas y conductuales de un individuo", para que este pueda ser identificado (Jain y Flynn, 2008, pág. 1).
Los rasgos para reconocer a una persona se dividen en (Simón, 2003, pág. 11):

- *Fisiológicos*: huella dactilar, iris, retina, la geometría de la mano, cara, estructura de las venas o de los poros (de alguna parte del cuerpo).
- *Químicos*: olor y ADN.
- *Conductuales*: voz, escritura, la firma escrita, el modo de andar o de pulsar un teclado.

Por tanto la biometría es una ciencia que se dedica a identificar a una persona de otra por medio de patrones únicos e irrepetibles, en seguida se presentan dos definiciones que serán de utilidad para comprender de una mejor manera como trabaja la biometría:

*Biometría Informática***:** automatización de los procesos biométricos, basado en técnicas matemáticas auxiliadas por computadoras (Hernández, 2009, pág. 2).

Sistema Biométrico: Es esencialmente un sistema de reconocimiento de patrones, este sistema lo podemos dividir en cuatro módulos (Jain y Flynn, 2008, pág. 3-5):

1. *Módulo de escaneo*: Se requiere de un lector o escáner de imágenes para obtener las características del usuario. Por ejemplo para obtener la imagen de una huella dactilar se requiere de un lector de huellas. En la mayoría de los casos se requiere de un escáner que obtenga imágenes en 2D o en 3D. Existen dos excepciones en las que no se puede utilizar un escáner: el reconocimiento de voz, donde se requiere algún dispositivo para grabar audio; y el reconocimiento por medio del olor, donde se requiere el uso de la química para su identificación.
2. *Módulo de evaluación de calidad y extracción de características*: La calidad de los datos obtenidos por el sensor son evaluados en este módulo, los algoritmos que se tengan ayudan a determinar si la información obtenida es suficientemente clara para la extracción de patrones. Por ejemplo, la posición y orientación de las minucias en una huella dactilar ayudan a crear la plantilla que identificará a una persona, por tanto, los datos obtenidos por el escáner o lector de huellas debe ser claros, en caso contrario se debe de realizar nuevamente la toma de datos.
3. *Módulo de base de datos*: Es el módulo encargado de almacenar los datos obtenidos durante el proceso de registro de patrones, la captura de datos puede ser supervisada por una persona o por una máquina dependiendo de la aplicación utilizada, la base de datos puede contener el identificador de usuario y datos referentes a este, por ejemplo los escáneres de huellas dactilares crean un vector numérico único por medio una plantilla obtenida con las minucias extraídas del dedo escaneado, este vector es guardado en la base de datos y sirve como referencia para cuando el usuario desea identificarse.
4. *Módulo de comparación y toma de decisiones*: La características obtenidas de un individuo serán comparadas con datos previamente registrados en una base datos para identificar si se trata de la persona o no. En el caso de un sistema biométrico basado en el reconocimiento de huellas dactilares, al obtener un cierto

número de minucias de una huella estas se comparan con el vector creado durante la extracción y registro de datos, se realiza una evaluación y si las minucias obtenidas coinciden con ciertas parte de la plantilla se puede decir que la persona es quien dice ser.

Estos son los pasos generales que debe seguir un sistema biométrico para su correcto funcionamiento.

4.2.2 Tipos de Biométricos

La biometría estudia las características de una persona para que esta pueda ser identificada, para lograr su objetivo, esta ciencia se divide en *Biometría Estática* y *Biometría Dinámica* (Hernández, 2009, pág. 23).

La *Biometría Estática* se dedica al estudio de las características fisiológicas y químicas que puede tener un individuo para ser identificado, por otra lado, la *Biometría Dinámica* desarrolla sus estudios en el comportamiento de los seres humanos para determinar que los hace únicos de los demás.

En el siguiente diagrama se pueden observar las principales ramas que estudia la biometría (Fig. 9):

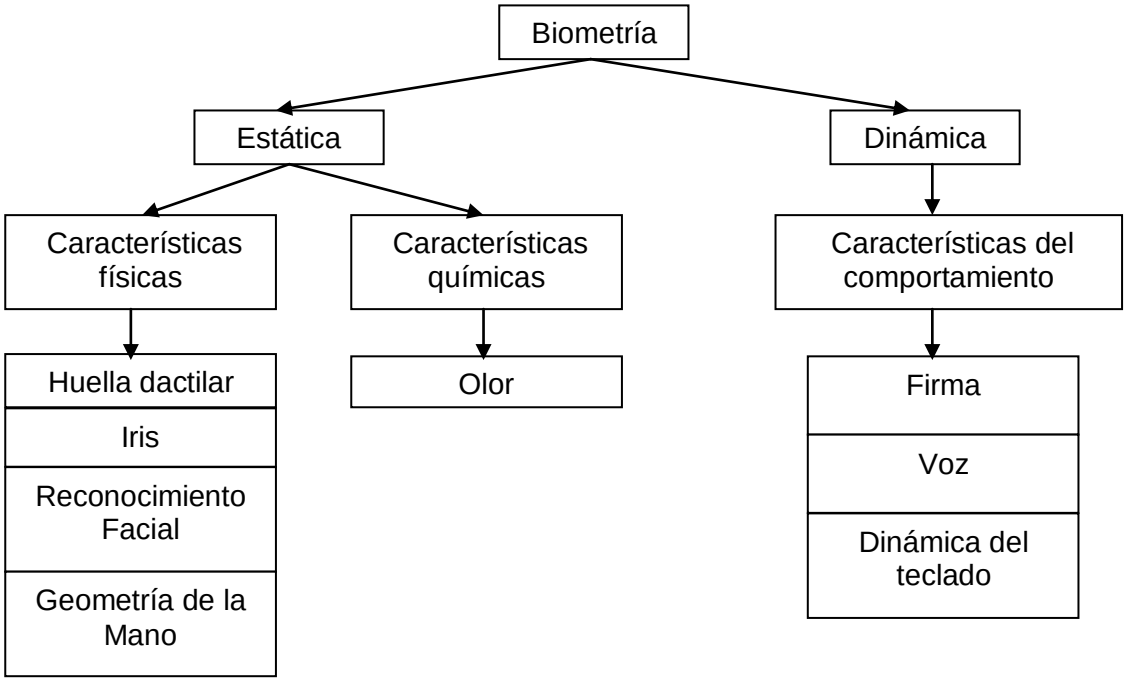


Fig. 9 Ramas de la Biometría

Para fines de este trabajo, de las ramas que estudia la biometría, la investigación se centrará en el reconocimiento de huellas dactilares y se realizará una breve explicación de las demás áreas.

4.2.2.1 Reconocimiento de huellas dactilares

Es uno de los métodos más utilizados por su facilidad de uso y por el gran prestigio que tiene entre los usuarios. Gracias a sus características las huellas dactilares se puede obtener de manera rápida y eficaz para realizar el reconocimiento de un individuo. En la actualidad este tipo de métodos se han automatizado por medio de los denominados *sistemas biométricos*, esto ha ocasionado la mejora en el proceso de obtención de impresiones digitales y una mayor eficiencia y velocidad a la hora identificar a una persona.

Las huellas dactilares son características que distinguen a los seres humanos de manera única. La ciencia que se dedica a estudiar este rasgo es la *dactiloscopia*, palabra derivada del griego *daktilos* (dedos) y *skopein* (examen o estudio) (Hernández, 2009, pág.24), su objetivo es observar y clasificar los dibujos digitales de una huella dactilar con el fin de identificar a una persona.

Los sistemas dactiloscópicos están basados en cuatro principios (Dactiloscopia México, 2003, s/p):

- *Perennidad*: reconoce que desde los seis meses de gestación ya se han creado las huellas dactilares, mismas que permanecen hasta la muerte del individuo.
- *Inmutabilidad*: las huellas dactilares no son modificadas durante el desarrollo físico de una persona y no pueden ser afectadas por una enfermedad, en caso de que las huellas sean afectadas por un desgaste involuntario estas tienen la capacidad de regenerarse tomando su forma original en un periodo de 15 días.
- *Variedad:* no hay huella parecida a otra, son únicas e irrepetibles por su gran riqueza en la combinación de minucias. Cada huella es individual ya que no se encuentra ligada genéticamente y contiene más de 20 puntos característicos.
- *Clasificabilidad*: formación de bases de datos de consulta de las diferentes plantillas de huellas digitales que pueden haber sido obtenidas para fines de control de acceso.

Para aplicar esta ciencia se deben estudiar las diferentes características que tiene la piel, dentro de las cuales encontramos: las *papilas*, son pequeñas protuberancias que nacen de la dermis y sobresalen de la epidermis pueden tener formas muy diversas; las *crestas*, son bordes en la piel que están formados por una sucesión de papilas, las cuales forman una infinidad de figuras en las yemas de los dedos, son más anchas en la base que su cúspide y reciben el nombre de crestas papilares; los *surcos*, espacios hundidos que se encuentran entre papilas; y los *poros* son orificios que se encuentran ubicados en la cúspide de las crestas papilares, tienen la función de segregar sudor (Hernández, 2009, pág. 25).

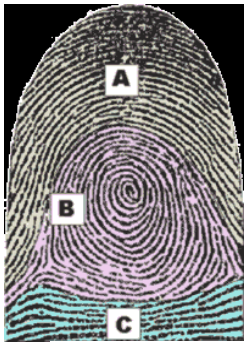
El conjunto de crestas papilares correspondientes a cada dedo es denominado *dactilograma* (Fig. 10). Existen tres tipos de dactilogramas:

- *Natural*: El que existe en la yema de los dedos.
- *Artificial*: Es el dibujo impreso por cada dedo después de entintarlo.
- *Latente*: producido por un dedo en virtud de un contacto con cualquier superficie.

Fig. 10 Dactilograma

Los dactilogramas son clasificados de diferentes maneras. Para realizar la lista de características de una huella dactilar se deben tomar en cuenta los siguientes aspectos (Ministerio de Seguridad Argentino, 2010, s/p):

- Cada dactilograma está compuesto por tres zonas fundamentales (Fig. 11):



A. Zona marginal.
B. Zona nuclear.
C. Zona bacilar.

Fig. 11 Zonas Fundamentales

- Para identificar las tres zonas fundamentales se deben tomar en cuenta las siguientes características (Fig. 12):

  a) Dentro del dactilograma se debe encontrar una delta (color verde Fig. 12), las deltas se pueden dividir en negros o salientes y en blancos o hundidos, los negros se dividen en cortos o largos y los blancos en cerrados o abiertos, los negros siempre están unidos los blancos no.
  b) A partir de la delta podemos identificar una directriz la cual encierra la zona del núcleo (color azul Fig. 12).
  c) El núcleo es la parte más importante del dactilograma ya que a partir de este se pueden distinguir las características fundamentales para clasificar una huella dactilar (color rojo Fig. 12).

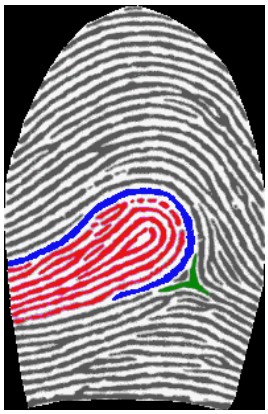


Fig. 12 Características para identificar zonas fundamentales

Enseguida se muestran los tipos de delta que pueden ser encontrados en una huella dactilar (Fig. 13):

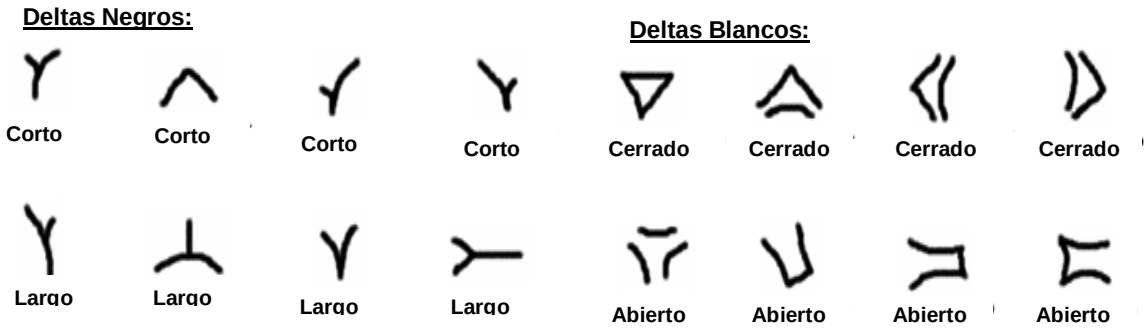


Fig.13 Tipos de deltas

Como ya se mencionó el núcleo es la parte más importante dentro de un dactilograma pues en base a este y según la clasificación que realizó Edwar Henry apoyado en el sistema de Galton en 1990, las huellas dactilares se pueden clasificar en cinco figuras fundamentales (Maltoni & Co., 2003, pág. 236):

- *Arco:* este dactilograma se caracteriza por no tener deltas en su dibujo y sus crestas corren de un lado a otro sin volver en si mismas (Fig. 14).

Fig. 14 Arco

- *Arco en "forma de casa de campaña":* este tipo de huella es similar al arco normal, la diferencia es que este cuenta con un delta y con una curva que toma la forma de una casa de campaña (Fig. 15).

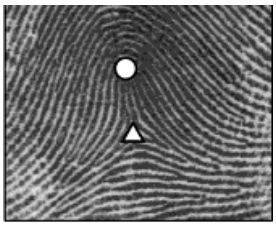


Fig. 15 Arco en forma de campana

- *Presilla interna*: se caracteriza por tener un delta a la derecha del observador, las crestas que forman el núcleo nacen a la izquierda y corren hacia la derecha dando vuelta sobre sí mismas, para salir al mismo lado de salida (Fig. 16).

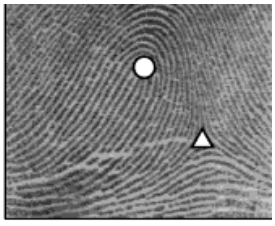


Fig. 16 Presilla interna

- *Presilla externa*: Al igual que las presillas internas, cuentan con un punto delta, pero éste se ubica del lado izquierdo del observador. Las crestas papilares que forman el núcleo nacen a la derecha y su recorrido es a la izquierda para dar vuelta sobre sí mismas y regresar al mismo punto de partida (Fig. 17).

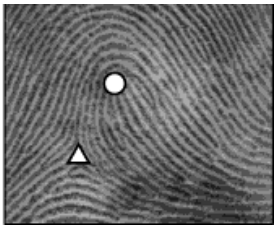


Fig. 17 Presilla externa

- *Verticilo*: Se denomina verticilo debido a que sus dibujos en muchos casos son similares a las flores; su característica más importante es que cuenta con dos puntos delta, uno del lado derecho y otro del lado izquierdo, su núcleo puede adoptar formas circulares, elípticas y espirales. Este tipo de dibujos se puede dividir en *verticilos simples* y *verticilos dobles*, su característica principal es que en el núcleo del primero sólo se encuentran una curva mientras en el segundo se encuentran dos (Fig. 18). También pueden encontrar verticilos con tres deltas llamados también trideltos, aunque esto sucede con poca frecuencia.

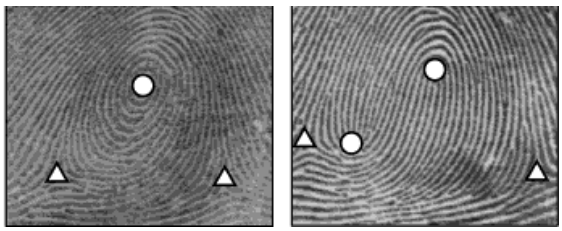


Fig. 18 Verticilo simple y Verticilo doble

De manera general los sistemas biométricos automatizados utilizan estas cinco figuras para dividir a las huellas dactilares, esto con el fin de permitir búsquedas más rápidas y eficientes dentro de las bases de datos y para tener un mayor control en el reconocimiento de dactilogramas.

Es de relevancia mencionar que según Wilson, Candela y Watson en 1994 (Tabla 1) realizaron un estudio donde encontraron las siguientes proporciones en la clasificación de las huellas dactilares (Maltoni & Co., 2009, pág. 238):

Figura dactilograma	Porcentaje de personas en esta clasificación
Arco	3.7%
Arco “casa de campaña”	2.9%
Presilla interna	33.8%
Presilla externa	31.7%
Verticilo	27.9%

Tabla 1. Resultados del estudio de Wilson, Candela y Watson

4.2.2.2 Reconocimiento del iris

El iris es un órgano interno del ojo que se encuentra por detrás de la cornea, es la zona coloreada del ojo y rodea a la pupila de color negro (Fig. 19).

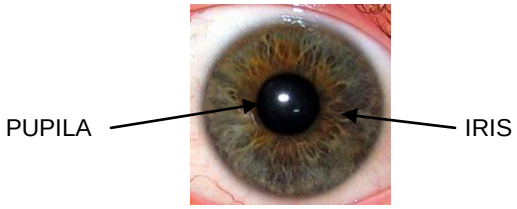


Fig. 19 Iris

En las últimas dos décadas se ha innovado en la creación de tecnología para el reconocimiento del iris ya que es un órgano que al igual que las huellas dactilares es estable y permanece sin alteraciones durante la vida de una persona.

Una de las ventajas del iris sobre otros órganos es que se puede tomar una fotografía a distancia y no hay necesidad de que la persona tenga contacto directo con algún aparato.

El procedimiento para el reconocimiento del iris es el siguiente (Fig. 20) (Florinda y Carranza, 2009, s/p):

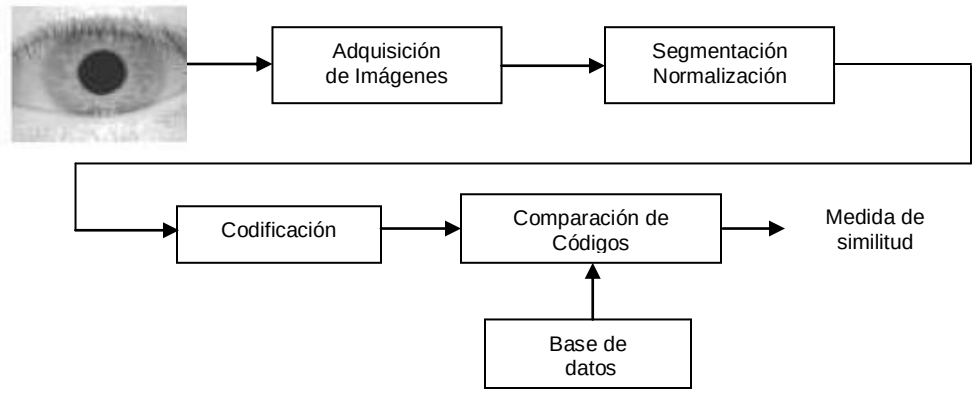


Fig. 20 Procedimiento para la captura de iris

- *Adquisición de imágenes*: El gran reto es que la imagen obtenida sea lo suficiente mente clara para obtener una imagen fiel del iris, esto se consigue con una buena iluminación y con una cámara de alta definición, aunque en estos tiempos una cámara digital común podría ser utilizada para realizar este tipo de tomas.
- *Segmentación y Normalización*: La segmentación es utilizada para reconocer la región del iris, la cual es extraída de la imagen obtenida, cuando se obtiene una imagen clara del iris se procede a la normalización, esto significa que se obtendrá una imagen estándar del iris lo cual elimina el problema que se presenta cuando la pupila crece o disminuye y así las comparaciones puedan realizarse sin mayor problema.
- *Codificación*: permite extraer los patrones, por lo general en un plano 2D se obtiene una plantilla que ayuda a ubicar cada parte del iris y generar un código único e irrepetible.
- *Reconocimiento*: Previamente se debió obtener una base de datos que contenga el código de iris de diferentes personas, después se deben realizan los pasos de obtención de imagen, segmentación, normalización y codificación. Al realizar este procedimiento y obtener el código del iris se realiza una comparación dentro de una base de datos para saber si se encuentra o no ese código, si es válido se puede permitir un acceso dependiendo del fin que tenga la autentificación.

En los últimos años el reconocimiento de iris ha mejorado considerablemente y poco a poco se está volviendo más popular entre los usuarios que requieren identificar a otras personas.

4.2.2.3 Reconocimiento facial

Es un método que ha adquirido cierta popularidad dentro del sector de la seguridad para el control de acceso y para la identificación de personas, el gran problema con el que han tenido que lidiar los diferentes algoritmos desarrollados, es que a diferencia del iris o la huella digital el rostro puede ser alterado, además se debe cuidar siempre la iluminación, el ángulo de donde se toma, las expresiones faciales, el uso de lentes y el tipo de cámara utilizada.

En este tipo de sistemas se pueden encontrar dos errores principales denominados: FAR y FRR (Jain, Flynn, Abraham, 2009, pág. 44). El primero tiene la probabilidad de que el

sistema autorice a una persona que no se encuentra en el sistema y el segundo al contrario que impida el acceso a una persona registrada en el sistema.

