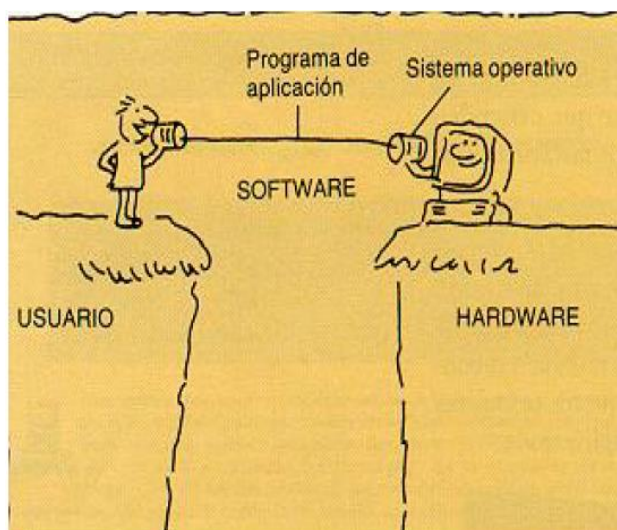


Clase Nº 3

Software

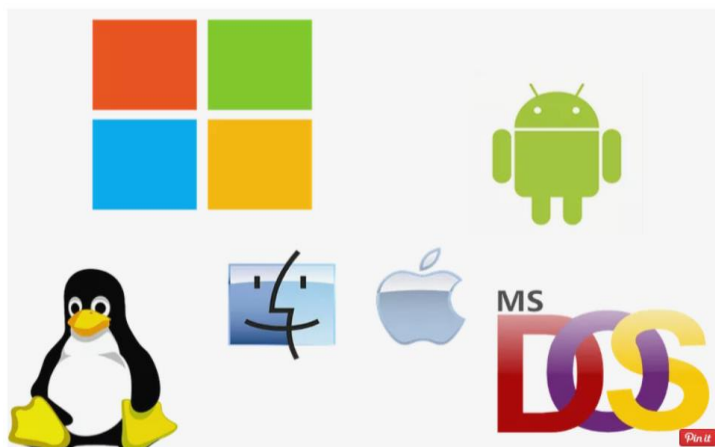
El Software es el conjunto de los programas de cómputo, procedimientos, reglas, documentación y datos asociados que forman parte de las operaciones de un sistema de computación. Es simplemente un conjunto de instrucciones individuales que se le proporciona al microprocesador para que pueda procesar los datos y generar los resultados esperados. Sin el software, la computadora sería un conjunto de dispositivos sin utilizar.



Clasificación de Software, de acuerdo a las tareas que realizan:

Software de Sistemas: Controlan y optimizan las operaciones de la máquina. Su objetivo es desvincular adecuadamente al usuario y al programador de los detalles del computador en particular que se use, aislándolo especialmente del procesamiento referido a las características internas de: memoria, discos, puertos y dispositivos de comunicaciones, impresoras, pantallas, teclados, etc. El software de sistema le procura al usuario y programador adecuadas interfaces de alto nivel, herramientas y utilidades de apoyo que permiten su mantenimiento. Incluye entre otros:

- Sistema Operativo
- Controladores de dispositivo
- Herramientas de diagnóstico
- Herramientas de Corrección y Optimización
- Utilidades



Sistemas Operativos: es el soporte lógico principal que controla el funcionamiento del equipo físico, ocultando todos los detalles del hardware y permitiendo así que el uso de la computadora sea más sencillo para el usuario.

El sistema operativo es el gestor y organizador de todas las actividades que realiza la computadora. Marca las pautas según las cuales se intercambia información entre la memoria central y la externa, y determina las operaciones elementales que puede realizar el procesador. El sistema operativo, debe ser cargado en la memoria central antes que ninguna otra información.

- El funcionamiento del Sistema Operativo implica de la existencia de al menos un programa que está permanentemente ejecutándose junto con nuestras aplicaciones.
- Esto significa que un subconjunto de los recursos de las computadoras son “tomados” por el Sistema Operativo que se comunica directa o indirectamente con los programas de aplicación que se ejecutan para cooperar en la ejecución o retomar el control del hardware en caso de problemas.

