TÉCNICAS DE PROGRAMACIÓN

Ingeniería de Ejecución en Computación e Informática

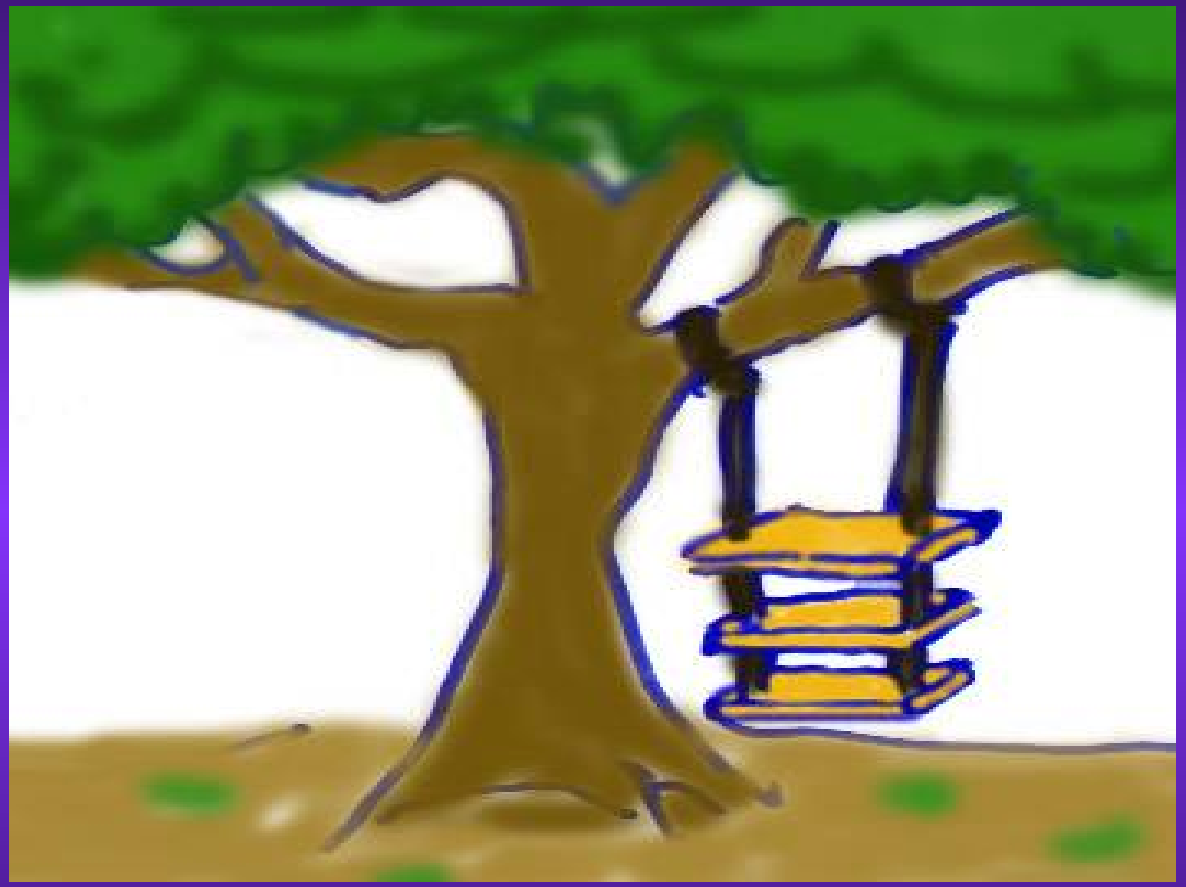
Todo elemento desarrollado por el hombre se concibe primero como una idea en su mente y nace como respuesta a requerimientos detectados. Antes que un programa se construya, debe existir una clara definición del problema que se pretende solucionar y del entorno en el que se sitúa. Debemos considerar que desde el punto de vista del usuario el problema se reduce a generar un listado o a capturar cierta información y no en algoritmos de clasificación o acceso a base de datos, interfaces, etc.

A continuación se presenta el “drama en 6 actos” que refleja el problema que comúnmente se produce.