Existen dos métodos básicos para el reconocimiento de rostro: el primero se basa en imágenes 2D, este tipo de sistemas toman una fotografía de la cara y miden la distancia entre las diferentes características del rostro para crear una plantilla de identificación, el segundo método recientemente creado es la toma de imágenes 3D, la ventaja de este tipo de modelos comparado con los 2D es que se elimina la posibilidad de que la imagen que es tomada sea una fotografía en papel, la desventaja de este tipo de técnica es que el procesamiento para crear el modelo 3D requiere de más tiempo.

El reconocimiento de rostro se puede dividir en dos categorías (Jain, Flynn, Abraham, 2009, págs. 44-45):

- *Reconocimiento de características*: este método se basa en las relaciones geométricas como las áreas, distancias y ángulos que pueden existir entre las características particulares de la cara.
- *Reconocimiento de apariencia:* esta técnica usa las propiedades globales de la imagen del rostro para determinar los patrones de verificación, esto lo hace utilizando vectores procesados por una computadora para representar el rostro de manera eficiente. Este método es el más utilizado dentro de la industria biométrica. En seguida se mencionan los algoritmos más populares (Jain, Flynn, Abraham, 2009, págs. 45-70):
    - Análisis de los Principales Componentes (PCA por sus siglas en inglés).
    - Análisis por Discriminación Lineal (LDA por sus siglas en inglés).
    - Análisis de Componentes Independientes (ICA por sus siglas en ingles).
    - Análisis de Rasgos Locales (LFA por sus siglas en inglés).
    - Reconocimiento a partir de múltiples coincidencias (EBGM por sus siglas en inglés).
    - Redes Neuronales (NN por sus siglas en inglés).
    - Máquinas de Soporte Vectorial (SVM por sus siglas en inglés).

En conclusión las técnicas para realizar el reconocimiento de rostro han evolucionado de manera significativa, considerando todos los problemas que ha tenido que atender a la hora de crear un algoritmo, este tipo de biométricos se pueden utilizar en conjunto con

otro tipo de técnicas, como el reconocimiento de iris, para hacer más eficiente la seguridad.

4.2.2.4 Reconocimiento de olor

Las investigaciones para este tipo de biometría se siguen desarrollando y de manera comercial no se tiene ningún producto a la venta, este tipo de biométricos tratan de reconocer a un individuo por medio de los químicos que despide el cuerpo humano, el olor que caracteriza a una persona de manera individual, el problema que se presenta con este tipo de dispositivos es que el olor que se puede encontrar en un cuerpo humano puede mezclarse fácilmente con los olores del medio ambiente. Uno de los modelos para la identificación de olores es el que presenta la Universidad Lappeenranta (Fig. 21) (Korotkaya, 2003, pág. 4):

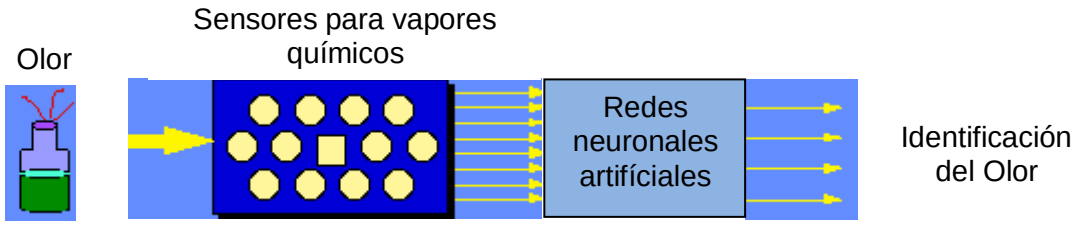


Fig. 21 Proceso para la identificación de olores

- *Sistema de sensores:* este artefacto cuenta con una serie de sensores que se dedica a identificar las diferentes sustancias químicas que puede contener el olor a reconocer, los sensores se activan al identificar los diferentes componentes que contenga la sustancia y se guardan en una base de datos para ser analizados posteriormente.
- *Sistema de reconocimiento de patrones:* al tener por separado las diferentes sustancias químicas del olor que se desea reconocer, estos se pasan a un sistema dedicado a identificar por medio de patrones el tipo de sustancia del que se está hablando, para ello se requiere de una red neuronas artificiales que permitan elegir de manera eficiente el resultado, uno de los problemas con los que se han enfrentado los investigadores es que es muy complicado crear un modelo matemático que identifique cuando se trata de una u otra sustancia.

En este tipo de biometría no se han tenido grandes logros debido a lo complicado que ha sido crear un sistema artificial que identifique grandes cantidades de olores, más aun de identificar a una persona de otra. Se han podido crear artefactos que detectan algunas sustancias, pero el problema que siempre se presenta como se describió al principio es que el medio ambiente está lleno de olores y esto impide aislar de manera correcta el olor que se está tratando de identificar.

4.2.2.5 Reconocimiento de Firma

Este tipo de técnica está basado en el reconocimiento de la firma escrita de un individuo, existen diferentes dispositivos que la pueden capturar, en general se utilizan pizarras digitales o PDAs, el tipo de características a ser tomadas en cuenta varían de un distribuidor a otro.
Los patrones a ser considerados para la identificación de una firma son los estáticos o geométricos y los dinámicos (NTS-Dynamic Signature, 2006, págs. 1-3):

- *Estáticos o geométricos:* para este caso se consideran una serie de coordenadas (X , Y , Z), las coordenadas X y Y determinan dentro de un plano cartesiano la posición de los diferentes símbolos de la firma mientras que la coordenada Z determina si se despego la pluma de la pizarra.
- *Dinámicos:* dentro de los dinámicos se encuentran características tales como la velocidad, la aceleración, el tiempo, la presión y la dirección, además algunos distribuidores también determinan el ángulo con el que se tomo la pluma, estas características son tomadas mientras el individuo realiza la firma.

El uso de la firma escrita para identificación de personas es de poca eficiencia, comparado con los demás métodos, aunque es difícil que un falsificador pueda replicar una firma si se combina el uso de las características dinámicas y estáticas, es recomendable usar este método en aplicaciones que no estén en el rango de seguridad máxima.

4.2.2.6 Reconocimiento de voz

El reconocimiento de voz a tenido avances significativos en los últimos años aunque todavía no es altamente confiable, esta metodología está basada en la captura de sonidos producidos por el tracto vocal, donde se considera el comportamiento de características físicas del individuo tales como la lengua, la laringe, la mandíbula y la resonancia que producen las fosas nasales al paso del aire, al tomar en cuenta estas particularidades se pueden extraer patrones suficientes para determinar si una persona es quien dice ser.
Existen dos modalidades para el reconocimiento de voz, las cuales son (NTS-Speaker Recognition, 2006, 1-6):

- *Dependiente del texto (modo limitado):* esta técnica trata de reconocer a un individuo por medio de la captura de una frase, la cual podría ser utilizada como contraseña, la frase puede ser por ejemplo “uno dos tres cuatro”. La frase es capturada por medio de un micrófono la cual posteriormente es trasformada de modo análoga a digital, enseguida se extraen las características de la frase para crear un modelo gráfico. La mayoría de los sistemas de verificación *dependientes del texto* utilizan el concepto de Modelos Markov Ocultos (HMMs), que son modelos que proveen de una representación estadística a los sonidos producidos por el individuo. El HMM representa las variaciones subyacentes y los cambios temporales a lo largo del tiempo en los estados del discurso utilizando las características de calidad, duración, intensidad y tono. El HMM utiliza la voz para crear un número de vectores de estado que representan las variaciones de las formas del sonido, que son características de la fisiología y el comportamiento de un individuo. Una vez que se a tomado una muestra y ha sido guardada en una base de datos un individuo puede ser identificado cuando este pronuncia la frase guardada, se consideran las mismas características de calidad, duración, volumen y tono para extraer los patrones y compararlos con el modelo de la identidad, o hipotética identidad. Si el modelo de voz identificado no es reconocido se le denomina *antivoz*. La muestra de la voz es comparada para producir un *radio de similitud*, si el individuo es legítimo la prueba será positiva de lo contrario se indicará que la voz no es reconocida.
- *Independiente del texto (modo ilimitado)*: Este modelo en lugar de tomar una frase o frases para ser identificadas, toma las características generales del espectro de

voz. Para realzar el análisis se deben considerar los niveles bajos y niveles altos del espectro. Las investigaciones actuales en el área de reconocimiento de voz *independiente del texto* están concentradas mayormente en el nivel bajo. Dentro del nivel bajo se consideran las características básicas de la señal captada tales como: amplitud, periodo de muestreo, frecuencia, etc. Mientras que las de alto nivel incluyen: funciones prosódicas como el ritmo, la velocidad, la modulación y entonación, tipos de personalidad e influencia parental, semántica, pronunciación, relaciones con el lugar de nacimiento, estatus socio-económico, etc. La fusión de características de alto nivel con la información de bajo nivel de espectro se está convirtiendo en una técnica muy popular en los laboratorios ya que ha permitido mejorar los métodos de reconocimiento de voz para *texto independiente*.

Se puede concluir que esta técnica crea muchas expectativas para el futuro, ya que si se consigue obtener un sistema estable esta metodología se podría implementar en muchas aplicaciones, el problema surge en los dispositivos que captan la voz así como el ruido del medio ambiente, además de que los algoritmos creados hasta el momento no son cien por ciento confiables.

4.2.2.7 Dinámica del teclado

La dinámica del teclado se basa en la forma en la que un individuo escribe sobre un teclado. Según la Fundación Nacional de Ciencia (NSF, por sus siglas en inglés) y el Instituto Nacional de Estándares y Tecnología (NIST por sus siglas en inglés) han encontrado patrones de tipificación para obtener características únicas que pueden ser identificadas cuando una persona utiliza un teclado.
Las principales características a ser consideradas dentro del reconocimiento de patrones son (360Biometrics, 2010, s/p):

- *Velocidad de escritura*: se considera que una persona siempre podrá escribir un número máximo de palabras en cierto tiempo, puede que en algunas ocasiones el individuo escriba con más lentitud pero nunca podrá escribir más rápido después de obtener su velocidad de escritura.
- *Velocidad de escritura de los números*: una persona tendrá un patrón en la velocidad de escritura de números.

- *Uso de las teclas*: siempre existe una determinada variación de tiempo entre la pulsación de una tecla y otra, en particular se puede considerar cuanto tiempo mantiene pulsada la tecla para cambiar de mayúsculas a minúsculas y viceversa, además del uso de la barra espaciadora.
- *Patrones de error: s*e considera que palabras ha escrito mal y en cuanto tiempo las corrigió y los errores más comunes dentro de su redacción.
- *Uso de laterales*: si la persona es diestra o zurda, dependiendo del lateral usado puede ser más rápida con la mano derecha o con la izquierda a la hora de pulsar teclas.

Este método es una alternativa muy útil dentro del mundo de la computación pues aunque un individuo haya ingresado a un sistema podemos verificar por medio del análisis del uso del teclado si se trata de una persona que en realidad tiene acceso a esa información o si se trata de un intruso.
Esta técnica se empezó a estudiar hace 30 años por Gaines y se ha ido popularizando poco a poco dentro del área de los biométricos ya que puede ser una pieza indispensable a la hora de reconocer intrusos en un sistema de información (Monrose, 1999, pág. 5).

4.2.2.8 Geometría de la mano

La mano está compuesta de muchos huesos, músculos y ligamentos diferentes que permiten una gran cantidad de movimientos y destreza. Existen tres tipos principales de huesos en la mano, incluyendo los siguientes:

- *Falanges*: los 14 huesos que se encuentran en los dedos de cada mano. Cada dedo tiene tres falanges (distal, media y proximal), el pulgar tiene sólo dos.
- *Huesos metacarpianos*: los cinco huesos que componen la parte media de la mano.
- *Huesos carpianos*: los ocho huesos que forman la muñeca. Los huesos carpianos están conectados a dos huesos del brazo: el cúbito y el radio.

Dentro de la biometría se encuentra el estudio de la geometría de la mano la cual básicamente se dedica a extraer la silueta de la mano que se genera de la toma de

imágenes de la piel que cubre la parte inferior de los falanges y los huesos metacarpianos, la extracción de los patrones se realiza mediante un escáner, como el que se muestra en las (Figs. 22 y 23), el dispositivo cuenta con cinco clavijas que ayudan a ubicar los dedos de la palma de la mano con la finalidad de tomar una imagen la cual será procesada para la extracción de las características como (Biometría Informática, 2009, s/p):

- Anchura de cada uno de los dedos excepto el pulgar.
- Alturas del dedo medio, del dedo meñique y de la palma de la mano.
- Ángulos entre la línea de unión de los puntos inter-dedo y la horizontal.
- Desviaciones de las falanges con respecto a la línea ideal que deberían trazar.

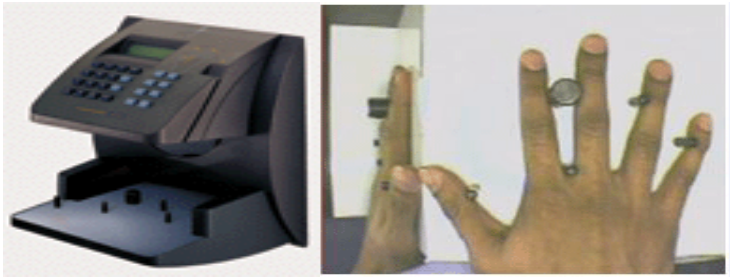


Fig. 22 Biométrico para la palma de la mano

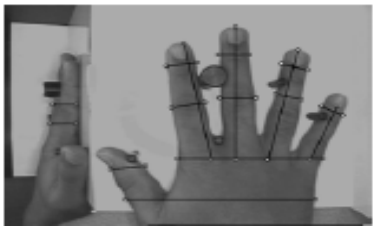


Fig. 23 Características de la geometría de la mano

Para obtener una descripción completa de la mano por lo general se deben realizar tres o cuatro tomas según el estándar ANSI INCITS 396-2005 Hand Geometry Interchange Format (Biometrics, 2006, pág. 3).

Este tipo de biometría no ha tenido mucho éxito dentro del mercado debido a los costos que puede generar la compra de los escáneres así como los altos precios que presenta el software.

4.2.3 Biometría en la Ingeniería en Procesamiento Digital

La mayoría de los métodos biométricos requieren de un escáner para obtener los patrones de la parte del cuerpo que nos sirve como *código de identificación* para saber si una persona es quien dice ser (iris, rostro, huella digital, geometría de la mano, firma, dinámica del teclado, etc.), en el caso particular de la huella dactilar se requiere de un escáner especial para extraer las características del dactilograma.
Básicamente existen dos técnicas utilizadas para la obtención de la imagen digital de una huella dactilar, las cuales se describen enseguida (Maltoni D., Maio D., Prabhakar, 2009, págs. 53-54):

- *Off-line:* esta técnica requiere que el individuo entinte de color negro sus dedos para después imprimir su huella sobre una cartulina blanca, posteriormente la imagen impresa en papel es digitalizada con un escáner común y corriente. Este método se aplica comúnmente cuando a ocurrido un crimen, se obtienen las impresiones dejadas por el criminal en el lugar de los hechos y posteriormente son digitalizadas para extraer los patrones de la huella.
- *Live-scan:* como su nombre lo indica, esta técnica realiza la obtención de los dactilogramas en tiempo real por medio de un escáner de huellas dactilares, este tipo de escáneres son aceptados hoy en día por los AFIS ya que son mucho más eficientes, rápidos y baratos, además el ahorro de tiempo es mayor que con el método *off-line.* La técnica *live-scan* es utilizada de manera comercial y es la que se utilizará en el presente trabajo.

En la (Fig. 24) se puede observar la estructura general de los escáneres de huellas dactilares *live-scan* donde: el sensor lee la superficie del dactilograma y transforma la imagen captada de analógica a digital (D/A), posteriormente el módulo de interfaz del escáner se encarga de realizar la comunicación entre la computadora y el dispositivo para que la PC procese los datos y obtenga los patrones de la huella o guarde una imagen de la misma (Maltoni D., Maio D., Prabhakar, 2009, pág. 54).
Este tipo de dispositivos por lo general requieren de una computadora para procesar y almacenar la información, pero afortunadamente algunos distribuidores han creado dispositivos que incluyen *todo* en el mismo dispositivo, el sensor capta la señal, la

transforma, la procesa y la guarda para ser consultada posteriormente, esto facilita a un más el uso de dispositivos con el uso del método *live-scan*.

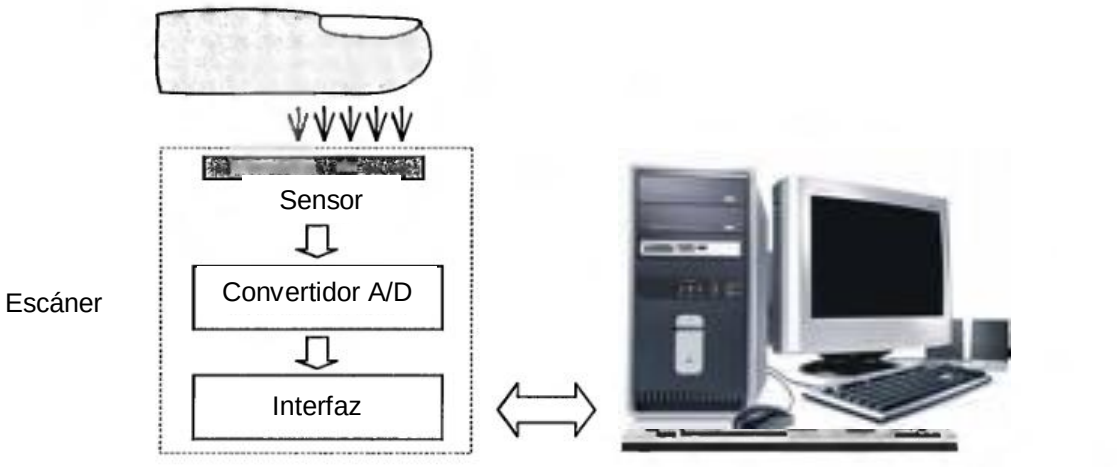


Fig. 24 Funcionamiento general de un escáner de huellas dactilares

En seguida se mostrarán las características principales que debe tener una imagen digital de una huella dactilar (Maltoni D., Maio D., Prabhakar, 2009, págs. 55-57):

- *Resolución:* indica el número de puntos o píxeles por pulgada de una imagen (dpi, por sus siglas en inglés). La resolución mínima que exige el FBI en sus estándares es de 500 dpi, esta resolución es cumplida por la mayoría de los distribuidores; sin embargo la resolución mínima para obtener las minucias de un dactilograma está entre 250 dpi a 300 dpi. En la (Fig. 25) se observa la imagen de la misma huella digital en diferentes resoluciones, lo que permite concluir que a mayor resolución mejor apreciación de los detalles, lo cual facilita la obtención de patrones.

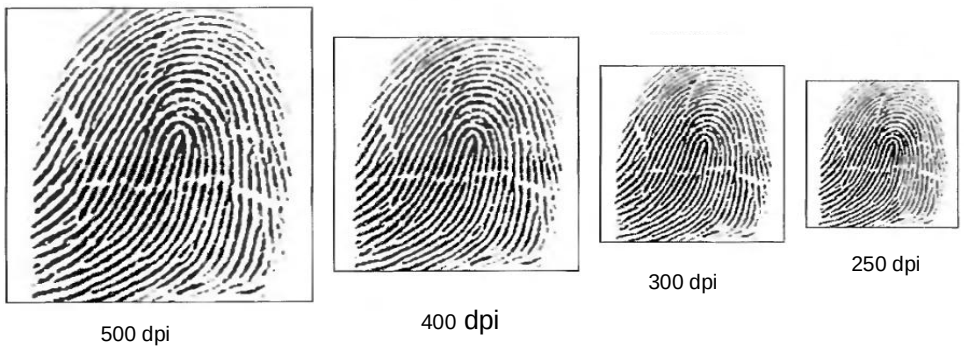


Fig. 25 Imagen de huella dactilar en diferentes resoluciones

- *Área:* el tamaño del área del escáner es un parámetro fundamental, se recomienda que tenga una medida mínima de una por unas pulgadas (parámetro obtenido de los estándares del FBI), esto permite que el dedo pueda colocarse de manera correcta sobre el sensor y este pueda captar una buena imagen. Algunos distribuidores sacrifican el tamaño del dispositivo por el costo, el problema reside en el tipo de personas que utilizaran el sensor, ya que si estas tienen dedos muy grandes y el sensor es demasiado pequeño a la hora de tomar las muestras de los dactilogramas al ser estos mayores al biométrico no se podrá delimitar su contorno, lo que acusará que los datos obtenidos no sean eficientes pues no se tendrá un punto de referencia apropiado para determinar las coordenadas de las minucias.
- *Número de píxeles:* es necesario saber que nitidez tendrán las imágenes por tanto se puede determinar el número de píxeles en una imagen con una operación muy sencilla: se toma en cuenta el tamaño del área del sensor y los píxeles por pulgada que el dispositivo puede obtener en cada imagen, la ecuación es la siguiente:

  r= píxeles por pulgada
  h = altura en pulgadas del sensor
  a= ancho en pulgadas del sensor

  (rh x ra) píxeles

  Por ejemplo tenemos un sensor de 500 dpi con un área de 1.5 pulgadas de ancho por 2 pulgadas de altura por tanto aplicando la ecuación tenemos que:

  r= 500 dpi
  h= 2 in (500) (2) x (500) (1.5) pixels = 1000 x 750 pixels
  a= 1.5 in

  Por tanto las imágenes obtenidas serán de 1000 x 750 píxeles.