Ejemplos de qué hace un Sistema Operativo:

- **Comunicación con los periféricos**

Una de las tareas más complejas realizadas por una computadora es la comunicación con pantallas, scanners, impresoras, unidades de disco, mouses, teclados, placas de sonido, placas conversoras analógico/digitales y otros dispositivos periféricos. El sistema operativo incluye programas que se encargan, de un modo transparente al usuario, de los detalles de comunicación con los periféricos o con el hardware que controla estos periféricos.

- **Control de autorización de usuarios**

En el caso de las computadoras monousuario el sistema operativo puede verificar (mediante una clave o password por ejemplo) que el usuario que trata de utilizar el equipo está habilitado para ello. Más aún puede tener derechos sobre determinados recursos del equipo pero no sobre todos los recursos.

Cuando se trata de computadoras multiusuario, o en el caso de redes de computadoras, la tarea de administración de usuarios del sistema operativo es bastante más compleja,

porque los derechos de cada usuario pueden ser diferentes sobre cada máquina, cada base de datos o cada periférico.

- **Control de la ejecución de programas**

La ejecución efectiva de un programa (escrito en cualquier lenguaje de programación o aplicación) requiere una comunicación permanente con el sistema operativo para acceder a los recursos de la computadora, recursos que el sistema operativo controla y verifica. De este modo se puede detectar que una orden de impresión escrita en un programa es imposible de ejecutar porque la impresora no está encendida, o que un dato de un archivo no se puede recuperar porque falla el dispositivo periférico, o que la ejecución de un programa ha tardado más de un tiempo máximo determinado, etc.

También el sistema operativo monitorea el resultado de la ejecución para transmitir al usuario el mensaje adecuado resultante de la evolución de la ejecución.

- **Control de concurrencia**

Las computadoras multiusuario (que tienen terminales conectadas a un gran procesador central), o las redes de computadoras, o las modernas computadoras paralelas con varios procesadores internos, pueden tener varios trabajos ejecutándose al mismo tiempo (procesamiento concurrente). Esto exige que el sistema operativo controle que hace cada proceso y permita que los mismos compartan datos y recursos (es decir se comuniquen y se sincronicen).

Por otra parte estos múltiples procesos pueden tener diferente prioridad para acceder a los recursos, lo que debe ser controlado también por el sistema operativo.

- **Control de errores**

Como se mencionó anteriormente, cada error de ejecución de una aplicación termina entregando el control al sistema operativo que debe manejar la solución al error (desde el punto de vista que el sistema de cómputo siga funcionando) y también la comunicación clara al usuario de las causas del error.

- **Administración de memoria**

Al poder procesar concurrentemente varios trabajos, el sistema operativo debe controlar la forma de usar la memoria de la computadora, de modo que un trabajo no invada el espacio físico de otro. Los esquemas de administración de memoria pueden ser muy sencillos (división en partes asignadas a cada proceso) o más sofisticada de modo de asignar y liberar memoria en forma dinámica según los requerimientos y prioridades de los procesos.

- **Controles de seguridad de datos**

Los datos almacenados en una computadora pueden tener protecciones (imaginen una base de datos con la información de cada alumno de la Facultad, incluyendo las notas de sus exámenes) de modo de autorizar las modificaciones, agregados o consultas. Estas funciones de seguridad también forman parte del sistema operativo.

Los componentes del Sistema Operativo son

- Kernel
- Interfaz de Usuario
- Sistema de Archivos

El Kernel: es el nombre más común para el núcleo del sistema operativo. El kernel es un conjunto de código relativamente pequeño que es cargado en memoria cuando se inicia la computadora. Este código de computación contiene instrucciones que permiten que el kernel administre los dispositivos de hardware, como los discos. El kernel también administra y controla la asignación de memoria, los procesos del sistema, y a otros programas. El software de aplicación y otras partes del sistema operativo dependen del kernel para proporcionar servicios básicos de organización y acceso al hardware y periféricos de la computadora.