DRAMA EN 6 ACTOS

Desarrollo de un Sistema

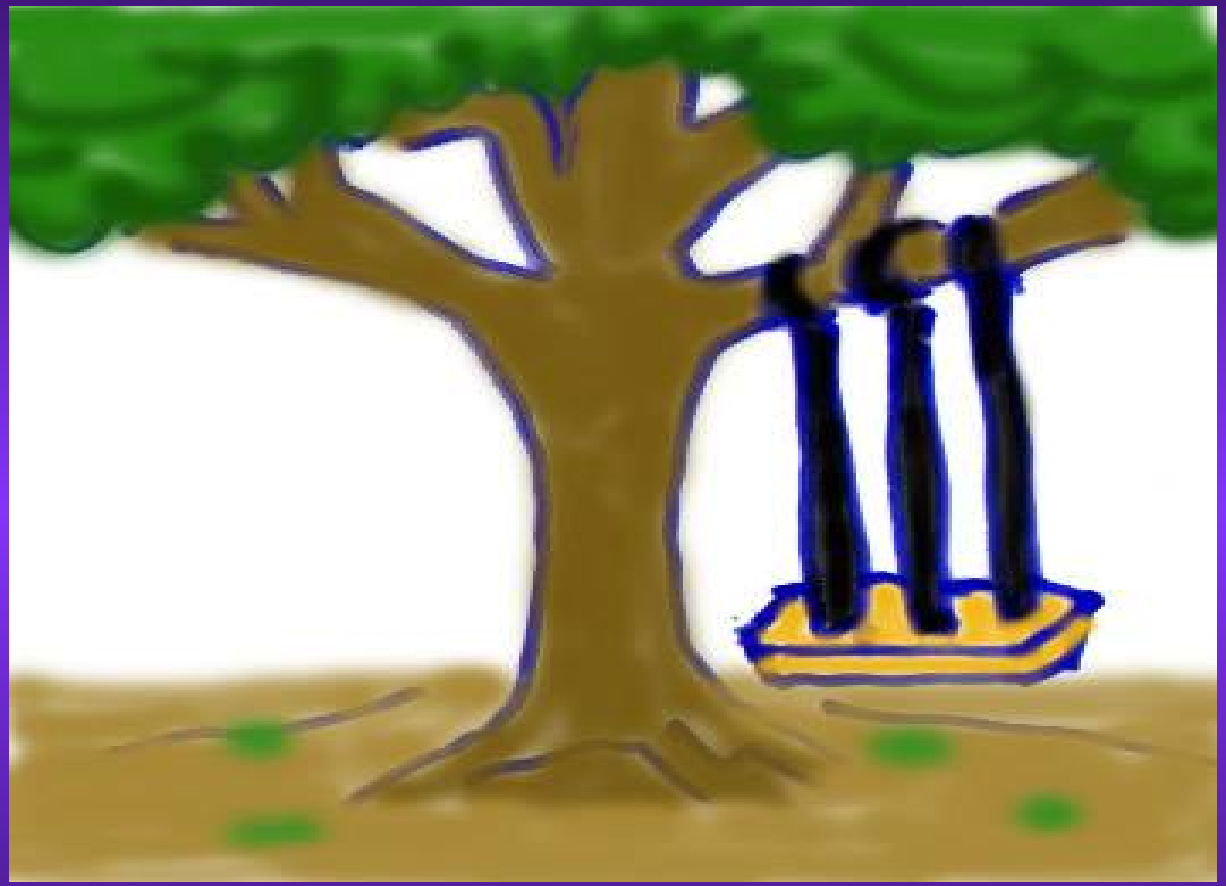
Acto 1:



PROPUESTA DEL INICIADOR DEL PROYECTO

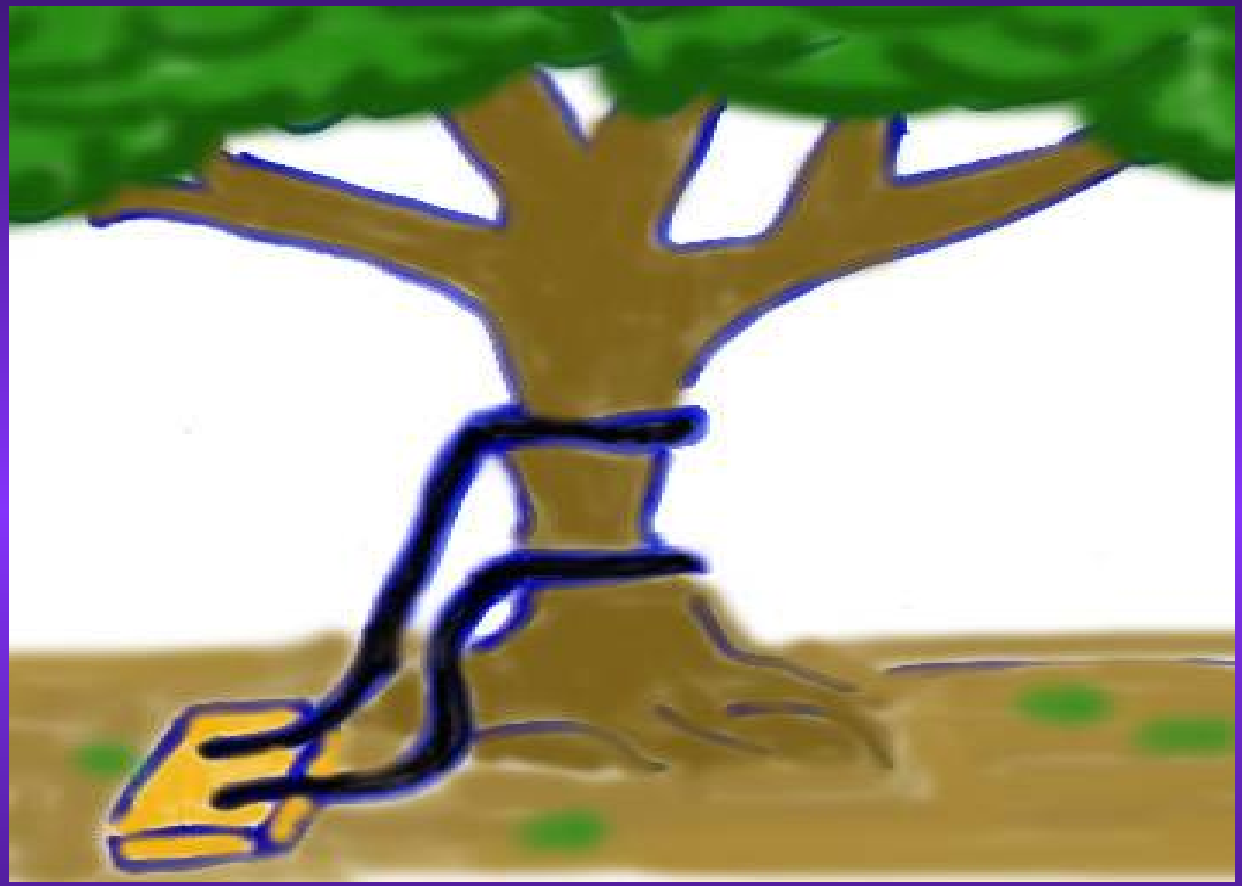
www.profmatiasgarcia.com.ar

Acto 2:



ASÍ SE ESPECIFICÓ EL PROYECTO

Acto 3:



EL DISEÑO DEL ANALISTA

Acto 4:

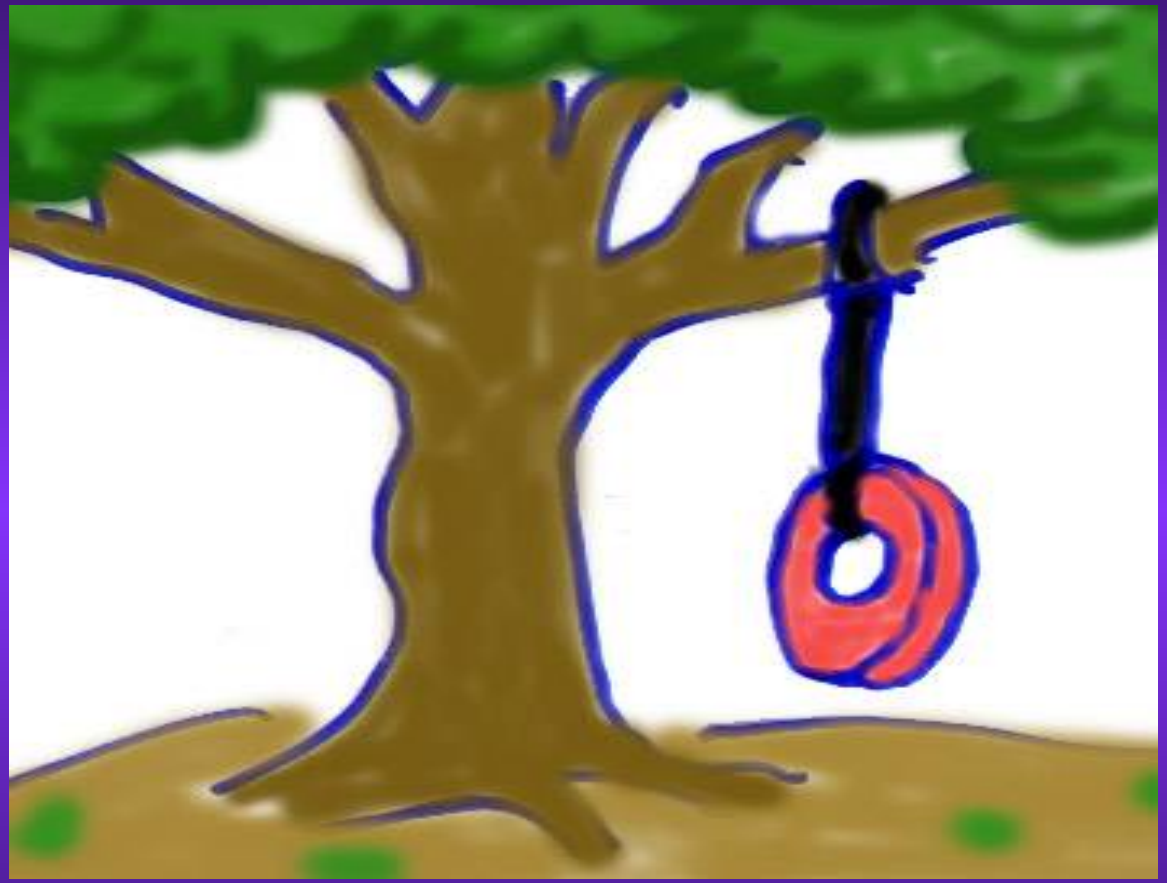
LA IMPLEMENTACIÓN DE LOS PROGRAMADORES

www.profmatiasgarcia.com.ar

Acto 5:

COMO SE INSTALÓ AL USUARIO

Acto 6:



LO QUE EL USUARIO NECESITABA

LOS PROGRAMAS

Un programa informático es un conjunto de instrucciones que una vez ejecutadas realizarán una o varias tareas dictadas por el programador en una computadora.

Ese conjunto de instrucciones fueron escritas en algún lenguaje de programación.

El programa debe ser compilado o interpretado para poder ser ejecutado y así cumplir su objetivo.

Al especificar un programa debemos considerar 3 cosas:

ENTRADA

- ¿Cómo están organizados y qué formato tienen los datos que el programa va a procesar?
- ¿Van a ser creados por el programa?
- ¿Se capturan mediante un terminal?
- ¿Se leerán desde un dispositivo de almacenamiento?
- ¿El dato será en formato de flujo continuo de caracteres?
- ¿Va a ser ingresados ítem a ítem por un operador en una pantalla?
- ¿Será leído registro a registro desde un archivo?
- ¿Cómo se accesan los archivos?
- ¿El dato será numérico o alfanumérico, ¿tendrá empaquetado?

PROCESAMIENTO

- ¿Cuál es el algoritmo requerido para transformar los datos de entrada en la salida deseada?
- ¿Qué cálculos son necesarios?
- ¿Es necesario ordenar los datos de entrada?
- ¿Son necesarias instancias de validación para asegurar la consistencia de la información?
- ¿Se debe considerar cortes de control?
- ¿Los datos mantendrán un formato entre la entrada y la salida?
- ¿Serán necesarios depósitos temporales de datos tales como tablas o archivos de paso?
- ¿Cuáles son las condiciones de borde?

SALIDA

- ¿Cuál es la estructura , contenido y formato de la salida deseada?
- ¿La información de salida se imprimirá, desplegará en pantalla o grabará en disco?
- ¿En caso de listados, cuáles serán los títulos? ¿Se incluirá pie de página?
- ¿Para una salida en pantalla, cuál es el formato de esta, se considera la posibilidad de impresión?
- ¿Para grabación en archivos , cuál es la descripción del registro? ¿En qué formato se grabarán los datos?