- *Rango dinámico (o profundidad):* esta característica determina la intensidad de los píxeles, indica que tan claros o obscuros son, dentro del procesamiento de imágenes de huellas dactilares no es conveniente el uso de colores, por tanto la

profundidad se refiere al número de bits que contiene cada pixel para determinar un tono en gris, por ejemplo el estándar usado por el FBI determina que la profundidad debe de ser de 8 bits lo que da un conjunto de 256 tonos de grises. Algunos dispositivos obtienen durante la toma de la imagen una profundidad de dos o tres bits para ser procesada y transformada posteriormente a una profundidad de 8 bits mediante el software. No se han realizado estudios sobre que tanto puede afectar la profundidad en las imágenes para extraer los patrones de los dactilogramas, sólo se ha determinado que para obtener una buena imagen los dispositivos deben usar más de un bit de profundidad.

- *Exactitud geométrica:* esta se refiere a la distorsión máxima que puede ser generada durante la adquisición de datos en cuanto es tomada la imagen, la mayoría de los dispositivos pueden corregir este error, pues si no es compensada esta distorsión puede generar errores a la hora de extraer los patrones de reconocimiento.
- *Calidad de la imagen*: la calidad es muy importante para la obtención de imágenes pero para un escáner de huellas dactilares es difícil determinar si una foto es adecuada o no, por tanto se debe considerar aspectos como la humedad excesiva o la falta de humectación del dedo, además se debe tomar en cuenta el tipo de personas que utilizarán el sensor ya que si se tiene un escáner que obtiene imágenes de baja calidad será poco eficiente si se trabaja con individuos que realizan trabajos manuales, ya que las crestas de sus dedos por lo general están desgastadas lo que puede generar que la imagen se vea como una simple mancha (Fig. 26).

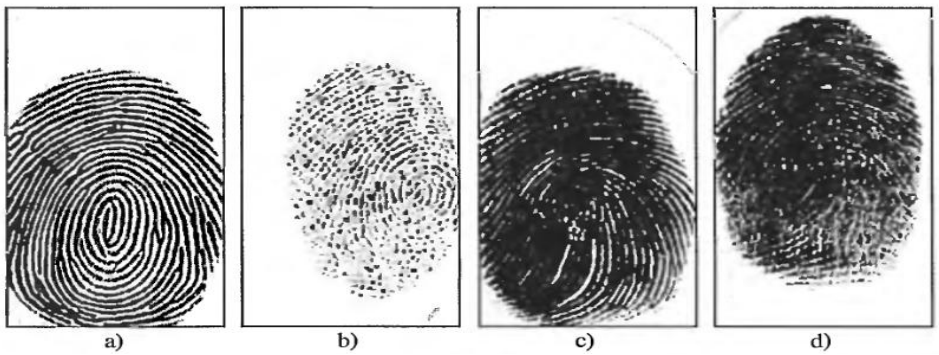


Fig. 26 Muestras de huellas digitales obtenidas con un escáner óptico. a) huella correctamente tomada, b) huella de un dedo muy seco, c) huella de un dedo muy húmedo y d) huella de un dedo sucio.

El procesamiento de imágenes dentro de la biometría es muy importante ya que de esto dependerá la eficiencia de las muestras que se obtengan.

Se debe ser muy cuidadoso a la hora de considerar el tipo de escáner que se adquiere dependiendo de las necesidades que se deseen cubrir, por tanto conviene hacer un balance sobre el presupuesto que se tiene y el tipo de seguridad que se desea brindar dentro de la empresa o institución. En el capítulo de implementación se describirá de manera sucinta la descripción de dos escáneres.

4.3 Bases de datos

Las bases de datos siempre han sido de gran utilidad para la humanidad. Su función principal es la de recopilar, estructurar y almacenar datos con la finalidad de consultarlos en cualquier momento, sin la necesidad de volverlos a recopilar u ordenar nuevamente.
Durante el paso del tiempo han existido diferentes métodos para organizar los datos, un dato curioso es que las técnicas que han surgido no siempre han sido aceptadas de inmediato pues el dominio de las empresas en el mercado lo ha impedido, por ejemplo, en el caso de las bases de datos jerárquicas y de red las cuales fueron implementadas por IBM, la metodología no fue cambiada a pesar de que existía un método mucho mejor para la organización de datos denominado *bases de datos relacionales,* esta técnica fue aceptada y difundida hasta que IBM se dio cuenta de que perdería a sus clientes si no comenzaba a trabajar con esta nueva tecnología.
En esta sección se podrá consultar la definición de *base de datos*, una descripción del método más utilizado en nuestros días para estructurar datos (bases de datos relacionales y diagrama entidad relación) y un breve recorrido sobre los diferentes manejadores de datos existentes en el mercado. Se encuentra en el (Apéndice III) una breve historia de las bases de datos.

4.3.1 Definición de bases de datos

Antes de la creación de las bases de datos tal como se conocen hoy en día, los datos se almacenaban en un sistema de archivos, los cuales contenían la información que se requería guardar, este método fue muy engorroso y desastroso, pues cuando se tenía gran cantidad de información era difícil encontrarla y modificarla, por tanto, para comprender que es una base de datos se describirá primero el concepto de archivo para dar el salto a la definición de bases de datos y entender mejor su significado e importancia, (Teorey, Ligtstone y Nadeau, 2006, pág. 2).
El componente básico de un *archivo* en un sistema de archivos es un *dato,* el cual se define con el nombre con el que se identifica algún objeto o cosa en el mundo real, (por ejemplo: el apellido paterno, la dirección, número de identificación, etc.). Un grupo de *datos* que hablan sobre un mismo tema son denominados *registros*, por ejemplo, información sobre un cliente, el salario de cada empleado, etc. De estos dos conceptos se

puede deducir entonces que un *archivo* es una colección de *registros* que hablan sobre un mismo tema.

Dentro de los sistemas de bases de datos podemos relacionar estos conceptos de la siguiente manera: en las bases de datos relacionales; un dato es denominado *columna o atributo;* un registro es llamado *fila o tupla;* y un archivo es conocido como *tabla.*

Por tanto se puede definir que una *base de datos*, según (Date, 2001, págs. 2-3), es una colección de tablas interrelacionadas, la cual sirve para almacenar la información de múltiples usuarios de una organización. El motivo del uso de las bases de datos en lugar de archivos se centra en la habilidad que se tiene con estas para organizar datos, la facilidad para acceder a la información, y la rapidez para actualizar los datos de un registro.

Para administrar una base de datos se requiere un *Sistema Manejador de Bases de Datos (DBMS por sus siglas en inglés)*. Del cual se abundará más adelante.

4.3.1.1 Características de una base de datos

Cualquier base de datos que se desarrolle debe contener las siguientes características:

- *Seguridad:* el administrador debe considerar la metodología que más le convenga para resguardar la información contenida en la base de datos. Se recomienda el uso de software especializado contra intrusos (por ejemplo la implementación: Firewalls, antivirus, etc.), también se deben imponer restricciones para el acceso a la base de datos, es conveniente utilizar reglas para que cada usuario sólo pueda ingresar a la información que le pertenece y para que los intrusos no puedan robar información.
- *Integridad:* esta característica permite establecer reglas con restricciones para que la información sólo sea modificada y observada por el usuario a quien le pertenece.
- *Concurrencia:* toda base de datos debe permitir el uso de múltiples procesos, esto quiere decir que varios usuarios puedan modificar o consultar un mismo registro al mismo tiempo sin perder la integridad de los datos.
- *Mínima redundancia*: dentro de las bases de datos llegan a existir problemas cuando un dato se repite muchas veces sin ser necesario (*redundancia de datos*), esto afecta el rendimiento del sistema, pues este se vuelve más lento y se

desperdicia espacio, por eso es conveniente realizar un diseño donde la redundancia sea mínima.

Los puntos que se acaban de describir son esenciales y deben ser considerados cuando se realiza el diseño de una base de datos.

4.3.2 Bases de datos relacionales

Como se menciona en la reseña histórica (ver Apéndice III) E. F. Codd de IBM en 1970, propuso el *modelo relacional* en su afán por mejorar los diseños de las bases de datos, su modelo cambio radicalmente la forma de estructurarlas, hizo que se pasara de bases de datos de *transmisión estándar* a bases de datos de *transmisión automática* (Rob y Coronel, 2004, pág. 32).
En sus inicios el modelo tuvo poco éxito ya que las computadoras no tenían la capacidad para trabajar con modelos relacionales debido a la cantidad de recursos que requería. Pero gracias al acelerado crecimiento de la informática este modelo se fue popularizando hasta llegar a ser el más utilizado a nivel mundial.

4.3.2.1 Estructura del modelo relacional

El *modelo relacional* es administrado y ejecutado mediante un *Sistema de Administración de Base de Datos Relacional* (*RDBMS por sus siglas en inglés*), este sistema es el encargado de mantener la comunicación entre el usuario y la base de datos de una forma lógica y clara para los seres humanos. El RDBMS realiza todos los procesos de bajo nivel para realizar las tareas que se le soliciten a la base de datos. Esto permite que las personas vean la base de datos como un conjunto de tablas en las que se pueden guardar datos.
Una *tabla* está formada por un conjunto de *filas* y *columnas* (Fig. 27).

ID_Persona	Nombre	Apellido_Paterno	Ocupación
01	Alexei	Carrizales	Ingeniero
02	Candy	Navarrete	Psicóloga
03	Rocío	Mayorga	Maestra

Fig. 27 Tabla de una base de datos

Las *tablas* también son llamadas *relaciones*, estas *relaciones* mantienen una *entidad* en común, por ejemplo (Fig. 28), se puede observar que en la primer tabla se muestran los nombres con sus apellidos de varios usuarios así como un campo llamado ID_Dirección, este campo es denominado *entidad en común* pues se encuentra en las dos tablas y gracias a esta se puede conocer el domicilio de cada individuo sin la necesidad de que todos los datos se encuentren en la misma tabla, esto proporciona un nivel de mínima redundancia en las base de datos y un mayor entendimiento en la estructura de la información.

ID_Usuario	Nombre	Apellido_Paterno	Apellido_Materno	ID_Dirección
01	Alexei	Carrizales	Mayorga	011
02	Candy	Navarrete	Barrios	011
|03	Eduardo	Pedroza	García	033

Relación entre tablas

ID_Dirección	Calle	Número	Colonia	CP
011	Costeñas	118	Lomas del sol	14000
022	Av. Universidad	2001	Coyoacán	15350
033	Lomas del Chairel	203	Hidalgo	19360

Fig. 28 Esquema básico del modelo relacional

Siguiendo con el ejemplo de la (Fig. 28) se puede observar que varias personas pueden tener una misma dirección pero que una persona solo puede tener un domicilio, este tipo de relaciones se mencionarán a detalle más adelante, lo que se desea mostrar con este ejemplo es la *estructura básica* con la que trabaja un modelo entidad relación.

4.3.2.2 Componentes del modelo relacional

Según (Rob y Coronel, 2004, págs. 58-59) dentro del modelo relacional el componente que nos dará los datos que deseamos obtener sobre una persona, objeto o cosa es denominada *entidad*. Un *conjunto de entidades* es un grupo de características que hablan sobre un mismo tema. Por ejemplo si tenemos un grupo de personas en una empresa podemos denominar a estas como un conjunto de entidades llamadas EMPLEADOS.

Las características de cada *entidad* son denominadas *atributos*, por ejemplo la entidad EMPLEADOS en una empresa puede tener los siguientes atributos: nombre, cargo, salario, edad, dirección, etc.

Todos los atributos deben ser definidos con claridad para que al realizar consultas a la base de datos el usuario identifique inmediatamente de que se está hablando.

Las entidades con sus atributos son depositados en *tablas* las cuales como ya se mencionó son estructuras bidimensionales compuestas por *filas* y *columnas.*

Las tablas deben cumplir con las siguientes características:

1. Una tabla es una estructura compuesta por filas y columnas (Fig. 27).
2. Cada fila representa una sola entidad de un grupo de entidades (Fig. 29).
3. Cada columna representa un atributo, por lo cual cada columna debe tener un nombre distinto (Fig. 29).
4. Cada intersección de entre una fila y una columna representa un dato único (Fig. 29).
5. Una tabla debe tener un atributo o combinación de estos para que identifique de manera única a cada fila se le denomina *clave primaria* (Fig. 29).
6. Todos los valores de una columna deben ajustarse al mismo tipo de dato. Por ejemplo si el atributo contiene caracteres toda la columna debe contener caracteres (Fig. 29).
7. El orden de las filas y las columnas no es importante en este modelo.
8. Cada columna tiene un intervalo de valores específico conocido como domino de atributo.

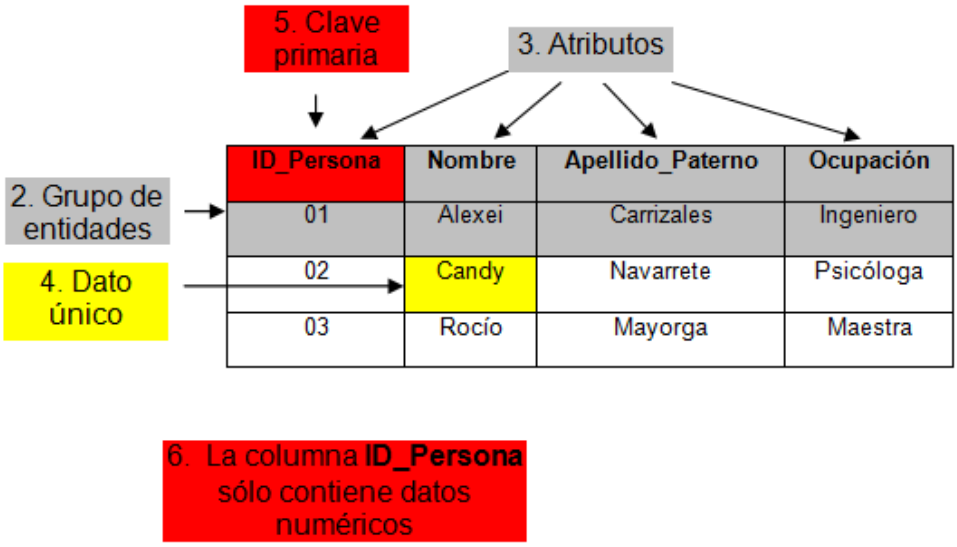


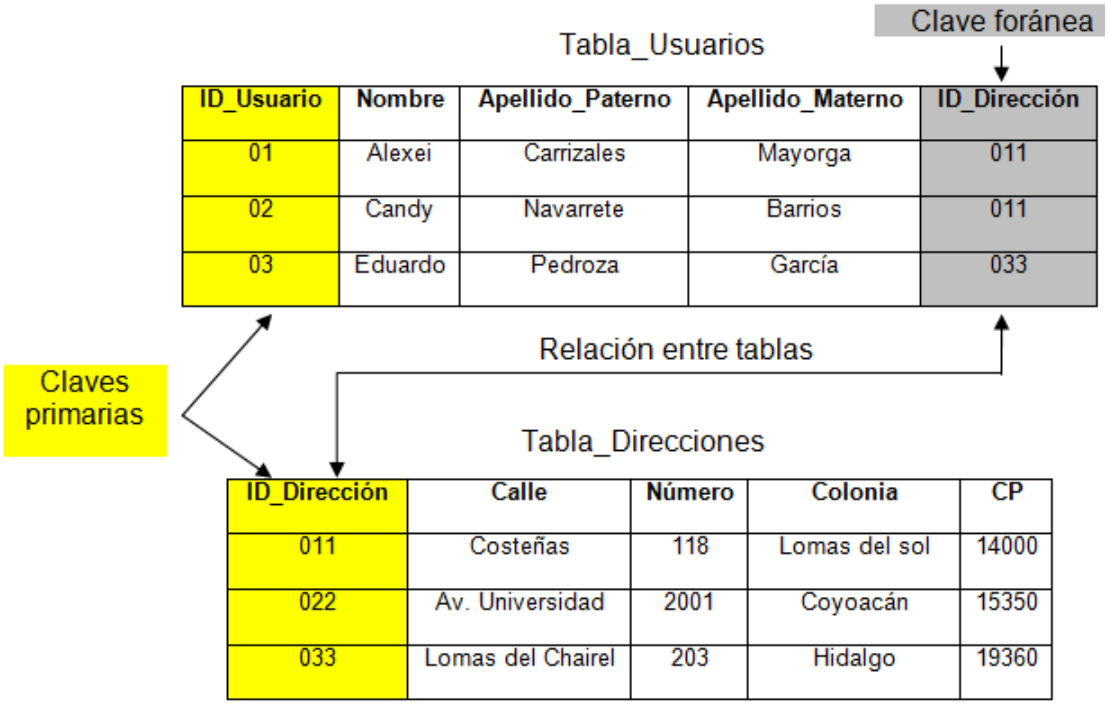
Fig. 29 Características de una Tabla

Es importante que las tablas cumplan con estas características para que el modelo funcione de manera adecuada. Además, es de fundamental importancia resaltar el punto número cinco pues las *claves* dentro de las tablas son determinantes ya que ayudan a identificar a cada fila, esto es esencial para crear la relación entre tablas, por ejemplo en la (Fig. 30) al tener el atributo *ID_Direccion* de un usuario se puede obtener tanto el nombre como la dirección de la persona.

Existen diferentes tipos de claves, las cuales se describirán enseguida (Rob y Coronel, 2004, pág. 64):

- *Clave compuesta*: clave compuesta por más de un atributo.
- *Clave primaria*: es una clave que identifica de manera única a cada fila, en este caso el atributo que se designa como clave primaria jamás debe ser nulo, esto permite que la base de datos se mantenga integra, en el ejemplo de la (Fig. 30) los atributos que contienen un dato único e irrepetible son: para la *Tabla_Usuarios* el *ID_Usuario* y para la *Tabla_Direcciones* el *ID_Dirección*.
- *Clave foránea*: este tipo de claves se utilizan cuando una clave primaria se encuentra en otra tabla, esta clave se puede repetir las veces que se requiera en otras tablas ya que sirve para hacer referencia a los datos de la fila de la tabla en la que se encuentra como clave primaria, por ejemplo en la (Fig. 30) en la *Tabla_Usuarios* donde se encuentran los nombres de las personas se encuentra el atributo *ID_Direccion* el cual en este caso es una *clave foránea* ya que hace

referencia a la tabla direcciones donde es una clave primaria y en donde se encuentra referenciada al domicilio que tiene cada usuario.



ID_Usuario	Nombre	Apellido_Paterno	Apellido_Materno	ID_Dirección
01	Alexei	Carrizales	Mayorga	011
02	Candy	Navarrete	Barrios	011
03	Eduardo	Pedroza	García	033

ID_Dirección	Calle	Número	Colonia	CP
011	Costeñas	118	Lomas del sol	14000
022	Av. Universidad	2001	Coyoacán	15350
033	Lomas del Chairel	203	Hidalgo	19360

Fig. 30 Ejemplo clave primaria, clave foránea

Por tanto, las claves son fundamentales para identificar y hacer referencia a las entidades de cada tabla, gracias a ellas se puede determinar la relación que existe entre estas.

Otro aspecto importante dentro del modelo relacional es el uso del *álgebra relacional*, la cual define la manera teórica de manipular los contenidos de las tablas mediante siete relaciones (Kroenke, 2003, págs. 221-227):

- *Unión*: combina las filas de dos tablas. Las tablas que se desean unir deben tener los mismos atributos, por tanto los nombres de las columnas deben ser iguales para realizar esta operación, si esta condición se presenta en las tablas se les denomina *tablas compatibles*.
- *Intersección*: da como resultado las filas que aparecen en ambas tablas, al igual que la unión las tablas deben ser compatibles.
- *Diferencia*: se realiza una operación de resta a la primer tabla se le quitan todas las filas que tiene la segunda.

- *Producto*: realiza el producto cartesiano de dos tablas, por ejemplo si se tienen 6 filas en una y 3 en otra, realiza todas las posibles combinaciones entre ambas lo que da un resultado de 18 filas en la nueva tabla.
- *Selección*: extrae todos los valores encontrados en una tabla, también se puede utilizar para enlistar todos los valores de los atributos que cumplan con cierto criterio.
- *Proyección*: indica todos los valores de los atributos seleccionados.
- *Join*: permite combinar información de dos o más tablas.