Cuando se utiliza un sistema operativo UNIX o Linux, puede estar presente un archivo llamado "kernel". En algunos casos puede ser necesario personalizar y compilar el código del kernel. Si este archivo se corrompe, el sistema dejaría de funcionar.

En un sistema Windows, se pueden ver nombres de archivo que incluyen la palabra "kernel" o "kern", como "kernel32.dll". Estos son archivos críticos usados por el núcleo del sistema operativo.

La Interfaz de Usuario: La Interfaz de Usuario (IU), es la parte más visible de un sistema operativo de computadora. La IU es el componente del SO con el que interactúa el usuario, actuando como un puente entre el usuario y el kernel. La UI es como un intérprete, traduce golpes de teclas, clicks del mouse, u otras entradas del usuario para los programas apropiados. La salida del programa puede ser organizada y mostrada por la IU. En un sistema UNIX o Linux, a la IU se la llama comúnmente shell.

Las interfaces de usuario se dividen en dos categorías generales:

- Interfaz de Línea de Comando (Command-Line Interface, CLI)
- Interfaz Gráfica de Usuario (Graphical User Interface, GUI)

Los primeros sistemas operativos de PC de escritorio usaban exclusivamente CLIs. La CLI proporciona al usuario un prompt visual, y el usuario ingresa comandos escribiéndolos. La computadora devuelve datos a la pantalla en forma tipográfica. En otras palabras, un entorno de CLI está completamente basado en texto y el usuario sólo puede operar ingresando comandos con el teclado.

Hoy, todos los SOs de escritorio populares soportan GUIs. Una GUI permite que el usuario manipule el software utilizando objetos visuales como ventanas, menús desplegables, punteros, e iconos. La GUI permite que el usuario ingrese comandos por medio de un mouse u otro dispositivo de señalización.

Los usuarios finales prefieren una interfaz gráfica porque facilita y hace más intuitivo el uso de la computadora. Un usuario puede ejecutar operaciones sencillas usando una GUI sin siquiera saber leer.

La desventaja de simplificar la interfaz del usuario aparece en la performance. Algunos software de GUI pueden consumir más de cien veces el espacio de almacenamiento que el software de CLI. Y como las GUIs son más complicadas que las CLIs, el software de GUI requiere mucha más memoria y tiempo de CPU.

El Sistema de Archivos: El sistema de archivos de un SO determina la forma en que los archivos son nombrados y cómo y dónde son colocados en los dispositivos de

almacenamiento, como discos rígidos. Los SOs Windows, UNIX y Linux emplean todos sistemas de archivos que utilizan una estructura jerárquica.

En un sistema de archivos jerárquico, los archivos son colocados en contenedores lógicos que son organizados en una estructura de árbol invertida. El sistema de archivos comienza en la raíz del árbol.

UNIX y Linux llaman "directorio" a un contenedor residente en el nivel superior del árbol. Los contenedores dentro de cada directorio son llamados "subdirectorios". Los SOs Windows utiliza el término "carpeta" y "subcarpeta" para describir a los directorios y subdirectorios.

El sistema de archivos de un SO determina más que sólo la forma lógica en que los archivos y carpetas están organizados. El tipo de sistema de archivos utilizado por la computadora determina si los archivos pueden estar protegidos o no de otros usuarios o programas. El sistema de archivos también define la forma en que los datos están organizados físicamente en el medio de almacenamiento (como un disco rígido). Algunos sistemas de archivos utilizan el espacio de disco más eficientemente que otros.

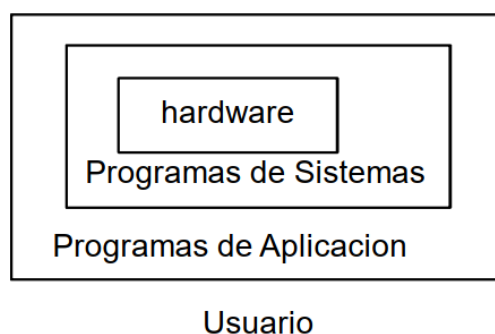
Diferentes sistemas operativos utilizan diferentes sistemas de archivos, y algunos sistemas operativos pueden usar más de un sistema de archivos.

