Universidad Nacional Autónoma de México

Facultad de Ingeniería

Materia: Fundamentos de Programación

Tarea: 2

Alumno: Andrik Uriel Reyes Roque

Fecha: 02/10/2020

INDICE

Breve historia del cómputo

El descubrimiento de la electricidad en el siglo XVIII también fue esencial, al igual que el conocimiento de cómo utilizarla a mediados del siglo XIX. La primera calculadora automática apareció en el siglo XVII, usando ruedas y engranajes para hacer cálculos.

En el siglo XIX, Joseph Jacquard (1752-1834) inventó un telar usando tarjetas perforadas unidas a agujas para decirle al telar qué hilos usar en qué combinaciones y colores. Con él, tejió patrones complejos en tela, todavía llamado diseño Jacquard. En el mismo siglo, Charles Babbage (1792-1871) diseñó un "motor de diferencias" para calcular e imprimir tablas matemáticas simples. Lo mejoró con su motor analítico usando tarjetas perforadas para realizar cálculos complejos, aunque nunca tuvo los fondos para construir uno.

Por lo tanto, a fines del siglo XIX, muchos elementos necesarios para hacer funcionar una computadora moderna estaban en su lugar: tarjetas de memoria, dispositivos de entrada, sistemas matemáticos, capacidades de almacenamiento, energía y sistemas de entrada.

Historia de la programación

Durante las tres primeras décadas de la informática el principal desafío era desarrollar el Hardware de las computadoras de forma que se redujera el costo de procesamiento y almacenamiento de datos. A lo largo de la década de los 80s´s los avances en microelectrónica han dado como resultado una mayor potencia de cálculo a la vez que una reducción de costos.

Durante la *primera era* de las computadoras el Hardware sufrió continuos cambios mientras el software se veía simplemente como un añadido ya que el desarrollo de software se realiza virtualmente sin ninguna planificación, Durante este periodo se utilizaba en la mayoría de los sistemas orientados por lotes.

En los primeros años la producción de software para ser vendido estaba en sus inicios debido a que el software se diseñaba medida para cada aplicación y tenía una distribución sumamente pequeña. La mayoría del Software se desarrollaba y utilizada por la misma persona u organización.

La *segunda era* se extiende desde la mitad de la década de los 60´s hasta finales de los 70´s. Está era se distingue por la aplicación de la multiprogramación y los sistemas multiusuarios Ya que ellos introdujeron nuevos conceptos de interacción hombre-máquina.

Los sistemas de tiempo real podían recoger analizar y transformar datos de múltiples Fuentes controlando así los procesos y produciendo salidas en milisegundos en vez de minutos.

La segunda era se caracterizó también por el uso de software como producto ya que se desarrollaba para una amplia distribución en un mercado multidisciplinario. Los programas se distribuían para computadoras grandes y minicomputadoras a cientos y a veces a miles de usuarios.

Con todo esto apareció un gran problema todos estos programas tenían que ser corregidos cuando se detectaron errores cambios en los requerimientos del usuario o bien adaptaciones al nuevo Hardware. Estas actividades se llamaron conjuntamente “mantenimiento de software”.

La *tercera era* de la evolución de los sistemas de computadoras comenzó a mediados de los 70´s. Aquí aparecen los sistemas distribuidos (computadoras múltiples cada un ejecutado funciones concurrentemente y comunicándose con alguna otra). Redes de área local y global, comunicaciones digitales de alto ancho de banda y la creciente demanda de acceso "instantáneo"

Esta era se caracteriza también por la llegada de los microprocesadores y las computadoras personales. Las computadoras personales fueron la base para el crecimiento de muchas compañías de software. El hardware de las computadoras personales se convierte rápidamente en un producto estándar mientras que el software suministrado con el mismo marca la diferencia.

La *cuarta era*. Ya las técnicas de la cuarta generación para el desarrollo de software están cambiando la forma en que algunos segmentos de la comunidad informática construyen los programas de computadora. Los sistemas expertos y el software de Inteligencia artificial finalmente se ha trasladado de laboratorio a algunas aplicaciones prácticas, en un amplio rango de problemas del mundo real.

Lista de los lenguajes de programación

1.- JAVA	**11.- Ruby**
2.- C	12.- Delphi / Object Pascal
3.- Python	13.- Objective-C
4.- C ++	14.- Go
5.- C #	15.- Lenguaje ensamblador
6.- Visual Basic .NET	16.- Visual Basic
7.- Javascript	17.- D
8.- PHP	18.- R
9.- Swift	19.- Perl
10.- SQL	20.- Matlab

Clasificación de los lenguajes de programación

Lenguajes imperativos: Los lenguajes imperativos o de procedimiento son lenguajes controlados por mandatos u orientados a enunciados (instrucciones). Un programa se compone de una serie de enunciados, y la ejecución de cada enunciado hace que el intérprete cambie el valor de una localidad o más en su memoria, es decir, que pase a un nuevo estado.

Lenguajes aplicativos. Un punto de vista alternativo de la computación representado por un lenguaje de programación consiste en examinar la función que el programa representa y no sólo los cambios de estado conforme el programa se ejecuta, enunciado por enunciado.

Lenguajes base en reglas. Los lenguajes con base en reglas se ejecutan verificando la presencia de una cierta condición habilitadora y, cuando se satisface, ejecutan una acción apropiada. El lenguaje más común con base en reglas es Prolog, que también se conoce como de programación lógico, puesto que las condiciones habilitadoras básicas son ciertas clases de expresiones lógicas de predicados.

Programación orientada a objetos. En este tipo de lenguaje, se construyen objetos complejos de datos y luego designa un conjunto limitado de funciones para que operen con esos datos Los objetos complejos se designan como extensiones de objetos más simples y heredan propiedades del objeto más sencillo. Al construir objetos concretos de datos, un programa orientado a objetos gana la eficiencia de los lenguajes imperativos, y al construir clases de funciones que utilizan un conjunto restringido de objetos de datos, se construye la flexibilidad y contabilidad del modelo aplicativo.

Tipos de paradigmas de programación

Existen diversos lenguajes y paradigmas de programación que se han diseñado para facilitar la tarea de la programación en diferentes ámbitos. Por ejemplo, la

programación orientada a objetos es un paradigma dirigido al mejoramiento en la calidad del software por medio de la observación de aspectos tales como la corrección, robustez, extensibilidad, compatibilidad y sobre todo la reusabilidad del software. La programación lógica, por su parte, es un paradigma orientado a la expresión de los problemas en términos lógicos para su posterior solución por métodos de inferencia y otras técnicas lógicas.

En la práctica, cada paradigma de programación es implementado a través de diversos lenguajes. Sólo como un ejemplo, la programación orientada a objetos encuentra su recipiente en lenguajes tales como Java, C++, Eiffel, Objective C. el paquete CLOS de Common Lisp, etc..

Bibliografía

http://fcasua.contad.unam.mx/apuntes/interiores/docs/98/4/informatica_4.pdf

PRESSMAN, S. ROGER, Ingeniería de Software, un enfoque práctico, ed. McGraw-Hill, primera edición, 1990.

FAIRLEY, E. RICHARD, Ingeniería de Sofware, ed. McGraw-Hill 1987