Como se mencionó estas operaciones son básicas para obtener información dentro de una base de datos relacional y son de gran ayuda para tener un control y dominio claro de los contenidos de las tablas.

4.3.2.3 Las 12 reglas de Cood para los RDBMS

A continuación se mencionarán las 12 reglas que Cood presentó para determinar que un RDBMS (*Sistema de Administración de Base de Datos Relacional*) pueda ser considerado relacional (Pertersen, 2002, págs. 13-16):

1. *Información*: Toda la información de la base de datos debe estar representada explícitamente en un esquema lógico. Es decir, todos los datos deben estar en tablas.
2. *Acceso garantizado:* Todo dato es accesible sabiendo el valor de su clave y el nombre de la columna o atributo que contiene el dato.
3. *Tratamiento sistemático de los valores nulos*: El DBMS debe permitir el tratamiento adecuado de estos valores.
4. *Catálogo en línea basado en el modelo relacional:* Los metadatos (datos que describen otros datos) deben de ser accesibles usando un esquema relacional.
5. *Sublenguaje de datos completo*: Al menos debe de existir un lenguaje que permita el manejo completo de la base de datos. Este lenguaje, por lo tanto, debe permitir realizar cualquier operación.
6. *Actualización de vistas*: El DBMS debe encargarse de que las vistas muestren la última información.

7. *Inserciones, modificaciones y eliminaciones de dato nivel*: Cualquier operación de modificación debe actuar sobre conjuntos de filas, nunca deben actuar registro a registro.
8. *Independencia física*: Los datos deben de ser accesibles desde la lógica de la base de datos aún cuando se modifique el almacenamiento.
9. *Independencia lógica*: Los programas no deben verse afectados por cambios en las tablas.
10. *Independencia de integridad*: Las reglas de integridad deben almacenarse en la base de datos (en el diccionario de datos), no en los programas de aplicación.
11. *Independencia de la distribución*: El sublenguaje de datos debe permitir que sus instrucciones funcionen igualmente en una base de datos distribuida que en una que no lo es.
12. *No subversión*: Si el DBMS posee un lenguaje que permite el recorrido registro a registro, éste no puede utilizarse para incumplir las reglas relacionales.

Estas reglas son esenciales para que una base de datos relacional sea identificada como tal y se pueda trabajar de manera adecuada con ella.

4.3.2.4 Diccionario de datos

Dentro de las bases de datos es necesario el uso de *metadatos* los cuales se definen como *la información que describe a los datos guardados en la base de datos*, los metadatos se utilizan para realizar un registro denominado *diccionario de datos* el cual contiene los detalles del contenido de una base de datos (Kroenke, 2003, pág. 352), un ejemplo sobre un diccionario de datos puede ser el siguiente (Tabla 2):

Nombre de la tabla	**Nombre del atributo**	**Contenido**	**Tipo**	**Requerido**	**Clave primaria (PK) o Calve foránea (FK)**	**FK Referida a tabla**
	ID_persona	Identificador único de la persona	INT(8)	Si	PK	
	Nombre	Nombre o nombres de la persona	CHAR(20)	Si		

Personal	Apellido_paterno	Primer apellido de la persona	CHAR(20)	Si		
	Apellido_materno	Segundo apellido de la persona	CHAR(20)	Si		
	Edad	Edad en años cumplidos de la persona	INT(2)	Si		
Dirección	ID_Direccion	Identificador único de la dirección	INT(8)	Si	PK	
	Calle	Calle donde vive la persona	CHAR(20)	Si		
	Número	Número externo del domicilio	CHAR(10)	No		
	Colonia	Colonia donde se encuentra el domicilio	CHAR(30)	Si		
	CP	Código Postal				
	ID_persona	Identificador único de la persona	INT(8)	Si		
			INT(8)	Si	FK	Personal

Tabla 2. Diccionario de Datos

En la (Tabla 2) se encuentra información sobre los datos guardaros en una base de datos, el *diccionario de datos* puede contener tantos detalles como se desee. La descripción y las referencias del contenido del *diccionario de datos* dependen del desarrollador de la base de datos.

Un *diccionario de datos* es desarrollado con la finalidad de tener toda la información necesaria para entender el contenido que tendrá o tiene una base de datos, además es de gran utilidad para encontrar errores ya que todos los RDBMS obtienen el *diccionario de datos* de manera automática, esto le permite al sistema identificar si existen atributos repetidos en la base de datos o si encuentran incongruencias entre las tablas a la hora de localizar llaves primarias y foráneas, más adelante se abundará sobre la relación entre tablas en una base de datos.

4.3.3 Manejadores de bases de datos

El *Sistema Manejador de Bases de Datos* (DBMS por sus siglas en inglés) es el software encargado de administrar una bases de datos. Como ya se mencionó un DBMS es una colección de numerosas rutinas de software interrelacionadas, cada una de las cuales es responsable de alguna tarea específica. Un DBMS funciona como enlace entre el usuario y la base de datos, básicamente lo que hace este sistema es obtener una petición, interpretarla, ejecutarla y obtener un resultado para mostrarlo al usuario.
Las funciones principales de un DBMS son (Date, 2001, pág. 44):

- *Definición de datos*: cualquier DBMS debe contar con un *Lengua de Definición de datos* (DDL por sus siglas en inglés) el cual se encarga de identificar los tipos de datos que se están almacenando dentro de la base de datos, por ejemplo se puede tener una entidad llamada ALUMNO de la cual se quieren extraer todos aquellos estudiantes que hayan ingresado hace tres años a una universidad, con el DDL podemos identificar el tipo de dato que estamos buscando, en este caso un número entero, el cual será comparado para determinar quienes tienen una estancia de tres años o más en una universidad.
- *Manipulación de datos:* para el manejo de datos se requiere de un *Lenguaje de Manipulación de Datos* (DML) el cual se encarga de recibir y ejecutar las peticiones de actualización, recuperación, eliminación o creación de nuevos registros dentro de la base de datos.
- *Optimización y ejecución:* las peticiones realizadas por el DML deben ser procesadas por un optimizador para poder ser ejecutadas, esto garantiza que las peticiones se realicen de manera correcta, lo cual nos permite una manipulación eficiente de los datos dentro del sistema.

- *Integridad y seguridad:* el DBMS es el encargado de verificar que no se violen la integridad de a base de datos ni las restricciones de seguridad impuestas por el administrador del sistema.
- *Recuperación de datos y concurrencia:* el DBMS en conjunto con otro software se encargan de imponer un control de recuperación de datos además de verificar que la concurrencia de los datos sea eficiente, para no duplicar y sobrescribir información.
- *Rendimiento:* el rendimiento es una parte esencial dentro de las bases de datos y el DBMS se encarga de esto.
- *Lenguaje de consulta*: el DBMS debe interpretar un lenguaje de consulta. El más utilizado dentro de las bases de datos relacionales es el lenguaje *SQL (*Structured Query Language) el cual está basado en álgebra relacional y el cual sirve para realizar todo tipo de acciones sobre la base de datos.

Por lo anterior, el DBMS es parte fundamental para la administración de una base de datos, pues sirve como enlace entre el usuario y la base de datos para realizar la manipulación de la información, además de encargarse de la seguridad y la integridad del sistema.

A continuación, se mencionan los diferentes DBM´s que existen en el mercado.

4.3.3.1 DBMS´s en el mercado

Hoy en día existen DBMS comerciales y no comerciales en el mercado, en seguida se enlistarán los más importantes hasta el 2010 (Wiki_en, s/p, 2010) (Tabla 3):

DBMS	Compañía	Lanzamiento al mercado	Última versión estable	Tipo de licencia
Ingres	Ingres Corp.	1974	Ingres Database 9.2	GPL y Propietaria
Oracle	Oracle Corporation	1979	11g	Propietaria
Informix Dynamic Server	IBM	1980	11.50.xC6	Propietaria
SQLBase	Unify Corp.	1982	11.5	Propietaria
DB2	IBM	1983	9.7	Propietaria
FileMaker	FileMaker	1984	11	Propietaria
Oracle RDB	Oracle Corporation	1984	7.2	Propietaria
OpenEdge	Progress Software Corporation	1984	10.1C	Propietaria
Paradox	Corel Corporation	1985	11	Propietaria
Adaptive Server Enterprise	Sybase	1987	15	Propietaria
HP NonStop SQL	Hewlett-Packard	1987	SQL/MX 2.3	Propietaria
Microsoft SQL Server	Microsoft	1989	2008 (v10.0)	Propietaria
PostgreSQL	PostgreSQL Global Development Group	1989	8.4.2	Licencia PostgreSQL
OpenBase SQL	OpenBase International	1991	11.0.0	Propietaria

Continua →

DBMS	Compañía	Lanzamiento al mercado	Última versión estable	Tipo de licencia
Advantage Database Server (ADS)	Sybase	1992	9.1	Propietaria
Microsoft Access	Microsoft	1992	12 (2007)	Propietaria
SQL Anywhere	Sybase	1992	11	Propietaria
mSQL	Hughes Technologies	1994	3.8 [1]	Propietaria
MySQL	Sun Microsystems (Ahora de Oracle)	1995	5.1.43	GPL o Propietaria
Valentina	Paradigma Software	1998	3.0.1	Propietaria
Firebird	Firebird project	2000	2.1.3	IPL y IDPL
Microsoft SQL Server Compact (Embedded Database)	Microsoft	2000	2008 (v3.5 SP1)	Propietaria
SQLite	D. Richard Hipp	2000	3.6.22	Dominio publico
Apache Derby	Apache	2004	10.5.3.0	Licencia Apache
Postgres Plus Standard Server	EnterpriseDB	2004	8.3	BSD
H2	H2 Software	2005	1.2.128	EPL y MPL
SmallSQL	SmallSQL	2005	0.2	LGPL
CUBRID	CUBRID Co., Ltd., NHN Corp.	2008	CUBRID 2008 R2.1	BSD, GPL v2

Tabla 3. Manejadores de bases de datos

Pareciera que existe gran diversidad de manejadores de bases de datos, pero lo cierto es que el mercado está dominado por Oracle, DB2, POSTGRESQL y MYSQL.

4.3.4 Diagrama entidad relación

En 1976 Peter Chen propone el modelo Entidad-Relación, creado para modelar de manera gráfica una base de datos. Desde que Chen publicó su artículo han existido muchas modificaciones al modelo y hasta el momento no existe un estándar para esta técnica, en esta sección se explicarán los conceptos básicos del modelo.
Como se mencionó existen muchos estilos para la representación gráfica del modelo entidad-relación. En el presente trabajo se presentarán los dos modelos más utilizados, el primero es el propuesto por el creador de la técnica, denominado modelo de *Chen* y el segundo es utilizado generalmente por los vendedores de software por consumir menos espacio a la hora de presentar el diagrama, este estilo es denominado *Pata de Gallo.*

4.3.4.1 Componentes de un diagrama Entidad-Relación (E-R)

Según (Barker, 1994, págs. 21-29) y (Rob y Coronel, 2004, págs. 129-149) los elementos básicos en un diagrama E-R son *las entidades, los atributos, las relaciones y la cardinalidad*. A continuación se explicarán las características de cada uno de estos:

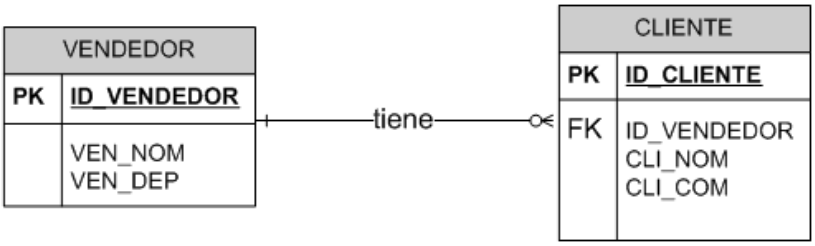
- *Entidad:* en un diagrama E-R identifica a una tabla en una base de datos (no confundir con las entidades en el modelo relacional que se refieren a una fila en una tabla), las filas en una tabla son denominadas *instancias de entidad* u *ocurrencia de entidad* dentro del diagrama E-R. Su representación en el modelo Chen y en el Pata de Gallo es un rectángulo y el nombre es un sustantivo que por lo general va escrito y en mayúsculas (Fig. 31).

Fig. 31 Entidad

- *Entidad Débil:* es aquella que es dependiente de la existencia, esto quiere decir que existe solo si la entidad con la que se relaciona existe, pero además la entidad débil contiene una clave primaria parcial o su clave primaria es la clave primaria de la entidad con la que se relaciona, por ejemplo, un vendedor tiene varios clientes, para que un cliente exista debe estar asignado a un vendedor por tanto el cliente

depende del vendedor, esto se puede expresar de la siguiente manera en los modelos pata de gallo y Chen (Fig. 32):

Modelo Pata de Gallo



Modelo de Chen

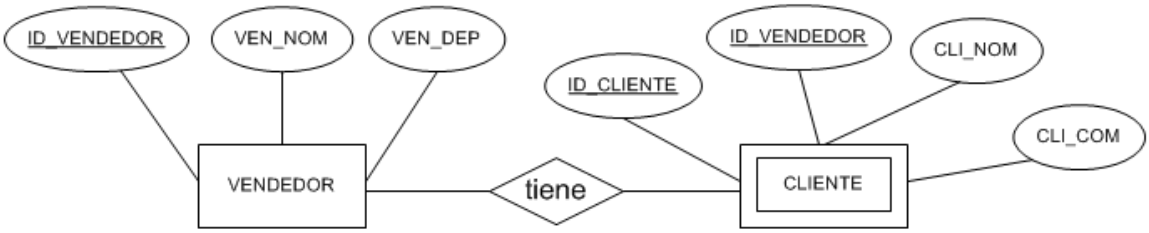


Fig. 32 Entidad débil

En la (Fig. 32) en el primer diagrama para identificar una entidad débil la línea de relación contiene un circulo del lado de la entidad débil, en el modelo Chen se coloca un rectángulo doble en la entidad débil.

- *Atributos:* sirven para calificar, identificar, clasificar o expresar las características de las entidades, estos son representados con óvalos en el modelo Chen mientras que en el modelo pata de gallo se representan simplemente con una caja en la que se escriben los atributos como una lista la cual se encuentra debajo del rectángulo de la entidad (Fig. 33).

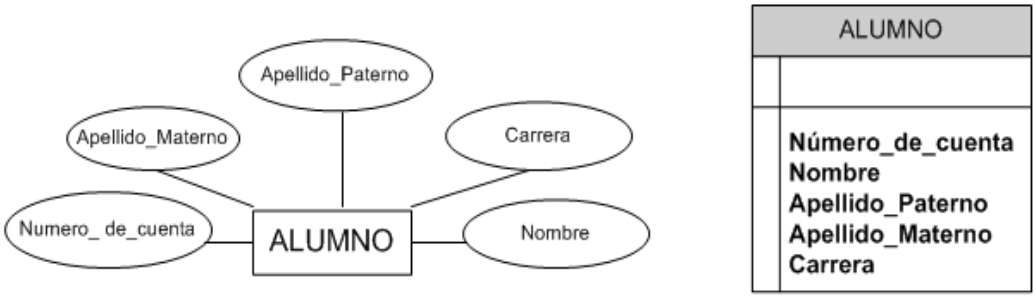


Fig. 33 Modelo Chen y Pata de Gallo para entidades y atributos

En seguida se muestran las características que pueden tener los atributos dentro de una base de datos:

- *Dominio:* todos los tributos cuentan con un dominio el cual define los posibles valores que pueden tomar este, por ejemplo si tenemos el atributo sexo al establecer el dominio se puede indicar que este sólo acepte las letra “M” o “F” según sea el caso, otro ejemplo puede ser el atributo edad el cual sólo puede aceptar números enteros entre el 1 y 110.
- *Claves primarias:* un atributo puede ser una llave primaria, la cual dentro de los diagramas debe ir subrayada, si existe una clave compuesta todos los atributos que la representan deben estar subrayados, en algunos casos se identifica también cuando se colocan las iniciales PK (Primary Key o Clave primaria) a un lado del atributo.
- *Atributo compuesto:* es aquel que puede subdividirse en más atributos, por ejemplo, el atributo domicilio, se puede dividir a su vez en: calle, número, colonia, código postal; en los diagramas del modelo Chen estos atributos se identifican con una doble línea que va del rectángulo de la entidad al óvalo del atributo, en el modelo pata de gallo no se pueden identificar.
- *Atributo simple:* al contrario del anterior este ya no puede dividirse más, por ejemplo el atributo apellido_paterno.
- *Atributo de un solo valor:* es aquel que sólo puede tomar un valor, por ejemplo, el número de cuenta es un valor único para un estudiante así como el número de seguro social para un empleado.
- *Atributo múltiple:* son los atributos que pueden tomar varios valores, por ejemplo, una persona puede tener varios grados escolares.

Un atributo mal asignado puede hacer que la base de datos falle por eso es conveniente hacer un análisis exhaustivo del las asignaciones que se les dará a cada uno.

- *Relaciones:* son verbos que indican la asociación que existe entre dos entidades. Existen tres tipos de relaciones que pueden ser utilizadas para conectar una tabla con otra (Batini, 1994, págs. 361-370):

- *Relación uno a uno* (1:1): esta relación indica que un elemento de la tabla A tiene únicamente relación con un elemento de la tabla B (Fig. 34 y 35).

Representación para el diagrama E-R del Modelo de Chen (1:1)

Dentro del modelo Chen las relaciones se representan con un rombo dentro del cual se escribe en minúsculas el verbo que indica la relación que existe entre las entidades (Fig. 34).

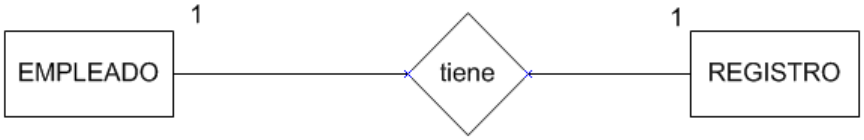


Fig. 34 Relación uno a uno (1:1)

Esta relación se reconoce cuando del lado de cada entidad se coloca el número “1”, cuando se tienen este tipo de relaciones pareciera que las entidades no debieran estar separadas, pero existen excepciones, por ejemplo, si se tiene una base de datos que contiene los registros de varios tipos de empleados (pilotos, licenciados, ingenieros, etc.), no se puede crear una sola tabla de registro de empleados ya que quedarían muchos atributos nulos pues un piloto no maneja las mismas características que un ingeniero, ya que el piloto tal vez tenga en su registro el número de horas de vuelo, el tipo de aeronaves que maneja, etc., mientras que el ingeniero registre el número de obras construidas, el tipo de materiales que maneja etc., por eso en este caso se puede decir que cada empleado sólo puede tener un registro y que cada registro puede ser de un empleado solamente.

Representación para el diagrama E-R del modelo pata de gallo (1:1)

Para este modelo el verbo que indica la relación simplemente se escribe en minúsculas arriba, abajo, o sobre la línea que indica la relación entre entidades. Se utilizaran los mismos ejemplos que en el modelo Chen pero con la nomenclatura del modelo pata de gallo.

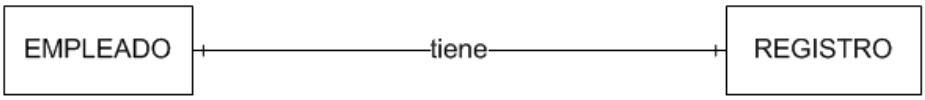


Fig. 35 Relación uno a uno (1:1)

Para indicar esta relación simplemente se cruza una pequeña línea sobre la línea que indica la relación (Fig. 35).

- *Relación uno a muchos* (1:M): cada instancia de la entidad A se relaciona con varias instancias de la entidad B (Fig. 36 y 37).

Representación para el diagrama E-R del Modelo de Chen (1:M)

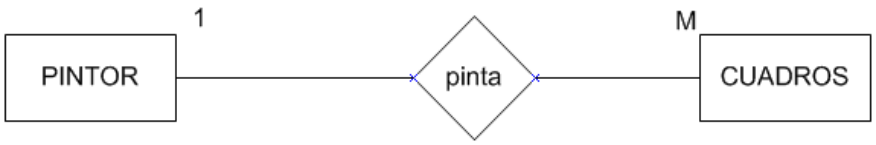


Fig. 36 Relación uno a muchos (1:M)

Para este caso en el modelo Chen la parte que se representa como muchos se indica con una “M”, en este ejemplo se dice que un pintor puede pintar muchos cuadros, mientras que un cuadro sólo puede ser pintado por un pintor (Fig. 36).

Representación para el diagrama E-R del modelo pata de gallo (1:M)

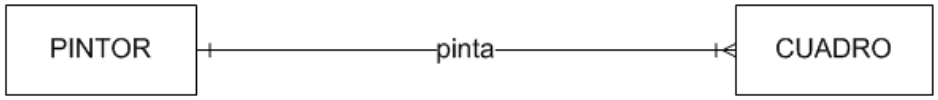


Fig. 37 Relación uno a muchos (1:M)

En esta relación para indicar el lado muchos se hace con una línea de la cual salen tres dedos (de allí el nombre pata de gallo) (Fig. 37).