UN ASPECTO QUE DEBE SER CUIDADOSAMENTE PLANIFICADO ES LA INTERACCIÓN CON EL USUARIO. ESTA DEBE HACERSE EN EL LENGUAJE MÁS SIMPLE POSIBLE Y POR NINGÚN MOTIVO INCLUIR TERMINOLOGÍA TÉCNICA QUE EL USUARIO NO TIENE PORQUE CONOCER.

CARACTERISTICAS DE UN BUEN PROGRAMA

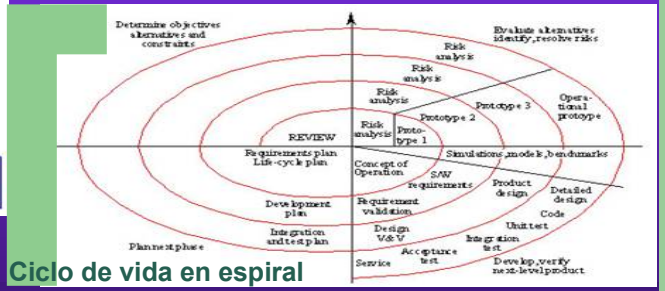
Un buen programa debería cumplir por lo menos con las siguientes características:

- EFICAZ: Que satisfaga las necesidades para las cuales fue creado, en otras palabras, que dadas las entradas produzca los resultados requeridos.
- EFICIENTE: Que consuma la menor cantidad de recursos (CPU, memoria, Sistema Operativo, etc.).
- LEGIBLE: Que su diseño sea simple, estructurado, ordenado, autodocumentado.
- FLEXIBLE: Fácil de adaptar a nuevos requerimientos.
- AMISTOSO: Mensajes claves, fácil de usar, presentación agradable.

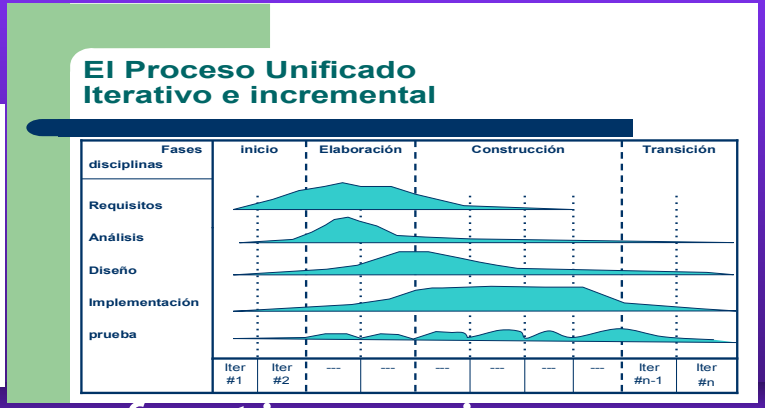
CICLO DE VIDA DE UN PROGRAMA

- El ciclo de vida de desarrollo de un programa es una sucesión de etapas por las que atraviesa el software desde que comienza un nuevo proyecto hasta que éste se deja de utilizar
- La elección de un modelo de ciclo de vida se realiza de acuerdo con la naturaleza del proyecto, los métodos a utilizar y los controles y entregas requeridos.

Ciclo de vida en cascada



Ciclo de vida en espiral



CICLO DE VIDA DE UN PROGRAMA

- ANÁLISIS: Consiste en realizar un estudio del problema que se va a solucionar, ver alternativas de solución y el correspondiente estudio de factibilidad.
- DISEÑO: Elaborar un modelo conceptual de la solución del problema. Deben quedar claramente establecidas las reglas que regirán el proceso.
- CODIFICACIÓN: Escribir el algoritmo resultante de la etapa anterior en algún lenguaje de programación.
- PRUEBAS: Verificar que el programa entregue los resultados esperados. Se debe tener especial cuidado con las condiciones de borde.