El sistema de archivos determina las convenciones de nombres y el formato para especificar una ruta, o camino, hacia la ubicación del archivo. Estas reglas para poner nombres a los archivos varían dependiendo del sistema de archivos e incluyen temas como los siguientes:

- Cantidad máxima de caracteres permitidos en un nombre de archivo
- Longitud máxima de las extensiones de archivo o sufijos
- Si se permiten espacios entre palabras en un nombre de archivo
- Si los nombres de archivo son sensibles a las mayúsculas
- Cuáles caracteres son "legales" para utilizarlos en los nombres de archivo
- El formato para especificar la ruta

Software de Aplicación: El software de aplicación específica está diseñado y escrito para realizar tareas específicas personales, empresariales o científicas. Contienen instrucciones precisas para comunicar a la máquina el conjunto de operaciones que debe realizar sobre determinada información, de acuerdo a los requerimientos de los usuarios.

Los programadores de aplicaciones no necesitan conocer a fondo el funcionamiento del hardware, ya que sus programas van a ser independientes de los detalles de éste y podrán utilizarse en distintas computadoras.



Los Programas de Sistemas y de Aplicación, están relacionados entre sí, de modo que los usuarios y los programadores pueden hacer así un uso eficiente de la computadora. La figura muestra una vista organizacional de una computadora donde se ven los diferentes tipos de software a modo de capas de la computadora desde su interior (el hardware) hasta su exterior (usuario).

Los Programas de Aplicación se dividen en:

- **Aplicaciones Verticales:** Tienen una función específica para determinados usuarios.



- **Aplicaciones Horizontales:** Útiles para distintos tipos de usuarios. Son los más estándares y utilizados.



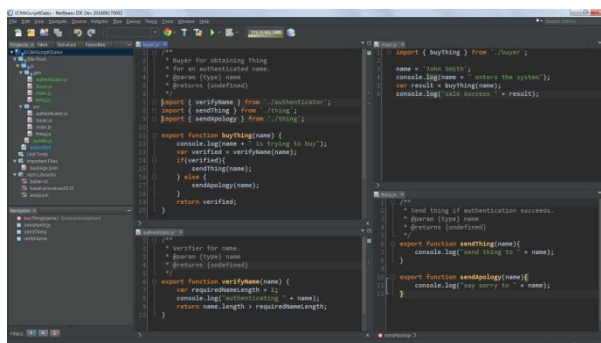
- **Software de Programación:** Es el conjunto de herramientas que permiten crear/desarrollar/programar otras aplicaciones, usando diferentes alternativas y lenguajes de programación, de una manera práctica.

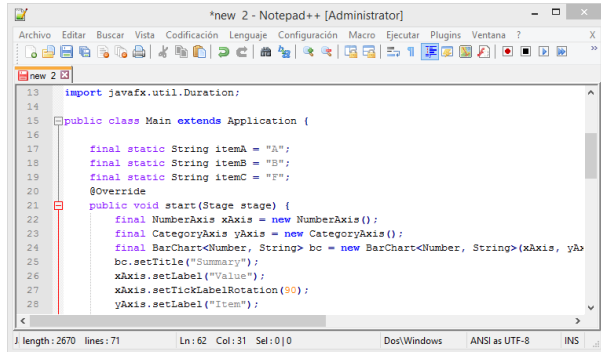
Los software de programación son los que dan origen a los programas que utilizamos día a día. Es en ellos donde se emplean los lenguajes de programación, los cuales sirven para crear las instrucciones que luego la computadora realizará.

Si pensamos por un momento, seguramente vamos a darnos cuenta de que el software de programación a primera vista es una especie de paradoja, ya que son programas para crear programas, que a su vez tuvieron que haber sido creados por otros programas.