- *Relación muchos a muchos* (M:N): cualquier instancia de la entidad A se relaciona con cualquier instancia de la entidad B (Fig. 38 y 39).

Representación para el diagrama E-R del Modelo de Chen (M:N)

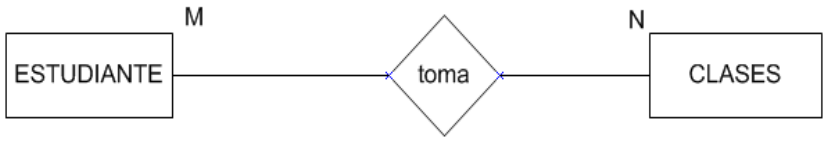


Fig. 38 Relación muchos a muchos (M:N)

En esta relación se utilizan las letras “M” en un extremo y “N” en el otro para indicar una relación muchos a muchos, en el ejemplo se observa que un estudiante puede tomar muchas clases y que una clase puede tener muchos alumnos (Fig. 38).

Representación para el diagrama E-R del modelo pata de gallo (M:N)

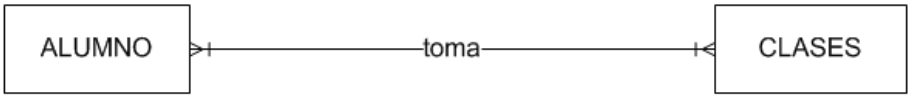


Fig. 39 Relación uno a muchos (1:M)

La relación muchos a muchos se indican con los tres dedos en ambos lados de la línea de relación (Fig. 39).

- *Relación Fuerte:* también conocidas como *relaciones identificadoras* son dependientes de la existencia, esto quiere decir que una entidad no puede existir sin que la entidad padre este presente, ya que de las dos entidades relacionadas

una debe contener la clave primaria de la otra. Por ejemplo, se tienen las entidades CURSO y CLASE, dentro de la entidad CLASE se encuentra la clave primaria de CURSO, por tanto la clave primaria de CLASE es compuesta por la clave primaria de CURSO y la clave primaria de CLASE, esta es una relación fuerte ya que si no existe un curso no puede existir una clase. Dentro del diseño de bases de datos para este caso se tendría que crear primero la tabla CURSO ya que si no es así se podría caer en un error de integridad ya que al final de cuentas el atributo ID_CURSO (clave primaria de CURSO) no podría heredarse en la tabla CLASE puesto que no existe (Fig. 40).

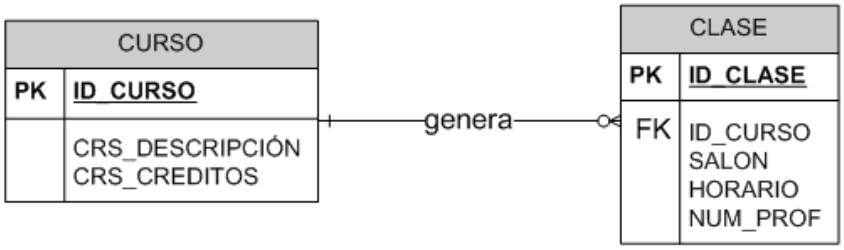


Fig. 40 Relación fuerte

En la (Fig. 40) en el modelo pata de gallo en la línea de relación del lado de la entidad dependiente se coloca un pequeño círculo, el modelo Chen no distingue las relaciones fuertes.

- *Relación Débil:* estas se crean cuando una entidad es independiente de otra con la cual está relacionada, suponiendo el mismo ejemplo que en las relaciones fuertes, la clase puede existir sin la necesidad de que exista un curso específico, la entidad CLASE contiene la clave primaria de la entidad CURSO pero no es necesario que el curso exista para que se pueda crear CLASE (Fig.41), tanto las entidades débiles como las fuertes deben ser analizadas por el diseñador de la base de datos ya que como se observa un mismo problema puede ser tratado de diferente forma dependiendo de las necesidades de los usuarios.

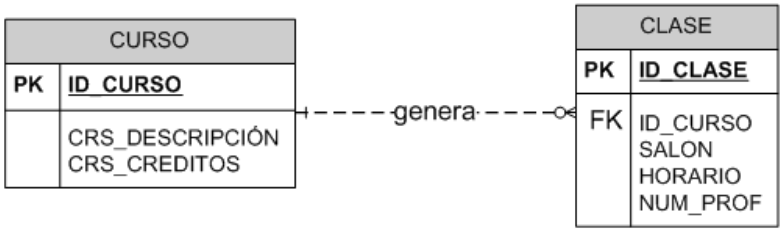
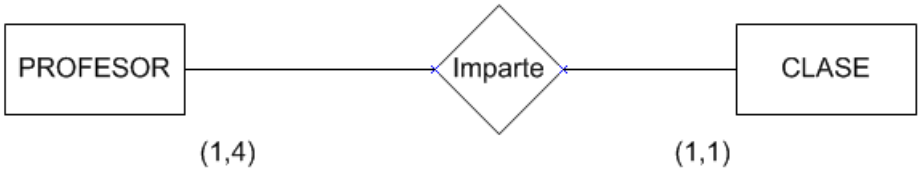


Fig. 41 Relación débil

En la (Fig. 41) las entidades débiles se identifican en el modelo de pata de gallo por que la línea de relación entre entidades es punteada y además tiene un pequeño círculo al final de la línea del lado de la entidad débil.

- *Cardinalidad:* representa el número específico de ocurrencias de una entidad relacionada. En el diagrama del modelo Chen se expresa como (x,y) mientras que en diagrama del modelo pata de gallo no se específica, por ejemplo (Fig. 42):

Modelo de Chen



Modelo pata de gallo

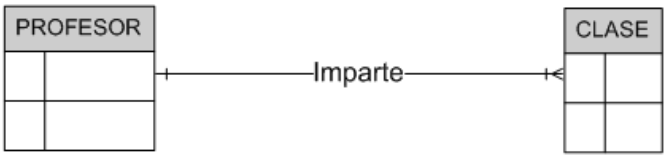


Fig. 42 Cardinalidad

Como se observa en la (Fig. 42) en el diagrama del modelo Chen la representación de la cardinalidad se hace colocando entre paréntesis el número de ocurrencias que pueden existir en cada entidad para este caso se observa que un profesor puede impartir como máximo cuatro clases mientras que una clase sólo puede ser impartida por un profesor. Por otra parte en el diagrama de pata de gallo no se indica la cardinalidad sólo se indica la relación que existe entre entidades, en este caso un profesor puede impartir varias clases. Por lo anterior se concluye que el diagrama entidad-relación es muy útil para comprender de manera conceptual la estructura de una base de datos, esto permite un mayor control sobre el almacenamiento de datos ya que se sabe cómo y en dónde se están colocando.

4.4 Redes de datos y seguridad

Las *redes de datos* se crean por la necesidad que tiene el ser humano de transmitir y recibir información de manera rápida y eficiente haciendo uso de recursos informáticos, pero, ¿qué pasa con esa información cuando se traslada de un lugar a otro?, ¿alguien más la puede observar?, para resolver estos problemas existe las *redes de datos* y la *seguridad informática.*

En esta sección se abundará sobre las redes de datos, se definirán de manera concreta y se explicará su funcionamiento, además se presentarán las técnicas y algunos consejos básicos sobre seguridad para administrar una red de computadoras, y se estudiarán algunos métodos de encriptación de datos. Se encuentra en el (Apéndice IV) una breve historia de las redes de datos.

4.4.1 Fundamentos de redes de datos

4.4.1.1 Conceptos básicos

Para comprender que es una red de computadoras primero se deben estudiar algunos conceptos, por lo que se comenzará por definir que es un *dato,* según el diccionario de la Real Academia de la Lengua Española (DRAE, 2010, s/p) lo define como “un antecedente necesario para llegar al conocimiento exacto de algo o para deducir las consecuencias legítimas de un hecho”, en otras palabras son cifras o hechos crudos que no han sido analizados. Tomado como referencia la definición anterior se deduce que un conjunto de datos organizados y analizados son denominados *información*, la información puede ser transmitida por medio de una *red de computadoras*.

Una *red de computadoras* es una colección interconectada de dos o más computadoras autónomas que intercambian información (Tanenbaum, 2003, pág. 2). El objetivo de una *red* es compartir recursos tanto físicos (discos duros, impresoras, periféricos, etc.) como lógicos (programas, información, software, etc.), otro objetivo es que se obtienen ventajas económicas pues los costos para mantener la comunicación dentro de una organización se reducen al utilizar una red de computadoras.

Las red de computadoras deben seguir *protocolos* para el intercambio de información, los *protocolos* se definen como “reglas o normas que se deben seguir para lograr un

objetivo", para el caso particular de las redes, el *protocolo* toma la misma definición, pero el objetivo común es establecer comunicación entre las computadoras.

Según (Verderber & Co., 2009, pág. 4) para que se pueda establecer *comunicación* entre dos entidades se deben tener los siguientes elementos: participantes (quiénes), los mensajes (qué), el contexto (dónde), los canales (cómo), la presencia o ausencia de ruido y la retroalimentación (Fig. 43).

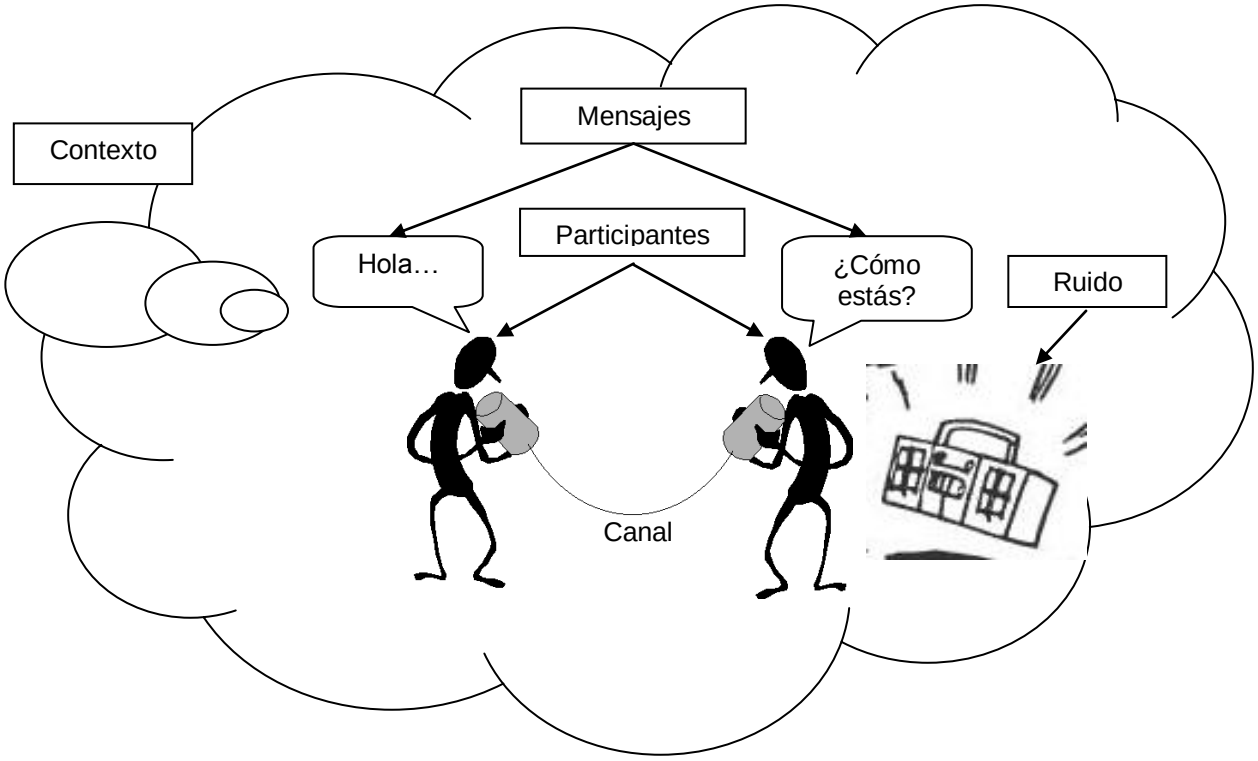


Fig. 43 Comunicación

Para las redes de computadoras la explicación anterior se puede acoplar de la siguiente manera: los participantes son los *nodos o computadoras*; el mensaje es la *información* que se desea transmitir; los *canales* pueden ser *alambre de cobre, fibra óptica o el aire*; la presencia de *ruido* en el *medio ambiente*, por ejemplo, el magnetismo que pueda existir alrededor del canal de transmisión entre otros; y la retroalimentación que es *la respuesta* que da un nodo a la información recibida.

Por tanto la comunicación es la parte esencial de una red de computadoras.

4.4.1.2 Topología de redes alámbricas

Según (Herrera, 2003, págs. 65-67) la *topología* se refiere a como se realiza la conectividad física de la red, en otras palabras es la "es la forma geométrica en cómo están acomodadas las estaciones de trabajo". Existen tres formas básicas:

- *Topología lineal (Fig. 44):* todas las estaciones de trabajo se conectan a un mismo medio denominado BUS, por lo general el cable de transmisión es cable coaxial, su instalación es de las más sencillas, si una terminal es desconectada la información sigue fluyendo. Las desventajas que presenta esta topología es que si el medio de transmisión principal se rompe la red deja de funcionar, además si dos computadoras desean enviar información al mismo tiempo se genera una *colisión* lo que impide el correcto funcionamiento de la red, por tanto debe existir una sincronización para el envío de datos.

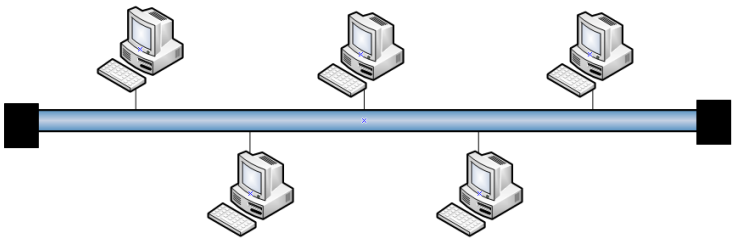


Fig. 44 Topología lineal

- *Topología en anillo (Fig. 45):* las estaciones de trabajo se encuentran conectadas en forma de anillo esto quiere decir que cada computadora está conectada con otras dos, la información es transmitida en un solo sentido y pasa por los nodos que se encuentran entre el nodo transmisor y el receptor, esta configuración permite el ahorro del medio físico de transmisión pues se requiere muy poco para realizar las conexiones, la desventaja de esta topología es que si el canal de transmisión se rompe toda la red se paraliza.

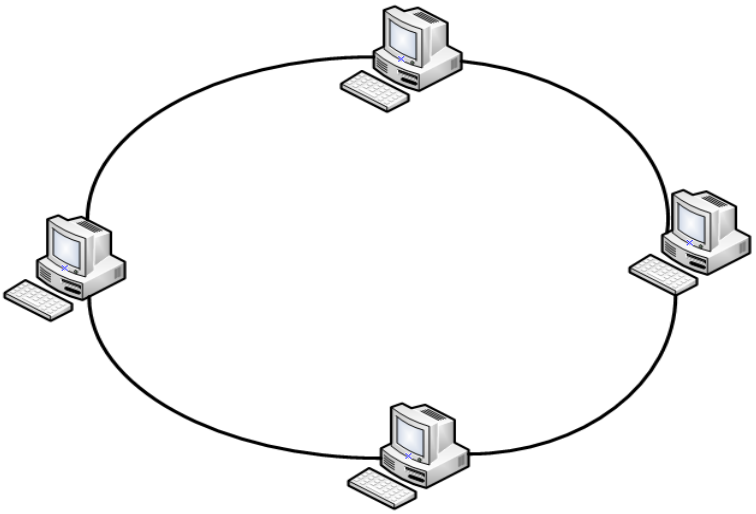


Fig. 45 Topología en anillo

- *Topología en estrella (Fig. 46)*: esta configuración consiste en conectar todas las estaciones de trabajo a un concentrador, en esta topología no pasa nada con la red si el cable de alguna máquina se rompe o si la computadora deja de trabajar, el concentrador se dedica a administrar el paso de la información y si este falla la red deja de funcionar, esta técnica es más costosa que las anteriores, pues se utiliza mucho más cable para su instalación, pero es una de las más eficientes y de las más utilizadas.

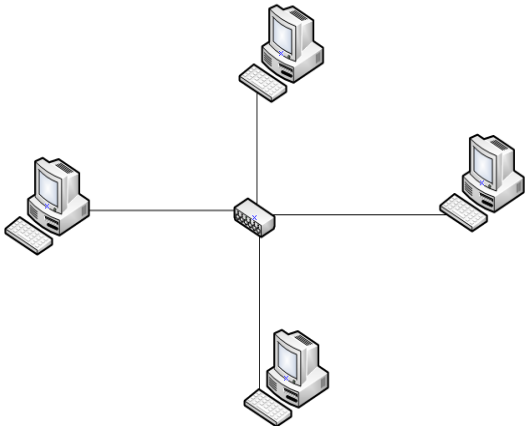


Fig. 46 Topología en estrella

4.4.1.3 Topología de redes inalámbricas

Así como las redes alámbricas tienen diferentes formas de conectarse, las redes inalámbricas cuentan con dos configuraciones básicas (Microsoft, 2010, s/p):

- *Modo Infraestructura:* para este tipo de conexión se requiere de un *Punto de Acceso* (AP, por sus siglas en inglés), el AP es un dispositivo que está conectado a una red alámbrica, el cual puede realizar el intercambio de información entre computadoras que cuenten con una tarjeta de red inalámbrica, la ventaja de este tipo de conexiones es que el usuario tiene libertad completa para moverse de un lugar a otro con su equipo de cómputo, siempre y cuando se encuentre en el rango de cobertura del AP (Fig. 47).

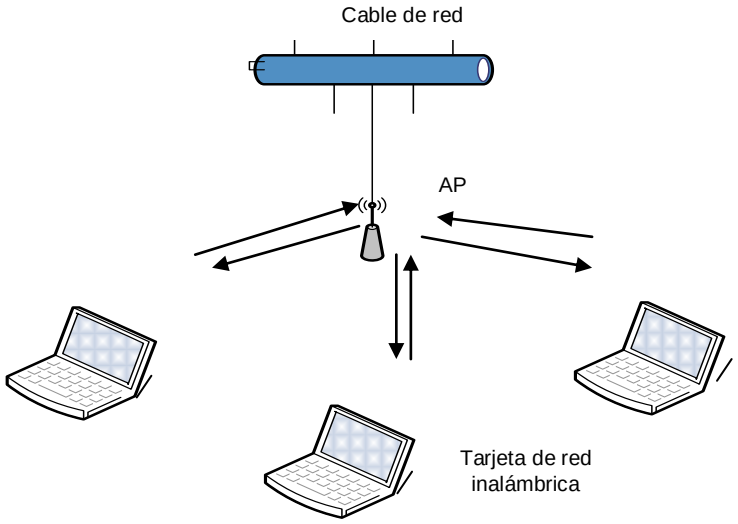


Fig. 47 Modo Infraestructura

- *Modo Ad Hoc:* esta configuración es la más sencilla y se caracteriza por equipos individuales que cuentan con tarjetas de red inalámbrica comparten información entre ellos (Fig. 48).

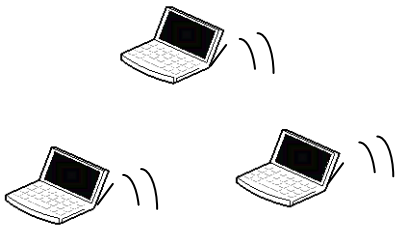


Fig. 48 Modo Ad Hoc

4.4.1.4 Clasificación de las redes

Las redes se clasifican según el área geográfica que cubren:

- *Red de Área Personal (PAN por sus siglas en inglés)*: cubre un espacio pequeño, como una oficina.
- *Red de Área Local (LAN por sus siglas en inglés):* abarca generalmente un edificio.
- *Red de Área Campus (CAN por sus siglas en inglés):* cubre generalmente varios edificios sin rebasar los diez kilómetros.
- *Red de Área Metropolitana (MAN por sus siglas en inglés):* engloba toda una ciudad sin exceder los cien kilómetros.
- *Red de Área Amplia (WAN por sus siglas en inglés):* puede concentrar una red que cubra todo un país o continente.
- *Internet*: cubre todo el planeta.

Este es el caso de las redes alámbricas, si se manejan redes inalámbricas a los términos antes definidos se les debe agregar una "W" al principio que significa "Wireless", por ejemplo, WLAN seria Red Inalámbrica de Área Local.