CICLO DE VIDA DE UN PROGRAMA

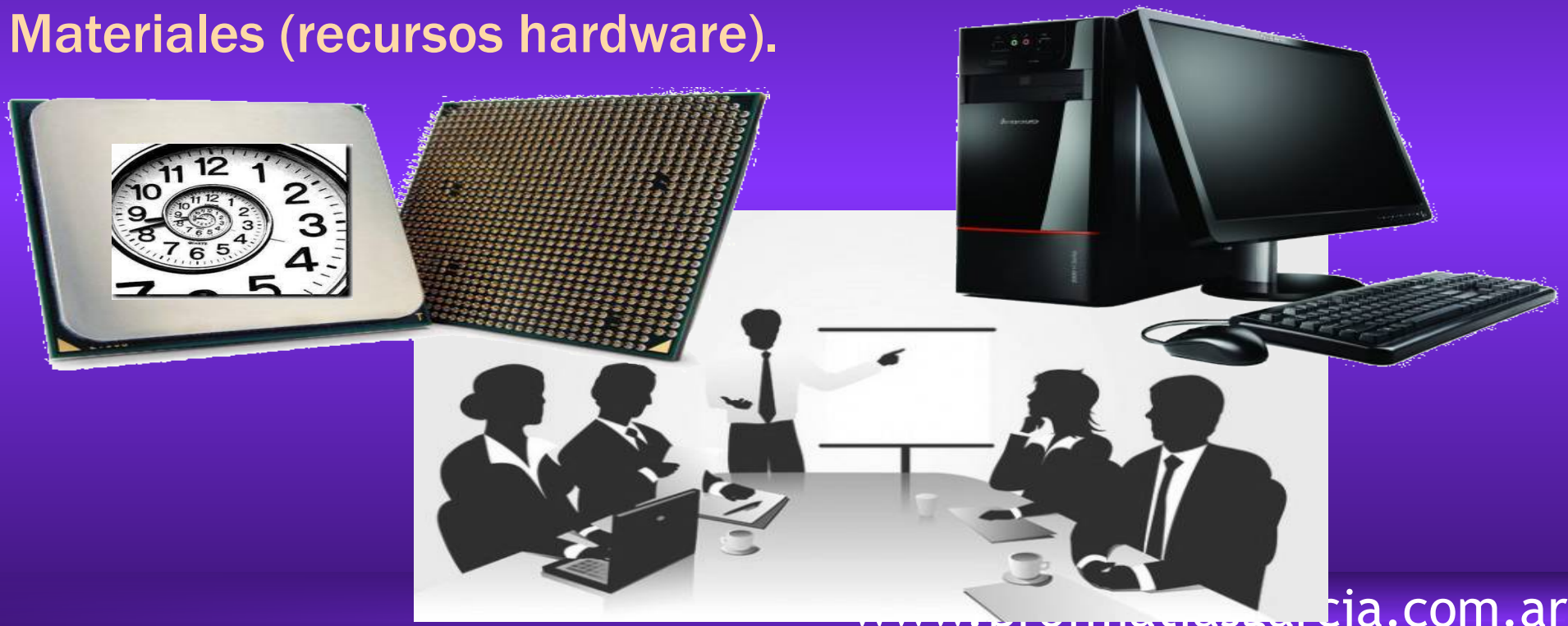
- PUESTA EN MARCHA: Ejecutar el programa con los datos reales, afinando últimos detalles para su explotación.
- EXPLOTACIÓN Y MANTENCIÓN: Va a estar presente desde que el programa fue entregado para su uso.
  - Solución de errores.
  - Adaptación computacional.
  - Mejoramiento (se adapta a nuevas necesidades por parte del usuario)
- DOCUMENTACIÓN: Debe quedar claramente definido que acción se debería seguir en caso de producirse algún error; siempre debe estar actualizada.

FACTORES DE COSTO

- Capacidad del programador.
- Complejidad del producto.
- Tamaño.
- Tiempo disponible.
- Nivel de confiabilidad requerido.
- Nivel tecnológico.

ELEMENTOS DE COSTO

- Tiempo de CPU.
- Horas hombre requeridos en análisis, programación, etc.
- Materiales (recursos hardware).



CARACTERÍSTICAS DE UN BUEN PROGRAMADOR

- Ordenado y metódico.
- Capacidad de abstracción, mentalidad analítica.
- Posee facilidad de comunicación.
- Capacidad de adaptación a nuevas situaciones.
- Previsor.
- Simplista.
- Perseverante.