Tipos de software de programación:

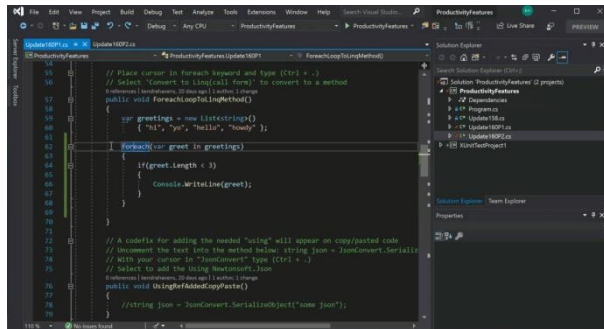
- **Editores de texto:** Programas cuyo propósito es procesar y almacenar texto, simple y llano. No debe confundirse con programas como Word, que brindan muchas más posibilidades y funciones. La tarea del editor de texto es que podamos usarlo para escribir y para ver el contenido de archivos de texto plano.
- **Compiladores:** Es una herramienta cuya función radica en realizar una traducción del código de un software, de forma que el mismo pueda ser correctamente interpretado por una computadora para lograr una ejecución exitosa. El compilador básicamente tiene la tarea de convertir un lenguaje de alto nivel en un lenguaje de bajo nivel que el hardware pueda interpretar.
- **Intérpretes:** Se trata del programa que nos permite realizar un análisis y/o una ejecución de un código escrito en un lenguaje de programación de alto nivel. Comparados con los compiladores, los intérpretes son más complejos y por lo general trabajan de forma más lenta, aunque también tienen una mayor flexibilidad.
- **Enlazadores:** A veces llamados *linkers*, la función de estos es de crear enlaces entre diferentes objetos obtenidos de la primera fase de la compilación, para posteriormente unirlos en un solo archivo o fichero que puede ser ejecutado.
- **Depuradores:** Conocidos a menudo con *debuggers*, son utilidades que permiten al desarrollador realizar pruebas con el código de sus software, para así poder detectar y eliminar errores en el mismo. La mayoría brindan la posibilidad de interpretar un código paso a paso, lo cual hace más sencillo detectar fallos en el software.
- **Entornos de Desarrollo Integrados (IDE):** Agrupan las anteriores herramientas, usualmente en un entorno visual, de forma que el programador no necesite introducir múltiples comandos para compilar, interpretar, depurar, etc. Habitualmente cuentan con una avanzada interfaz gráfica de usuario (GUI).



```

13 import javafx.util.Duration;
14
15 public class Main extends Application {
16
17     final static String itemA = "A";
18     final static String itemB = "B";
19     final static String itemC = "C";
20     @Override
21     public void start(Stage stage) {
22         final NumberAxis xAxis = new NumberAxis();
23         final CategoryAxis yAxis = new CategoryAxis();
24         final BarChart<Number, String> bc = new BarChart<Number, String>(xAxis, yAxis);
25         bc.setTitle("Summary");
26         xAxis.setLabel("Value");
27         xAxis.setTickLabelRotation(90);
28         yAxis.setLabel("Item");
    
```

```

54 // Place cursor in foreach keyword and type (Ctrl + .)
55 // Select 'Convert to LINQ (call form)' to convert to a method
56 // Follow the instructions. It will add a using directive
57 public void ForeachLoopToLinqMethod()
58 {
59     var greetings = new List<string>()
60     {
61         "hi", "yo", "hello", "howdy"
62     };
63     foreach (var greet in greetings)
64     {
65         if (greet.Length < 3)
66         {
67             Console.WriteLine(greet);
68         }
69     }
70
71 // A tooltip for adding the needed 'using' will appear on copy/pasted code
72 // Uncomment the line into the method below. string json = JsonConvert.SerializeObject(
73 // With your cursor in 'JsonConvert' type (Ctrl + .)
74 // Select to add the using Newtonsoft.Json
75 // Public void HigherForeachCopyPaste()
76 {
77     //string json = JsonConvert.SerializeObject("some json");
78 }
79
80
    
```