4.4.1.5 Ventajas y desventajas redes inalámbrica vs redes alámbricas

En esta sección se hará referencia a las redes inalámbricas como (RI) y a las redes alámbricas como (RA), enseguida se enlistan las ventajas y desventajas entre ambas tecnologías (Andreu, 2010, pág. 212):

- Existe una *rápida instalación* en una RI, no se requiere de mucho cableado a diferencia de las RA en las que se requiere hasta de modificaciones estructurales en las habitaciones o edificios donde se desea instalar la red.
- Las RI permiten *movilidad,* si un usuario conectado a la red desea moverse de lugar puede hacerlo sin ninguna dificultad por medio de las RI, comparado con las RA en donde para seguir en la red y moverse de lugar debe existir suficiente cable para cambiar o debe existir otro nodo para la conexión.

- *El costo* de las RI es menor, ya que no se requiere de tanta infraestructura para montar una red de este tipo.
- *Conectividad*, mientras que las RA son más eficientes frente a cambios climáticos y estructurales (edificios, paredes, etc.), las RI presentan mayores problemas, ya que como con las RI la información viaja a través del aire cualquier cambio climático u obstáculo entre el AP y la computadora puede haber deficiencias y hasta interrupciones en el servicio.
- *Velocidad*, las RA pueden mantener una velocidad y conectividad constante y mucho mayor que con una RI.
- *Mantenimiento,* para dar mantenimiento a una RI se requiere de menor tiempo y dinero.
- *La seguridad*, en la actualidad las RI no son muy seguras, por tanto es importante mencionar que esta característica es la más importante a la hora de tomar una decisión para la selección del tipo de red que se desea manejar, ya que si la información que se desea transmitir es crítica y de alta prioridad es recomendable el *USO de una RA* pues al ser esta una red que transmite a partir de medios físicos estos pueden ser controlados con mayor eficiencia, pues a diferencia de las RI como la información es transmitida por el aire, cualquier persona puede captúrala con mayor facilidad.

De los puntos anteriores se puede concluir que dependiendo de las necesidades del usuario se debe seleccionar el tipo de red. Para el caso particular de este trabajo, como se manejará información crítica, se seccionará el manejo de una red alámbrica.

4.4.1.6 Modelo OSI

Según (Forouzan, 2002, págs. 41-54), la Organización Internacional de Estandarización (ISO por sus siglas en inglés) crea el modelo de *Interconexión de Sistemas Abiertos* (OSI por sus siglas en inglés) con la finalidad de que los sistemas existentes en la red de redes pudieran conectarse y comunicarse independientemente de su arquitectura de software y de hardware. El modelo OSI no es un protocolo, como su nombre lo indica es un modelo que sirve para diseñar una red flexible, robusta e interoperable.

El modelo está formado por siete capas o niveles los cuales se pueden apreciar en la (Fig. 49):

7	APLICACIÓN
6	PRESENTACIÓN
5	SESIÓN
4	TRANSPORTE
3	RED
2	ENLACE
1	FÍSICA

Fig. 49 Capas del Modelo OSI

Capa 1. Nivel Físico: coordina las funciones necesarias para transmitir los datos por medios físicos, además es responsable de establecer la conexión inicial y de interrumpirla cuando se termina la comunicación, también se encarga de definir el tipo de codificación de los bits para que estos pueden ser transmitidos. En esta capa se encuentra el medio físico de transmisión (conector, concentrador, etc.).

Capa 2. Nivel de Enlace: su función principal es la de transformar el medio físico, en un enlace confiable y sin errores de transmisión, esta capa divide el grupo de bits enviados por el emisor en tramas (paquete de datos que contiene una cabecera, datos y una cola), cada una de las cuales contiene la dirección de origen y destino final, en esta capa se evita el duplicado de tramas y se tiene un control de errores para retransmitir las tramas pérdidas o defectuosas.

Capa 3. Nivel de Red: su trabajo es entregar un paquete de su origen a su destino, para lograr su objetivo controla el congestionamiento de paquetes, mientras que la capa de enlace otorga una dirección local a la trama, la capa de red se encarga de dar una dirección lógica, la cual permitirá salir a la trama de la red de origen y ser identificada en otras redes.

Capa 4. Nivel de Transporte: su tarea es entregar el mensaje completo a su destinatario, mientras que la capa de red realiza la entrega de tramas de su origen a su destino tratándolas a cada uno de forma individual como si fueran un mensaje, la capa de transporte se encarga de supervisar que el mensaje completo llegue intacto.

Capa 5. Nivel de Sesión: es el controlador de diálogo de la red. Permite que los usuarios de diferentes máquinas establezcan comunicación entre ellos, controlando la sincronía y la gestión del diálogo.
Capa 6. Nivel de Presentación: se encarga de la sintaxis y semántica de la información que se transmite, está relacionada con los aspectos de representación de la información, comprensión de datos, criptografía, etc.

Capa 7. Nivel de Aplicación: su función es la de proporcionar las interfaces de usuario final y el soporte de los servicios como el correo electrónico, la transferencia de archivos, el uso de páginas http, la gestión de datos y servicios de información distribuida.

4.4.1.7 Modelo TCP/IP

El modelo Transmision Control Protocol / Internet Protocol (TCP/IP) es un conjunto de protocolos estandarizados de la industria del Internet que permite la comunicación a diferentes ambientes de redes, el modelo es abierto ya que no está manejado por una sola organización.
Como se menciona en la reseña histórica (ver Apéndice IV), este modelo fue creado por el Departamento de Seguridad de los Estados Unidos, con la finalidad de obtener una red en la cual se pudieran transmitir datos de un emisor a un receptor sin depender del estado de la red, esto quiere decir, que mientras las dos entidades se encontraran trabajando la información podría ser enviada y recibida, para el Departamento de Defensa era imprescindible lograr su objetivo pues así la información manejada en la red seria integra y segura. El protocolo era manejado en una red denominada ARPANET que posteriormente pasaría a ser el Internet que se conoce hoy en día.
Este modelo está diseñado en cuatro capas (Hunt, 2002, págs. 9-22), aunque algunos autores llegan a manejar hasta cinco capas, enseguida se muestra un esquema de este modelo (Fig.50):

Capa de Aplicación
Capa de Transporte
Capa de Internet
Capa de Acceso a la Red
Capa Física (manejada por algunos autores)

Fig. 50 Capas del Modelo TCP/IP

Capa de aplicación: esta capa es la más alta del modelo y contiene todas las aplicaciones necesarias para tener acceso a la red, tales como: el protocolo http, imperioso para la visualización de páginas web; el File Tranfer Protocol (FTP), utilizado en la transferencia de archivos; Secure Shell (SSH), creado para establecer conexiones remotas; Simple Mail Transfer Protocol (SMTP), necesario en el uso de servicios de correo electrónico; Simple Network Management Protocol (SNMP), requerido en la administración de dispositivos de la red; etc.

Capa de transporte: garantiza la comunicación entre computadoras, esto quiere decir que los paquetes enviados sean recibidos sin errores por el receptor, esta capa contiene dos protocolos que controlan la forma en que serán enviados los datos, se utiliza el que más convenga.

- *Transmision Control Protocol (TCP)*: éste garantiza la entrega de los datos por medio de un certificado.
- *User Datagram Protocol (UDP)*: provee de un envió rápido de los datos pero no garantiza su entrega.
- *Capa de Internet*: es la encargada de direccionar, empaquetar y enrutar los datos transmitidos, se manejan cuatro protocolos dentro de esta capa: el Internet Protocol (IP), responsable de otorgar una dirección a los datos que son transmitidos para que lleguen a su destino; el Address Resolution Protocol (ARP), desarrollado para identificar la dirección física del medio (MAC-Media Access Control); el Internet Control Message Protocol (ICMP), utilizado en la captura y reporte de los errores generados en caso de existir problemas con el envío de

datos; y el Internet Group Management Protocol (IGMP), el cual maneja la administración del Multicasting con el TCP/IP.

- *Capa de Red:* es la responsable del intercambio de datos entre dos sistemas conectados a la red, esta capa contiene estándares como Ethernet, Asynchronous Transfer Mode (ATM), etc. Los cuales indican cómo serán transmitidos los datos sobre la red.
- *Capa Física*: como se indicó esta capa sólo la mencionan algunos autores, y no está muy bien detallada, sólo se indica que define las características del medio de transmisión, la codificación de los datos, la velocidad de transmisión, etc.

El estudio de estos modelos apoya a la comprensión del funcionamiento de las redes de datos y del Internet en general.

4.4.2 Fundamentos de seguridad

Cuando se crearon las redes de datos el principal objetivo era que la información enviada por la red llegará a su receptor de manera íntegra, no se hizo énfasis en si alguien la podía interceptar o no, pues la información manejada se encontraba dentro de una red privada, pero conforme se ha desarrollado el crecimiento de las redes y la apertura de la información a miles de personas en Internet, la *seguridad informática* se volvió un tema de mayor importancia, pues se paso del envió de simples correos electrónicos a manejar grandes cantidades de información, las cuales pueden contener desde información turística hasta las claves secretas de cuentas bancarias. Por esta razón se empezó a estudiar la *seguridad* de los sistemas informáticos, la cual se encarga en términos generales de *evitar que la información enviada a través de una red de computadoras sea vista o interceptada por un desconocido o intruso*. La información es poder y debe ser controlada únicamente por las personas interesadas.

Un sistema de información puede sufrir daños en su estructura original lo cual es denominado *ataque*. Los ataques se clasifican en pasivos o activos los cuales se definen de la siguiente manera según (Álvarez, 2000, s/p):

Para poder explicar el tipo de ataques que se mencionan, primero se debe tomar en cuenta que el flujo normal de información es llevado de la siguiente manera (Fig. 51):

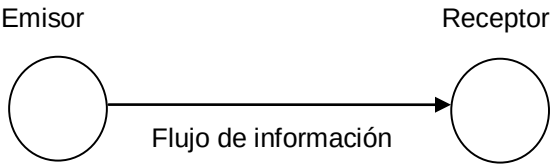


Fig. 51 Flujo normal de información

Un *ataque pasivo* se da en forma de escucha u observación, el objetivo del intruso es obtener información del sistema sin afectar su funcionamiento, para este tipo de ataque existe una amenaza denominada *intercepción de información* en la cual el atacante solo observa la información que pasa a través de la red para extraerla sin modificarla y sin evitar que llegue a su destinatario, esto atenta contra la *confidencialidad* (Fig. 52):

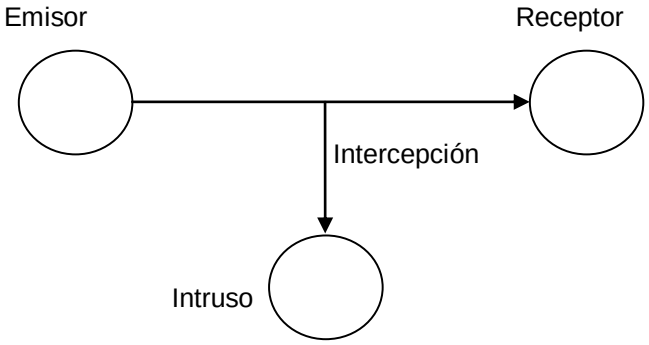


Fig. 52 Ataque pasivo de intercepción

Por otra parte un *ataque activo* se lleva a cabo cuando el perpetrador (intruso) modifica el flujo normal de datos o altera el funcionamiento normal del sistema, existen tres amenazas generales que pueden concluir en un ataque activo:

- *Interrupción*: su objetivo es destruir algún elemento del sistema para eliminar el flujo de datos, esto afecta la *disponibilidad* de los servicios (Fig. 53).

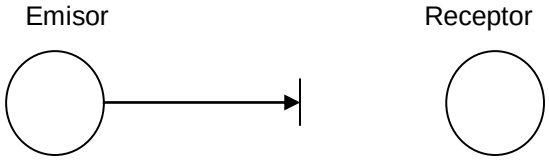


Fig. 53 Ataque de Interrupción

- *Suplantación:* este tipo de ataque se genera cuando un intruso envía información en nombre de otra persona, afectando así la *autenticación* de los usuarios de un sistema (Fig. 54).

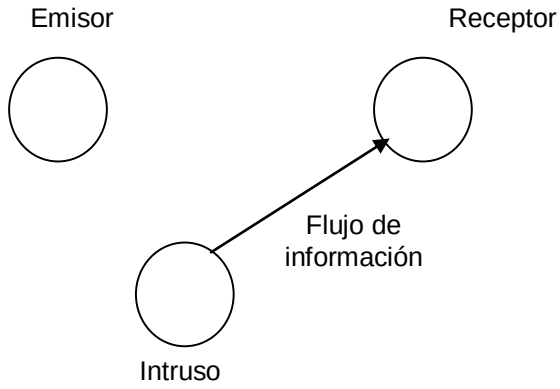


Fig. 54 Ataque de Interrupción

- *Modificación*: se genera cuando un atacante intercepta la información la modifica y la envía a su destinatario, afectando así la *integridad* del flujo de información (Fig. 55).

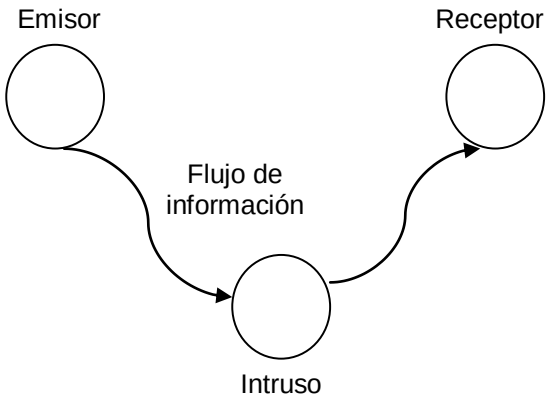


Fig. 55 Ataque de modificación

Como se observa los ataques en una red de datos se centran en: *la confidencialidad, la disponibilidad, la autenticación y la integridad*. El objetivo de la seguridad informática es mantener estos cuatro puntos fundamentales fuera de peligro, además de garantizar el *no repudio*, para cumplir este objetivo se requiere un servicio de seguridad para cada punto vulnerable, esto garantiza el resguardo adecuado del sistema y de la transferencia de datos.

En seguida se enlistan los servicios generales según las recomendaciones de la Unión Internacional de Telecomunicaciones (ITU, por sus siglas en inglés) en su documento (X.800, 1991, s/p):

- *Confidencialidad*: característica de la información que no está disponible ni es divulgada a personas no autorizadas.
- *Disponibilidad:* es la propiedad que debe tener un sistema, este de ser accesible en el momento que una entidad interesada lo desee.
- *Autenticación:* mecanismo utilizado para garantizar que las entidades que intervienen en el proceso de comunicación son quien dicen ser, por tanto el control de acceso pertenece a este servicio.
- *Integridad*: garantiza que los datos contenidos en un sistema o en un flujo de datos no sean alterados o destruidos sin autorización previa.
- *No repudio*: esta propiedad garantiza que ni el emisor ni el receptor nieguen el envió de un mensaje, esto quiere decir que el emisor puede comprobar que le mensaje a llegado a su destinatario y que el receptor verifique que el mensaje ha sido enviado por el emisor.

Según (Stallings, 2007, págs. 17-18), las técnicas empleadas en los servicios de seguridad tienen dos componentes: el primero es una transformación relacionada con la seguridad de la información que se va a enviar, por ejemplo, el cifrado de los mensajes, en donde se desordena la información para que sea ilegible para un intruso (oponente); y el segundo es el intercambio de información secreta por los interlocutores y desconocida por el intruso, un ejemplo puede ser el uso de una clave para el cifrado de los datos antes de enviarlos y otra para descifrarlo al momento de recibirlo (Fig. 56).

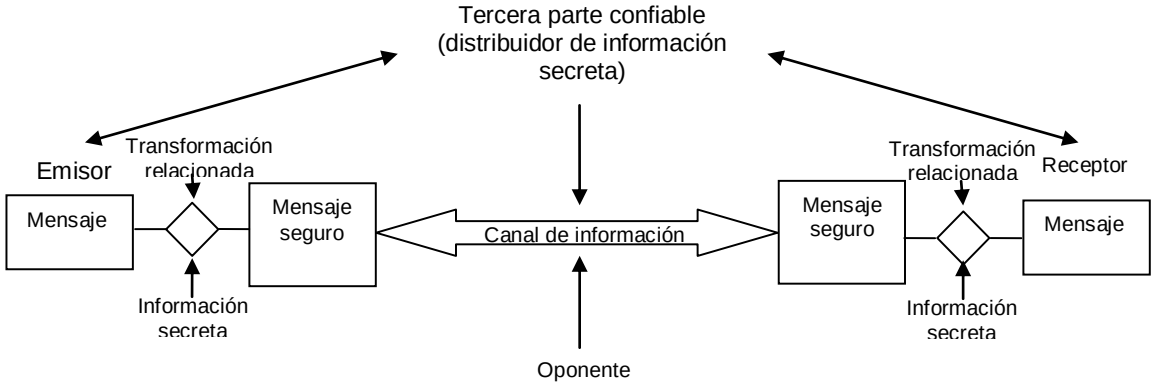


Fig. 56 Modelo general para la seguridad

La información enviada por un canal de comunicación debe cumplir con las características citadas anteriormente para que se pueda considerar que un mensaje está protegido y seguro.

4.4.3 Algoritmos de cifrado

En la (Fig. 56) para lograr que un mensaje sea seguro, la informática recurre a la *criptografía* la cual es la encargada de proporcionar los dos componentes esenciales de los servicios de seguridad (transformación relacionada e información secreta).
Según (Ramio, 2006, 39) la *criptografía* es la ciencia que conjunta métodos y técnicas para transformar un mensaje inteligible (llamado generalmente *texto llano*), para que sólo pueda ser comprendido por personas que estén autorizadas a hacerlo, y que es denominado *texto cifrado*, el proceso de cifrar el *texto llano* requiere de un conjunto de reglas preestablecidas entre quienes se comunican, las cuales se designan como *claves*.
La ciencia dedicada a descifrar los mensajes sin tener la clave del *texto cifrado* es denominada *criptoanálisis*, en conjunto el criptoanálisis, la criptografía son llamados *criptología*.
Existen distintos *sistemas criptográficos* los cuales conjuntan una serie de reglas para obtener claves para cifrar y descifrar un mensaje, estos sistemas se basan en tres técnicas el *cifrado simétrico o de clave secreta*, el *asimétrico o de clave pública* y *autentificación de mensajes*.

4.4.3.1 Cifrado simétrico o de clave secreta

Según (Tanenbaum, 2003, págs. 737-746) el *cifrado simétrico o de clave secreta* es denominado así porque usa la misma clave tanto para cifrar como descifrar un mensaje, los algoritmos más comúnmente usados para esta técnica utilizan bloques de texto, esto quiere decir que se toman bloques siempre del mismo número de bits y se genera un bloque cifrado del mismo tamaño, los cifradores de bloques más utilizados son:

- *Data Encrytion Estándar (DES):* es un algoritmo utilizado desde hace más de 30 años usado generalmente en la Banca su antecesor fue *Lucifer* un algoritmo creado por Feister en IBM. DES cifra bloques de 64 bits, con claves de 64 bits,

está regido por la norma X3.92 el cual describe al algoritmo y la norma X3.108 donde se describen los modos de operación según el ANSI, su objetivo principal es evitar que se descubra la clave utilizada para cifrar.

- *Triple DES*: funciona de la misma manera que el DES pero con un triple cifrado usando tres claves distintas, esto da como resultado una clave de 168 bits lo que proporciona una mayor seguridad, el tamaño de esta clave evita hace casi imposible ataques por fuerza bruta.
- *Advanced Encryption Standar (AES):* en 1997 la National Institute of Standards and Technology (NIST) llama a un concurso público para presentar un nuevo algoritmo estándar, que sea eficiente en su implementación tanto en software como en hardware, los investigadores Vincent *Rij*men & Joan *Dae*men presentan el algoritmo *Rijndael* el cual en noviembre de 2001 se vuelve el algoritmo estándar del NIST. El tamaño de clave utilizado del AES utiliza claves que varían de 128, 192, 256 bits o bien múltiplos de 4 bits, utiliza bloques de 128 bits, no utiliza la estructura presentada por Filser, su manejo es más efectivo en su implementación que el DES.

A pesar de que el NIST ya no certifica al Triple DES como su algoritmo estándar, este sigue siendo uno de los más utilizado dentro de las redes de computadoras para proteger sus datos, por otra parte el algoritmo AES cada año adquiere mayor fuerza por el potencial que presenta en su implementación, cabe señalar que ninguno de los algoritmos ha sido vulnerado de manera efectiva.