FUNDAMENTOS DEL DISEÑO DE PROGRAMAS

El diseño es técnicamente la parte central de la ingeniería de software.
Durante el diseño se desarrollan, revisan y se documentan los refinamientos progresivos de las estructuras de datos, de la estructura del programa y de los detalles procedimentales. El diseño da como resultado representaciones cuya calidad puede ser evaluada.

FUNDAMENTOS DEL DISEÑO DE PROGRAMAS

Mediante algunas metodologías de diseño se realiza el diseño de datos, el diseño arquitectónico y el diseño procedimental.

- El diseño de datos transforma el modelo de campo de información, creado durante el análisis, en las estructuras de datos que se van a requerir para implementar el software.
- El diseño arquitectónico define las relaciones entre los principales elementos estructurales del programa.
- El diseño procedimental transforma los elementos estructurales en una descripción procedimental del software.

PROGRAMACION ESTRUCTURADA

A finales de los 60 se propuso la utilización de un conjunto de construcciones lógicas con las que podía formarse cualquier programa.

Cada construcción tenía estructura lógica predecible. Se entra por ella por el principio, se sale por el final y facilita al lector el seguimiento del flujo procedimental.

Las construcciones son secuencia, selección y repetición, y son fundamentales en la programación estructurada.

- La secuencia implementa los pasos de procedimiento esenciales de la especificación de cualquier algoritmo.
- La selección da la posibilidad de seleccionar un procedimiento basándose en alguna ocurrencia lógica.
- La repetición proporciona iteración.

ESTRUCTURA SECUENCIAL

Indica que las instrucciones de un programa se ejecutan una después de la otra, en el mismo orden en el cual aparecen en el programa.

```
INPUT x
INPUT y
auxiliar= x
x= y
y= auxiliar
PRINT x
PRINT y
```

Esta secuencia de instrucciones permuta los valores de x e y, con ayuda de una variable auxiliar, intermedia.

1. Se guarda una copia del valor de x en auxiliar.
2. Se guarda el valor de y en x, perdiendo su valor anterior, pero se mantiene una copia del contenido en auxiliar.
3. Se copia a y el valor de auxiliar, que es el valor inicial de x.

El resultado es el intercambio de los valores entre x e y, en tres operaciones secuenciales.

ESTRUCTURA SELECTIVA

Plantea la selección entre dos alternativas con base en el resultado de la evaluación de una condición; equivale a la instrucción IF de todos los lenguajes de programación

```
IF a > b THEN
  PRINT a ; " es mayor que " ; b
ELSE
  PRINT a ; " no es mayor que " ; b
END IF
```

La instrucción selectiva anterior puede presentar uno de dos mensajes: a es mayor que b o a no es mayor que b, según el resultado de la comparación entre a y b; si el resultado de a > b es verdadero, se presenta el primer mensaje, si es falso se exterioriza el segundo. Las palabras clave IF, THEN, ELSE, y END IF; constituyen la propia estructura de la instrucción condicional (palabras reservadas), proporcionada por el lenguaje, el usuario no debe utilizar sus nombres salvo para esta función.