4.4.3.2 Cifrado asimétrico o de clave pública

Por otra parte el *cifrado asimétrico o de clave pública* es denominado así por que utiliza dos claves para su implementación, utiliza una clave pública y una privada, la clave pública puede ser vista por cualquier persona, es utilizada como un candado, esto quiere decir que cualquier persona que desee enviar información confidencial al dueño de la clave pública puede hacerlo utilizando esta para cifrar el mensaje, cuando el dueño recibe el mensaje el únicamente podrá descifrarlo ya que tiene la clave privada la cual sirve como llave para abrir el candado puesto a la información, los algoritmos más utilizados para este tipo de técnicas son el *Diffie y Hellman y* el *RSA.*

- *Diffie y Hellman* crean el primer sistema criptográfico de clave pública su algoritmo tiene únicamente la finalidad de intercambiar claves para el cifrado posterior de mensajes. El algoritmo se basa en la dificultad de computar algoritmos discretos, su funcionamiento es básicamente el siguiente, se tiene un número primo *a* y un número entero *b* que es una raíz primitiva de *a*, este entero debe ser menor que *a,* para realizar el intercambio de una clave entre Juan y Pedro, Juan elige un número entero al azar, en este caso β y obtiene un número nuevo *X* aplicando $b^{\beta} mod\ (a)$ de igual manera Pedro selecciona un número entero α y aplica la misma función para obtener $Z= b^{\alpha}\ mod\ (a)$, los dos guardan en secreto tanto α como β que serán sus claves privadas y hacen público *X* y *Z*. Por último Juan utiliza el número público de Pedro en este caso Z y realiza la función $K=(Z)^{\beta}$ mod *a* y viceversa Juan realiza $K=(X)^{\alpha}\ mod\ a$ y obtienen el mismo resultado, este resultado es la clave privada que sirve para cifrar los mensajes que se enviaran Juan y Pedro de forma segura y solamente ellos conocerán, la seguridad del algoritmo se basa en el tamaño del número primo empleado y de la capacidad de cómputo que se tenga.
- Según (UV, 2005, s/p) el *algoritmo RSA* fue creado por Ronald Rivest, Adi Shamir y Leonard Adleman en el Instituto Tecnológico de Massachusetts, el algoritmo de manera básica consiste en obtener dos números primos suficientemente grandes *p* y *q* estos son multiplicados para obtener *n*, como se puede ver es muy fácil realizar la operación producto y obtener *n* pero resulta imposible factorizar *n* y obtener los números *p* y *q*, por tanto se toma *n* y se aplica la función de Euler que se denominará α(n) posteriormente se selecciona un número *e* que sea primo relativo de α(n), este número será el exponente de la clave pública *n* posteriormente se determina un número *d* que se obtiene mediante $d= 1\ mod\ (\alpha(n))$, de todos estos números obtenidos la clave pública es (*n, e*) es el módulo y el exponente de cifrado, mientras que la clave privada es (*n, d*) el módulo y el exponente de descifrado, estas claves sirven para cifrar y descifrar cualquier tipo de texto, este algoritmo es ampliamente utilizado en la red para el intercambio de información y su seguridad radica en el tamaño de *n*.

4.4.3.3 Autenticación de mensajes

Para finalizar con el análisis de algoritmos de cifrado se hablará de la *autenticación de mensajes* (Stallings, 2007, págs. 59-63), a lo largo de la historia el hombre ha tenido que autentificar documentos mediante su firma escrita, esto ha funcionado bien cuando se tiene un papel donde plasmar la firma, pero, ¿qué pasa cuando se tienen documentos electrónicos y se desean manejar en la Internet?; Se utilizan diferentes técnicas, primero se puede utilizar un método que certifique que el documento que será transferido no sea alterado de ninguna manera, para llegar a este objetivo se usan comúnmente las funciones HASH y SHA-1 los dos fueron desarrollados por el NIST, y el segundo fue una mejora del primero. Básicamente lo que hace este algoritmo es tomar como entrada cualquier tipo de archivo y dividirlo en una secuencia de *n* bits la cual es transformada en una cadena de caracteres de 160 bits única e irrepetible, esto quiere decir que si el documento es alterado en el mínimo detalle la cadena de identificación será completamente diferente, esto garantiza que el documento pueda ser considerado como legitimo.

Pero ¿cómo saber que el documento pertenece a su propietario?, para este caso se utiliza el método de cifrado *por clave pública*, en donde el usuario Pedro que desea mandar un mensaje a Claudia, Pedro cifra el mensaje con su clave privada y Claudia al recibirlo lo descifra con la clave pública de Pedro, esto garantiza que el documento pertenece a Pedro ya que este le pertenece únicamente la clave privada, el uso de esta técnica se le denomina *firma digital*.

Aunque los dos métodos permiten autentificar un documento no cubren la necesidad de cifrado, ya que los textos enviados serán llanos, esto quiere decir que no estarán cifrados.

4.4.3.4 Comparación entre algoritmos de cifrado simétrico y asimétrico

Para cerrar esta sección vale la pena especificar algunos puntos en el uso de los algoritmos de cifrado *simétrico y asimétrico*.

Siempre existe la confusión en relación a la fortaleza y seguridad que proporciona cada algoritmo, por tanto se puede indicar que los dos son muy eficientes, la seguridad del cifrado se basa en el tamaño de la clave que se utiliza y del resguardo de las mismas por parte de sus propietarios. Los algoritmos simétricos se utilizan comúnmente para cifrar información en sistemas de bases de datos mientras que los asimétricos se utilizan para

el intercambio de datos, como se puede observar cada uno cubre distintas necesidades lo que significa que ninguno sustituye al otro.

Para el presente trabajo se utilizará un algoritmo simétrico para el cifrado de datos, en el caso particular de los datos obtenidos al tomar la lectura de la huella dactilar se tomará a consideración si es necesario cifrar los datos, ya que el algoritmo que lee la huella obtiene un dato de tipo binario.

Por tanto, se debe estar consciente que la utilización depende de los requerimientos en el uso de la información.

4.5 Ingeniería de software orientada a objetos

El *software* según (Sommerville, 2005, pág. 5), se define como un conjunto de programas que interactúan entre sí, para permitir el correcto funcionamiento de un dispositivo (computadoras, electrodomésticos, etc.), estos programas van acompañados de documentos, los cuales deben describir al usuario de forma clara las actividades que estos pueden desempeñar.

Con la rápida evolución de las computadoras y de los dispositivos electrónicos en general, el software tuvo que evolucionar a la misma velocidad, esto implico escribir programas cada vez más complejos. El problema que surgió de este rápido crecimiento fue que los programadores no establecieron procesos de desarrollo específicos, ni metodologías, ni estándares, lo que ocasiono un completo caos a la hora de desarrollar software. Para resolver este problema se creó la disciplina denominada *ingeniería del software* la cual se dedica a desarrollar técnicas, métodos y herramientas para resolver los problemas relacionados con todo el proceso requerido para desarrollar e implementar un software (Campderrich, 2003, pág. 17).

Dentro del *desarrollo de software* existen diferentes paradigmas de programación, estos dictan la manera en la que se estructura un programa. Enseguida se mencionan los dos paradigmas más utilizados dentro del desarrollo de aplicaciones:

- *Programación estructurada* (Weitzenfeld, 2005, págs. 21-23): también conocida como programación tradicional o tecnología tradicional, este tipo de programación se basa en datos y funciones *globales*, esto quiere decir que los datos y funciones son visibles en todo el programa y por tanto pueden ser llamados desde cualquier parte de la aplicación. Esta manera de programar tiene sus orígenes en las primeras computadoras donde las instrucciones de un programa se guardaban en memoria creando el concepto de *programa almacenado*. En la actualidad esto no ha cambiado mucho las instrucciones se definen dentro de funciones o procedimientos, y son ejecutadas por el procesador de manera secuencial afectando los datos del programa los cuales son almacenados en otra sección de la memoria. Debido a esta separación de datos y funciones en la memoria, se han desarrollado un gran número de lenguajes de programación que explotan este concepto, sin embargo existen tres problemas principales para este tipo de programación: el primero hace que el programador deba organizar la estructura

del programa de acuerdo a la arquitectura de la computadora, de manera general *lo obliga a que piense como la máquina*, por eso este tipo de programación es clasificada como de *bajo nivel,* ya que se maneja el uso de apuntadores para el manejo de la memoria*;* en segundo lugar se basa en que los datos se vuelven *globalmente* visibles dentro de todo el programa, esto quiere decir que cualquier modificación en la estructura de los datos puede llegar a afectar a todas las funciones del programa, lo que requeriría de muchas modificaciones y mantenimiento; y tercero, indica que al tener programas muy extensos, es muy complicado para el programador encontrar un error y solucionarlo dentro de las cientos de líneas de código.

Un ejemplo muy claro de este inconveniente fue la estructura de los programas con el denominado *problema del año 2000*, cuando se quiso realizar el cambio a un simple dato como la *fecha*, que pasara ser de dos a uno de cuatro dígitos fue necesario hacer modificaciones extensas a lo largo de todos los programas lo cual implico largos y costosos periodos de mantenimiento.

- *Programación orientada a objetos (POO)* (Mcgraw-hill, 2006, s/p; Weitzenfeld, 2005, pág. 23): al contrario de la programación tradicional que se centra en los algoritmos y en la estructura del hardware, la POO se centra en los datos, en lugar de tratar de ajustar el problema al lenguaje de programación, el lenguaje de programación debe amoldarse al problema. La idea principal es crear formatos de datos que correspondan a las características del problema. Los lenguajes orientados a objetos son una combinación de *unidades o módulos*, los cuales manejan de manera individual tanto los datos, como las funciones dentro de ellas. Estas *unidades* son denominadas *objetos*, donde si se desea modificar los datos de un *objeto*, hay que realizarlo mediante las funciones específicas del objeto, esto implica que ninguna otra función puede acceder a los datos, lo que implica una simplificación para la escritura, además de una depuración y mantenimientos mucho más sencillas para cualquier programa. En resumen se puede decir que la POO tiene dos ventajas significativas sobre la programación estructurada: la primera es que permite al programador organizar su programa de acuerdo a la estructura del problema, esto implica que el desarrollador pueda crear un concepto del programa que se acerque más a la realidad y a la manera de pensar de la gente, dicho de otra forma, los objetos son las unidades de representación de las aplicaciones, por ejemplo, cuentas de banco, reservaciones de vuelo, etc.; y

segunda es que los *datos globales* desaparecen siendo estos junto con las funciones parte interna de los objetos, por tanto, cualquier modificación en la estructura de alguno de los datos sólo afectará a las funciones definidas dentro del mismo objeto y no en los demás. En la (Fig. 57) se puede observar la estructura general de la POO:

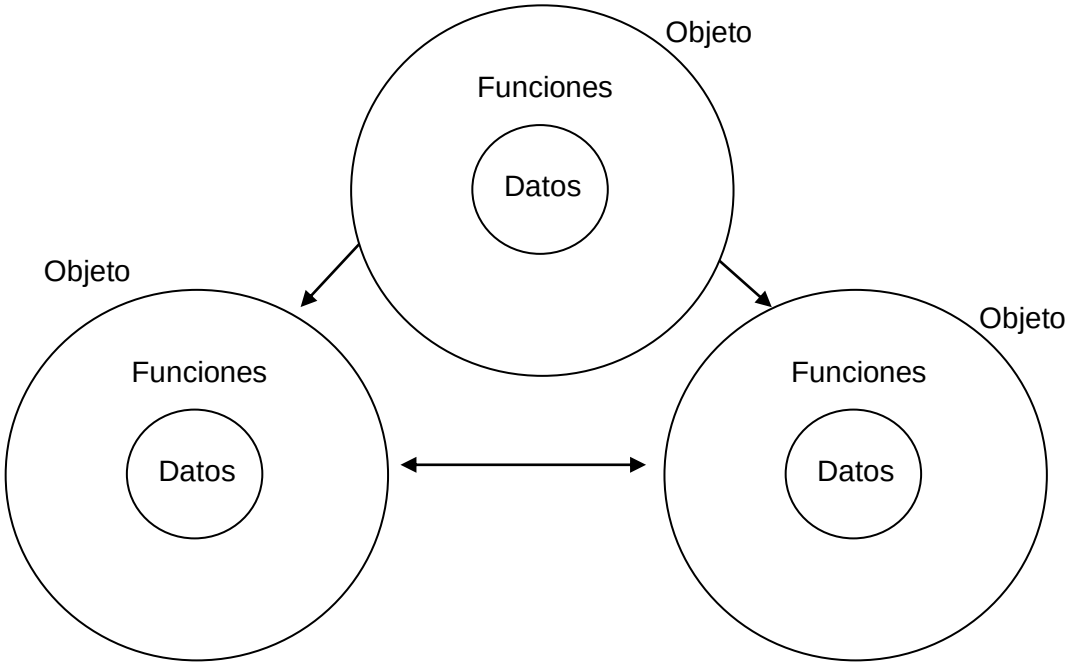


Fig. 57 Estructura de la POO

Para este trabajo se empleará el paradigma denominado *Programación Orientada a Objetos*. Por tanto en este tópico se desarrollarán los fundamentos y características de la POO, se desarrollará una pequeña introducción sobre el lenguaje *UML (Lenguaje Unificado de Modelado*). Se explicará qué es y para qué sirve el lenguaje de programación *Java*, así como descripción del IDE de programación *NetBeans* y para finalizar se dará una breve explicación sobre el uso de la tecnología *JSP (JavaServer Pages).* En el (Apéndice V) se encuentra una lista de los Lenguajes de Programación Orientados a Objetos más importantes existente hasta el momento.

4.5.1 Fundamentos de programación orientada a objetos

La Programación Orientada a Objetos (POO) está enfocada a resolver problemas de desarrollo de software, de tal manera, que el programador pueda diseñar un software

como si estuviera construyendo *objetos* de la vida real. Un *objeto* dentro de la POO puede ser cualquier cosa (un automóvil, un perro o hasta una persona), cada *objeto* según la POO está compuesto de los siguientes componentes: primero se tienen sus *atributos (características)*, los cuales definen el color, el tamaño, la fuerza, etc.; en segundo lugar la funcionalidad, por ejemplo, un perro puede ladrar, correr, morder, etc., a estas funciones dentro del paradigma de programación se les denomina *métodos;* para obtener la respuesta de un *objeto* debe existir un canal de comunicación por eso se encuentra en el tercer puesto a los *mensajes*, los cuales ayudan a indicar que *método* se desea ejecutar, por ejemplo, se tiene un *objeto* denominado *perro,* si se desea que el *objeto perro* ejecute el método *ladrar*, se le debe dar la indicación mediante el *mensaje ladra* el *objeto perro* llama a la *función (método)* debe ejecutar para realizar la acción .

Estas tres características básicas las tiene cualquier objeto (Eckel, 2007, págs. 1-3), pero ¿de qué sirve identificarlas?, la POO trata de encontrarlas para crear un *molde* (denominado *clase*) con el fin de obtener todos los objetos que se requieran de ese molde, por ejemplo, suponga que se desea construir una computadora, primero se debe seleccionar la *clase*, para este caso se tienen dos opciones: *laptop y computadora de escritorio*, se desean construir *objetos* de tipo laptop, todas las laptops están construidas, con un procesador, un disco duro, una memoria RAM, un botón para prender y apagar, estos son los *atributos*, además cada una tiene un programa que sirve para apagar y encender la computadora, esta función (*método*), ayuda a que se ejecute una acción por medio de un *mensaje* que se envía cuando se presiona el botón “encender/apagar”. Si se observa con cuidado se acaba de crear un molde, no importa que velocidad tenga cada procesador, ni el tamaño del disco duro, tampoco se requiere saber a qué compañía pertenece el programa que sirve para encender o apagar la computadora.

Por tanto se pueden crear tantas laptops como se desee, cada que se armar una laptop (*objeto*), simplemente se tienen que *asignar a los atributos*, por ejemplo, el tamaño de la memoria RAM, la velocidad de procesador, etc.

La ventaja de usar la POO es que podemos reutilizar código, no es necesario crear piezas cada vez que se desea obtener un *objeto*, pues con una *clase*, podemos obtener una *plantilla* la cual se puede modificar de acuerdo a las necesidades requeridas.

Por tanto los fundamentos para comprender a la POO son las siguientes (Barnes & Kolling, 2007, págs. 3-9):

- *Clase*: es la plantilla que permite la creación de un tipo de objeto.

- *Objeto*: se definen como cualquier cosa del mundo real (auto, casa, perro, etc).
- *Atributos*: son las características particulares del objeto (color, olor, tamaño, etc.).
- *Métodos*: definen el comportamiento de los objetos (caminar, correr, transportar).
- *Mensajes*: sirven para intercambiar información entre objetos.

4.5.2 Características de la programación orientada a objetos

Las características fundamentales de la POO son las siguientes (Ceballos, 2002, págs. 32-34):

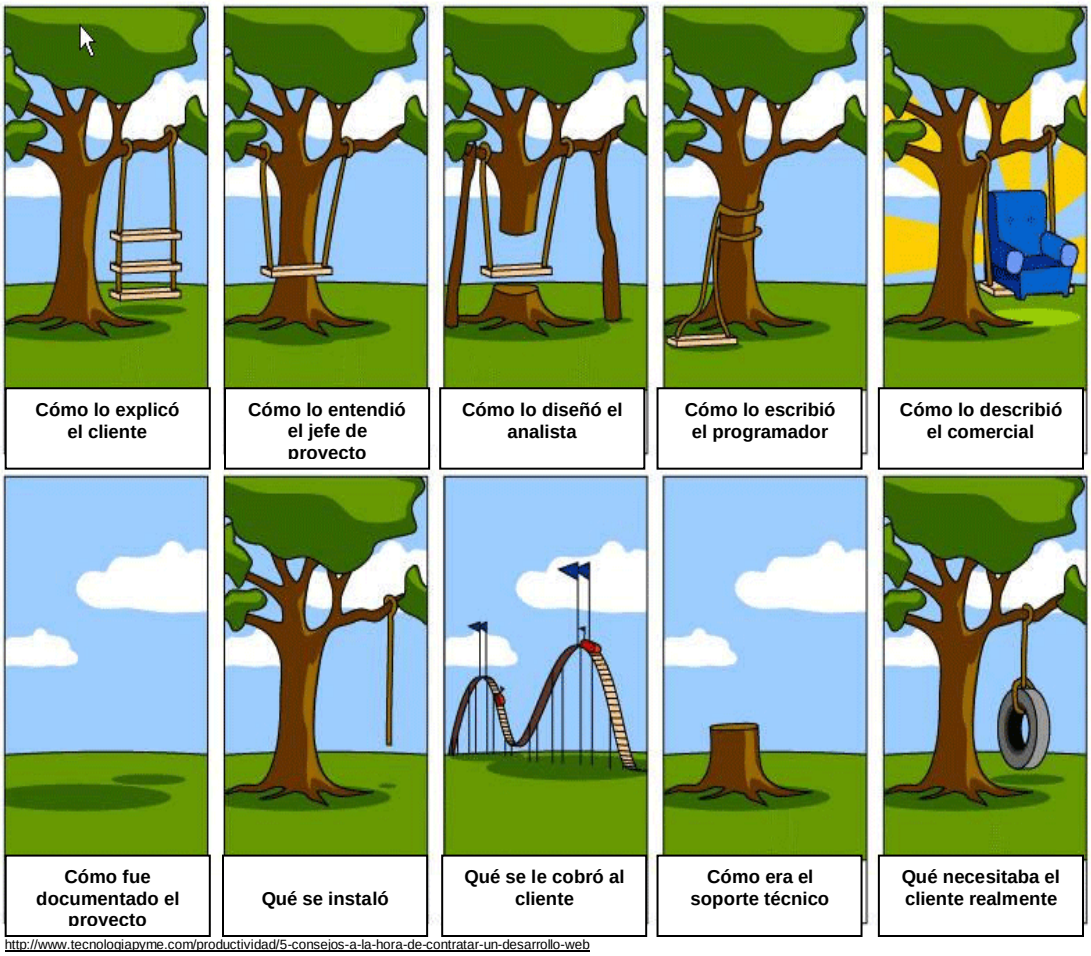
- *Abstracción*: permite identificar las características generales de un objeto, por ejemplo, una persona puede caminar, en este caso no es necesario enfocarse en ver si la persona es alta o enana o si es veloz o no, si no como se hace para que cualquier persona camine.
- *Encapsulamiento*: hace que el objeto se vea como una caja negra, en la que se encuentra toda la información necesaria para que un objeto funcione, esto permite que los objetos se vuelvan unidades básicas, pudiendo así ocultar su estructura interna**.**
- *Herencia*: permite el acceso automático de una clase a la información contenida en otras clases, permitiendo así la reutilización de código, una clase que hereda a otra es conocida como superclase, mientras que la que obtiene la herencia es denomina subclase, esto quiere decir que una clase puede utilizar métodos de otra clase.
- *Polimorfismo*: admite la utilización de un mismo método para ser implementado en múltiples formas, por ejemplo, si se tiene un método denominado identifica_persona, con este mismo método podemos buscar una persona en una base de datos pasando distintos *atributos*, se podría indicar en una búsquela por medio de la edad y el sexo, mientras que en otra se podría indicar el nombre de la persona que se desea encontrar.

Es imprescindible comprender como funciona el paradigma orientado a objetos, pues si se comprenden las bases no se tendrá ningún problema para manejar cualquiera de los distintos lenguajes de POO creados hasta el momento, la dificultad estará basada en la comprensión de la sintaxis de cada uno de ellos.