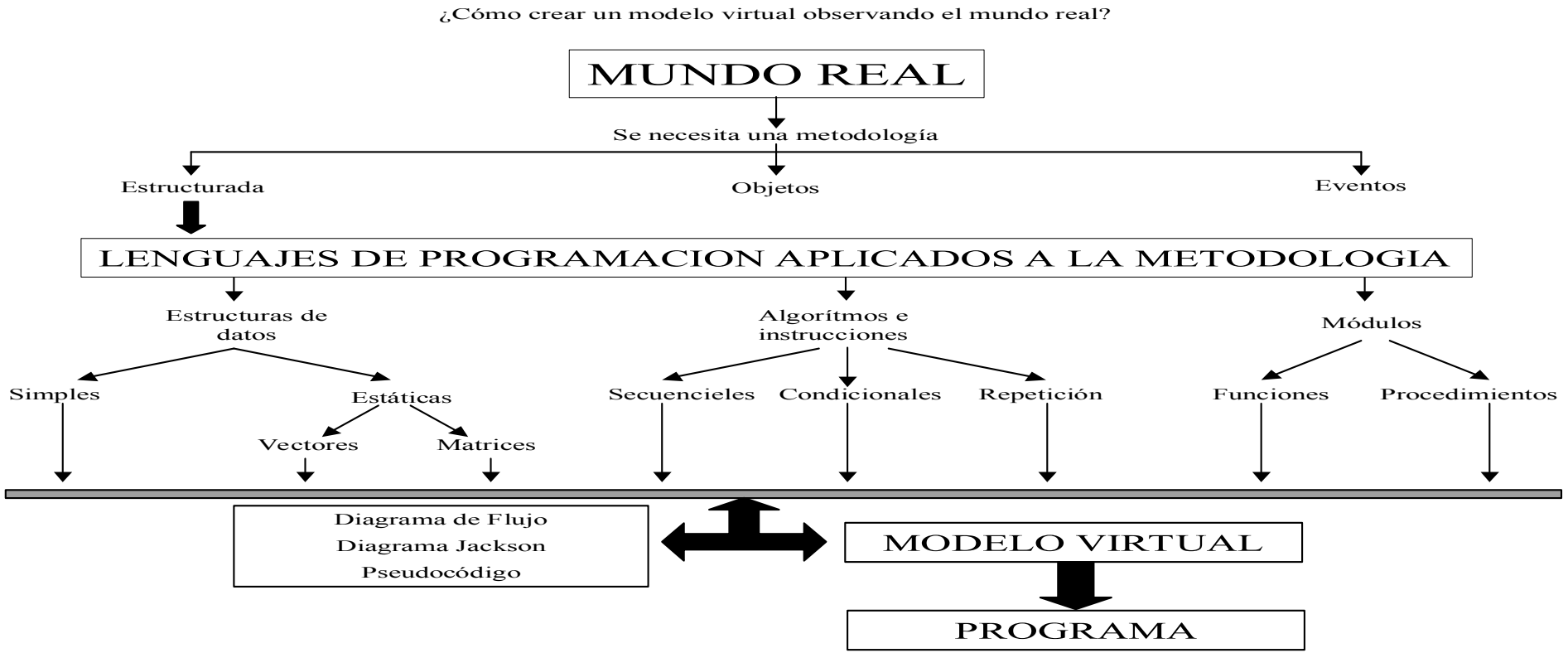
ESTRUCTURA REPETITIVA

Corresponde a la ejecución repetida de una instrucción mientras que se cumple una determinada condición.

```
a= 0
b= 7
DO WHILE b > a
   PRINT a
   a= a + 1
LOOP
```

El bucle mientras, se repite mientras la condición sea verdadera, esta condición se comprueba o chequea antes de ingresar al cuerpo del bucle, por lo que el mismo puede que no se ejecute nunca (cuando la condición es falsa desde un principio) o bien que se repita tantas veces como resulte y mientras la condición sea cierta.

MAPA CONCEPTUAL PROGRAMACION ESTRUCTURADA



VENTAJAS DE LA PROGRAMACION ESTRUCTURADA

- Los programas son más fáciles de entender. Un programa estructurado puede ser leído en secuencia, de arriba hacia abajo, sin necesidad de estar saltando de un sitio a otro en la lógica.
- Se logra una reducción del esfuerzo en las pruebas. El seguimiento de las fallas o depuración (debugging) se facilita debido a la lógica más visible, de tal forma que los errores se pueden detectar y corregir más fácilmente.
- Se crean programas más sencillos y más rápidos.
- Los bloques de código son casi auto-explicativos, lo que reduce y facilita la documentación.

BIBLIOGRAFÍA & LICENCIA

- Textos tomados, corregidos y modificados de diferentes páginas de Internet, tutoriales y documentos, entre los que destaco el libro: C/C++ Curso de programación, 2da Ed, Javier Ceballos, Alfaomega Ra-Ma.
- Este documento se encuentra bajo Licencia Creative Commons Attribution – NonCommercial - ShareAlike 4.0 International (CC BY-NC-SA 4.0), por la cual se permite su exhibición, distribución, copia y posibilita hacer obras derivadas a partir de la misma, siempre y cuando se cite la autoría del Prof. Matías E. García y sólo podrá distribuir la obra derivada resultante bajo una licencia idéntica a ésta.
- Autor:

Matías E. García

Prof. & Tec. en Informática Aplicada

www.profmatiasgarcia.com.ar

info@profmatiasgarcia.com.ar

www.profmatiasgarcia.com.ar