4.5.3 UML (Lenguaje Unificado de Modelado)

UML (Lenguaje Unificado de Modelado), es una herramienta que permite a todos los involucrados en el desarrollo de un sistema de software, comunicar de manera eficiente la idea de lo que se desea desarrollar. Uno de los principales problemas antes de la creación y utilización de una metodología sólida como lo es UML, radicaba en la inexistencia de un lenguaje común y sencillo de comprender para todos los involucrados en el desarrollo de una aplicación.

Esto es lo que sucedía con frecuencia: el administrador de proyectos tomaba los requerimientos que el cliente le explicaba, posteriormente el analista realizaba sus propios diagramas y análisis, los cuales eran entregados al programador, el cual los entendía de una forma específica, al realizar entregables, el cliente modificaba las especificaciones, pero seguían siendo interpretadas de otra forma, lo que generaba al final un producto engorroso y difícil de utilizar (Fig. 58).



http://www.tecnologiapyme.com/productividad/5-consejos-a-la-hora-de-contratar-un-desarrollo-web

Fig. 58 Deficiencias en el diseño y creación de software.

Este problema generaba grandes pérdidas de dinero. En los inicios de la computación los encargados de crear un software no se involucraban demasiado con las especificaciones del cliente, simplemente tomaban las ideas como se iban generando, ocasionando desastres de grandes magnitudes. Con el desarrollo tecnológico en forma exponencial, y la necesidad de crear procesos eficientes, se hizo indispensable contar con un plan de trabajo y con una definición de roles específica. Así como un arquitecto necesita realizar un anteproyecto para la creación de un edificio, también el software debía ser planeado y comprendido antes de su construcción.

UML surge a partir de la necesidad de planear, diseñar, desarrollar, ejecutar y controlar la construcción de software de calidad. Según (Rumbaugh, Booch, y Jacobson, 2000, págs. 4-8) en 1994 Rumbaugh, Booch, y Jacobson unieron sus esfuerzos dentro de Rational Software Corporation para la creación de una metodología orientada a objetos que fuera eficaz. En 1996 el Grupo de Administración de Objetos (OMG por sus siglas en inglés) solicito a estos tres personajes una propuesta para la estandarización del modelado orientado a objeto, lo que concluyó en 1997 con la primera versión de UML, desde ese momento la OMG se hizo responsable sobre el desarrollo de futuras mejoras en el estándar de UML.

El significado de *unificado* se da por que combina conceptos comúnmente aceptados por métodos orientados a objetos, UML puede representar la mayoría de los modelos existentes tan bien o mucho mejor que sus metodologías originales. Por otra parte este lenguaje elimina cualquier discontinuidad dentro del ciclo de vida de desarrollo, lo que permite el uso de procesos iterativos en el diseño e implementación de una aplicación. Está pensado para el desarrollo de sistemas simples y complejos, con el uso de múltiples lenguajes de programación, plataformas y bases de datos.

El *modelado* de un sistema permite captar y plasmar las ideas de tal forma que todos puedan comprenderlas

Por tanto UML pretende ser un lenguaje simple, que pueda ser entendió por la mayor parte del equipo de trabajo involucrado en el desarrollo de un sistema.

UML se basa en el uso de diagramas para el modelado de un sistema, enseguida se enlistan los diagramas que utiliza este lenguaje (Schmuller, 2000, págs. 28-37):

- *Diagrama de clases.*
- *Diagrama de objetos.*
- *Diagrama de casos de uso.*

- *Diagrama de estados.*
- *Diagrama de secuencias.*
- *Diagrama de actividades.*
- *Diagrama de colaboraciones.*
- *Diagrama de componentes.*
- *Diagrama de distribución.*

UML es uno de los lenguajes de modelado más utilizado para el diseño de sistemas. Esta es una de las metodologías más aceptadas y mejor estructuradas para el diseño y modelado de sistemas de software.

4.5.4 ¿Qué es java?

Es un lenguaje de programación de alto nivel, surge en 1991 como un proyecto de SunMicrosystems actualmente propiedad de Oracle (Oracle, 2009, s/p), destinado a usarse en el desarrollo de aplicaciones para electrodomésticos. Debido a la poca memoria con la que cuentan estos dispositivos, tenía que ser sencillo y práctico.

Uno de los problemas con los que se enfrento la compañía fue que existían muchos dispositivos en el mercado, cada uno con su propio hardware, esto genero el interés en desarrollar un lenguaje que pudiera ejecutar sus aplicaciones de manera independiente al hardware o al sistema operativo que se tuviera, por tanto, se ideo la creación de una *Máquina Virtual*, la cual interpretaba el código del lenguaje java y lo transformaba en un código que podía ser interpretado por la máquina en la que se estuviera ejecutando la aplicación. Esto permitía enunciar un lema que decía "*escribe el código, ejecuta donde sea*". Aunque hubo un gran esfuerzo por el desarrollo de este lenguaje ninguna empresa de electrodomésticos se intereso en esta tecnología.

Posteriormente en 1995 se libera la primera versión para computadoras manejando el mismo principio del uso de una *Máquina Virtual*. La clave fue incorporar un *intérprete Java* en la versión 2.0 del programa Netscape Nevagator, esto causo una verdadera revolución en la Internet, pues independientemente del dispositivo de hardware con que se contará no había necesidad de hacer cambios para poder visualizar una aplicación Java.

Con el paso de los años y con la evolución de este lenguaje de programación, se le han otorgado las siguientes características (Gosling y McGilton, 1996, s/p):

- *Simple:* el lenguaje de programación Java es muy sencillo de comprender, el programador puede comenzar a producir software rápidamente, elimina las partes complicadas que agrega C++, haciendo el desarrollo más eficiente.
- *Orientado a objetos*: cumple con el paradigma de Programación Orientado a Objetos, este lenguaje incluye, polimorfismo, encapsulamiento, herencia y abstracción.
- *Multiproceso y dinámico*: el uso de hilos le permite aprovechar todo el potencial del equipo de cómputo con el que se trabaje, permitiendo un alto rendimiento en la ejecución de subprocesos.
- *Con una arquitectura neutral y portátil*: el uso de una máquina virtual permite que el software desarrollado con este lenguaje pueda ser ejecutado independientemente del hardware y el software con el que se trabaje.
- *Robusto y seguro*: este lenguaje permite la creación de software altamente confiable, cuenta con guías para realizar buenas prácticas de programación. En las aplicaciones Java Web está protegido contra intrusiones de código malicioso en sus archivos. Esta tecnología está diseñada para trabajar en sistemas distribuidos.

  La gestión de memoria es muy simple, no es necesaria la utilización de punteros, eliminando así el desbordamiento de memoria, además cuenta con un recolector de basura automatizado.

Java es uno de los leguajes de programación más utilizados en esta época debido a las características antes mencionadas.

4.5.4.1 Máquina Virtual de Java

La Máquina Virtual de Java (JVM por sus siglas en inglés), es la aplicación que ejecuta un programa Java. Cuando se instala la JVM en una computadora esta puede ejecutar aplicaciones Java, los programas no son autosuficientes, lo que quiere decir que no necesitan todo el código máquina para ser ejecutados en una computadora. Lo que sucede en realidad es que los programas compilados son convertidos en *bytecodes* y esos *bytecodes* son los que la JVM interpreta y ejecuta. En otras palabras todo el código máquina que se requiere para ejecutar una aplicación Java en una computadora ya se

encuentra dentro del ordenador desde el momento en el que se instala la JVM, esto permite que los programas Java sean muy pequeños.

Según (Java Tutorials, 2010, s/p) la ejecución de un programa java se realiza de la siguiente manera:

- Se tiene un archivo de texto con una extensión *.java* el cual contiene el código del programa a ser ejecutado.
- El compilador *javac*, compila el código y obtiene un archivo *.class*, este documento contiene los bytecodes.
- La JVM lee el documento .class, lo interpreta y lo ejecuta en la máquina en la que se encuentre, esto permite que la máquina virtual se encargue de agregar el código máquina requerido para ejecutar la aplicación.

En la (Fig. 59) que se muestra a continuación, se puede visualizar el proceso de un programa desarrollado con Java.

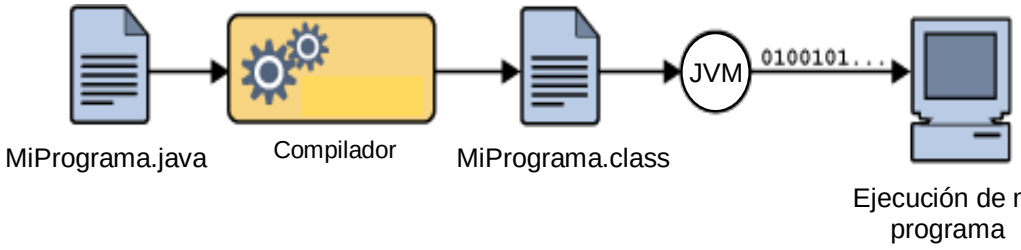


Fig. 59 Ejecución de un programa java

De esta manera cualquier aplicación java puede ejecutarse independientemente del hardware y del software con que se cuente, sólo se tiene que instalar la JVM para ese sistema en particular, por ejemplo (Fig. 60):

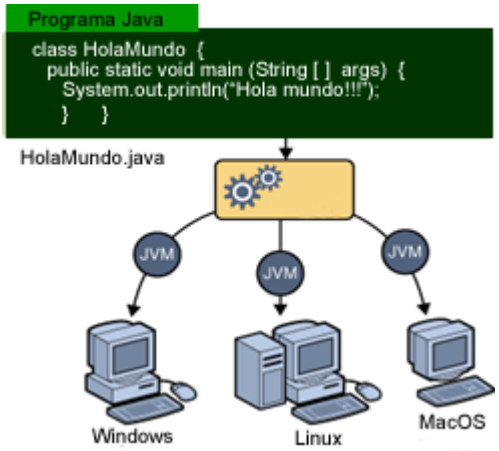


Fig. 60 Ejecución de un programa java

Otro complemento de java es su Interfaz de *Aplicación de Programación (API por sus siglas en inglés),* la *API* proporciona una gran cantidad de clases preexistentes (seguridad, componentes, interfaces para conexión a bases de datos), esto permite que el programador ahorre mucho tiempo a la hora de desarrollar una aplicación, pues simplemente crea los objetos de las clases que necesita y las aplica.

Por tanto se concluye que Java cuenta con una plataforma que está compuesta por la JVM y la API, esta plataforma es la base que permite la comunicación entre el código fuente y la computadora en la que se ejecuta una aplicación (Fig. 61):

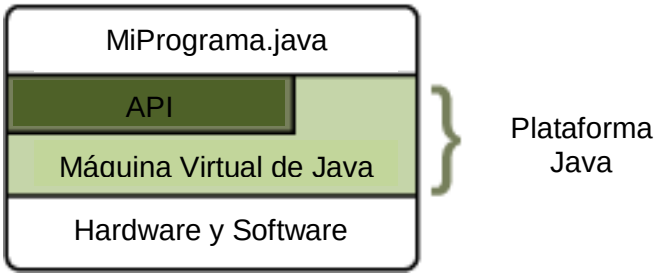


Fig. 61 Plataforma Java

Como ya se había mencionado SunMicrosystems fue comprada por Oracle, pero a pesar de esta compra por parte de Oracle el kit de desarrollo JDK de Java sigue siendo de uso público, existen tres versiones (SunMicrosystems, 2006, s/p):

- *Java SE (Standar Edition)*: esta es la versión básica de Java, contiene en su API las aplicaciones más utilizadas por los programadores, con esta versión se pueden trabajar con redes de datos, bases de datos, con interfaz gráfica y análisis XML.
- *Java EE (Enterprise Edition)*: la versión empresarial permite el desarrollo de aplicaciones más robustas, se utiliza para la creación de software a gran escala, es estable y confiable. La API de esta aplicación está dividida en tres niveles: de *aplicación*, este nivel permite crear aplicaciones para usuarios finales, donde el software se utilizara en una estación de trabajo*;* *nivel cliente*, permite un desarrollo en donde la aplicación desarrollada puede ser utilizada desde un servidor para un cierto número de personas; y por último está el *nivel web*, con el se pueden desarrollar aplicaciones que serán utilizadas en la red de redes.
- *Java ME (Micro Edition)*: esta edición permite el desarrollo de aplicaciones para móviles.

Como se observa Java es un lenguaje de programación que permite desarrollar aplicaciones seguras, confiables y robustas, de manera sencilla y eficiente, poniéndolo así como uno de los principales lenguajes de programación de la época.

4.5.4.2 NetBeans

Es un *Entorno de Desarrollo Integrado* (IDE por sus siglas en inglés) para Java de código abierto, esto quiere decir que puede ser utilizado sin restricciones, de manera gratuita y además su código puede ser modificado a conveniencia (NetBeans, 2010, s/p).
El proyecto para la creación de NetBeans surgió en junio de 2000 por parte de la empresa SunMicrosystems, actualmente su principal patrocinador es Oracle.
Este IDE es una herramienta que permite a un programador, escribir, depurar, compilar y ejecutar programas. Está escrito en java, pero esto no quiere decir que sólo se pueda utilizar para programar aplicaciones java, también es útil para otros lenguajes de programación, por ejemplo, php.
Por otra parte existe un número significativo de módulos que sirven para extender esta aplicación.
Cabe mencionar que no es el único entorno de desarrollo libre para java, existen también, Eclipse, JCreator Java, entre otros.

4.5.4.3 JSP (Java Server Pages)

Java Server Pages (JSP) es una tecnología de la plataforma Java 2, se encuentra dentro de la edición *Entreprise Edition (J2EE),* sirve para construir contenido Web dinámico, trabaja en conjunto con HTML, DHTML, XHTML y XML. Según (SDN, 2010,s/p) JSP permite desarrollar y diseñar contenido Web de forma rápida y flexible, permite un fácil mantenimiento de las aplicaciones. Las aplicaciones son ejecutadas desde el servidor lo que le permite ser una herramienta multiplataforma.
Esta tecnología se basa en los siguientes conceptos:

- *Uso de plantillas*: la mayor parte del contenido dinámico está basado en plantillas. Los archivos XML son los más utilizados como plantillas, la tecnología JSP tiene un soporte natural para este tipo de plantillas.

- *Adición de datos dinámicos*: los JSP permiten el uso simple y poderoso de datos dinámicos en las plantillas.
- *Encapsulamiento y funcionalidad*: esta tecnología proporciona el encapsulamiento y la funcionalidad por medio de dos mecanismos: los JavaBeans, que conforman la arquitectura y estructura del software, y el desarrollo de librerías, las cuales permiten la implantación de acciones, funciones, validaciones y clases.

Beneficios del uso de esta tecnología (Roth & Pelegrí-Llopart, 2003, s/p):

- Por ser una tecnología Java cumple con la condición "*escribe una vez, ejecuta donde sea*", esto quiere decir que las aplicaciones web que se desarrollen pueden ejecutarse en cualquier plataforma y sobre cualquier servidor.
- Una de las metas de esta tecnología es la creación de sitios web de *alta calidad y portabilidad.*
- *Reutilización de componentes y librerías,* una de las principales características de Java es la reutilización de código lo que permite reducir el número de líneas de programación.
- *Separación de contenido dinámico y estático*, la creación de contenido estático permite la creación de plantillas a las cuales les son insertadas las funciones del contenido dinámico, permitiendo la reutilización de código y un manejo más claro de datos.
- *Soporte para scripting,* JSP soporta el uso de scripts para la realización de acciones, las acciones son encapsuladas y pueden ser utilizadas de manera eficiente en todo momento. Otro elemento que se utiliza frecuentemente es el uso de expresiones para el acceso a datos, esta tecnología incluye un lenguaje de expresiones simples (EL, por sus siglas en inglés), este lenguaje está definido a través de librerías y permite un control de acceso mayor a las aplicaciones web.
- *Separación de roles*, se pude trabajar en un diseño general el cual para ser programado puede ser dividido en módulos lo que permite que se trabajen varias secciones a la vez, reduciendo así el tiempo de desarrollo y puesta en marcha de una aplicación.

Observando estas características se concluye que el uso de esta tecnología permite trabajar de manera efectiva con el uso de páginas web dinámicas, permitiendo un mantenimiento eficaz, debido a la separación de módulos y de código.

4.5.4.4 Applets Java

Un Applet es un programa escrito en el lenguaje de programación Java, este programa es compilado para su ejecución desde una página web. Cuando el navegador web carga una página que contiene un Applet, éste se descarga en el navegador y se ejecuta la Máquina Virtual de Java (JVM) (ver punto *4.5.4.1*) para ser interpretado y ejecutado, por tanto el Applet es ejecutado del el lado del cliente (Applet, 2011, s/p).

Ventajas:

- Son multiplataforma (funcionan en cualquier sistema operativo que tenga instalada la JVM).
- Es soportado por la mayoría de los navegadores Web.
- Puede ser almacenado en la memoria cache de la mayoría de los navegadores Web, de modo que se cargará rápidamente cuando se vuelva a cargar la página Web.
- Puede tener acceso completo a la máquina en la que se está ejecutando, si el usuario lo permite.
- Puede trasladar el trabajo del servidor al cliente, haciendo una solución web más escalable tomando en cuenta el número de usuarios / clientes.

Desventajas:

- Se requiere del plug-in de Java, que no está disponible por defecto en todos los navegadores web.
- No puede iniciar la ejecución hasta que la JVM esté en funcionamiento, y esto puede tomar tiempo la primera vez que se ejecuta un Applet.

- Si no está firmado como confiable, tiene un acceso limitado al sistema del usuario, en particular no tiene acceso directo al disco duro del cliente o al portapapeles.
- Algunas organizaciones sólo permiten la instalación de software a los administradores. Como resultado, muchos usuarios (sin privilegios para instalar el plug-in en su navegador) no pueden ver los Applets.
- Un Applet podría exigir una versión específica del JRE.

4.5.4.5 JavaBeans en JSP´s

Los JavaBeans (JB´s) surgen de la necesidad de contar con componentes reutilizables e independientes de la plataforma. Cuando un programador crea un JB no hace otra cosa que definir una clase, para en capsular la implementación y mostrar al usuario del JB los métodos y propiedades que son públicos.

Este concepto es un elemento esencial para un trabajo eficiente en equipo, colaborar sin que cada ingeniero tenga que repetir lo que ya hizo otro, sin que tenga que enfrentarse a las mismas dificultades que otro ya ha resuelto. El programador que usa el JB sólo debe preocuparse por ¿qué hace? el JB.

Existen JavaBeans gráficos (JBGráficos, 2011, s/p) de capa cliente (controladores de interfaz) y JavaBeans de capa Web (JB, 2011, s/p), en el presente trabajo se utilizarán los segundos, cundo se mencione a un JavaBeans (JB) se estará haciendo referencia a los JB de capa Web.

Para que un JB sea considerado como tal debe seguir ciertas propiedades:

- Se deben declarar métodos de sólo lectura y sólo escritura los cuales deben estar definidos de la siguiente manera:

    - Los métodos de sólo escritura sirven para insertar valores a los atributos del JB y deben tener la siguiente estructura:

```
public [tipo de variable que retorna]  set[Nombre del método] (  )
{
        cuerpo del método
}
```

- Lo métodos de sólo lectura sirven para obtener los valores guardados en un JB y deben tener la siguiente estructura:

  public void get[Nombre del método] ([tipo de variable que recibe] [nombre de la variable que recibe])
  {
  cuerpo del método
  }

- Hay una excepción a lo comentado anteriormente, si la propiedad es booleana, entonces para obtener su valor se usa el método:

  public [tipo de variable que retorna] is[Nombre del método] ( )
  {
  cuerpo del método
  }

Ejemplo de un JavaBean:

```
public class persona {

//Se declaran los atributos de tipo privado

        private String nombre;
        private int edad;
        private boolean estadoCivil; //Casado es igual a “True”, soltero “False”

//Constructor vacio
 public persona()
{

 }

//Constructor vacio que sirve para inicializar las variables
 public persona( String nombre, int edad ) {
     this.nombre = nombre;
     this.edad = edad;
   }
```

```
//Métodos de solo lectura

   public String getNombre()
{
            return nombre;
}

  public int getEdad()
 {
              return edad;
}

public boolean isEstadoCivil()
{
            return estadoCivil;
}


//Métodos de solo escritura

   public void setNombre( String nombre )
 {
           this.nombre = nombre;
}

   public void setEdad( int edad )
{

      this.edad = edad;
 }


public void setEstadoCivil(boolean estadoCivil )
{
           this.estadoCivil = estadoCivil;
}


}//Fin de la clase
```

Los JB serán utilizados en este trabajo para resolvemos los siguientes problemas:

- La necesidad de manejar múltiples variables que tendrían que ser generadas en los JSP´s.
- La pérdida de información al pasar de una página a otra.
- Se elimina la necesidad de hacer múltiples peticiones a la base de datos