UNIVERSIDAD AUTONOMA DE ENCARNACION

LENGUAJE DE PROGRAMACION 3

RESPONSABLE:

JUAN ALBERTO GODOY

PAMELA GONZALEZ

DOCENTE: OSVALDO MICNIUK

CURSO: 3°

SEMESTRE: 5°

AÑO:2020

LENGUAJE DE PROGRAMACION 3

DESCRIPCION DE ACTIVDADES

a) ¿Qué es el paradigma de programación orientada a objetos?

El paradigma orientado a objetos se asocia con el proceso de programación llamado programación orientada a objetos (POO) consistente en un enfoque totalmente distinto al proceso procedimental. El enfoque orientado a objetos guarda analogía con la vida real. El desarrollo de software OO se basa en el diseño y construcción de objetos
que se componen a su vez de datos y operaciones que manipulan esos datos. El programador define en primer lugar los objetos del problema y a continuación los datos y operaciones que actuarán sobre esos datos. Las ventajas de la programación orientada a objetos se derivan esencialmente de la estructura modular existente en la vida real y el modo de respuesta de estos módulos u objetos a mensajes o eventos que se producen en cualquier instante

b) ¿Cómo se inició la programación Orientada a objetos?

Los orígenes de la POO se remontan a los Tipos Abstractos de Datos como parte constitutiva de una estructura de datos. En este libro se dedicará un capítulo completo al estudio del TAD como origen del concepto de programación denominado objeto.
Podemos considerar como inicio del paradigma de Programación Orientada a Objetos, el lenguaje de programación Simula desarrollado por Kristen en la mitad de los años 60 en el Centro de Computación Noruego (The Norwegian Computing Center). Simula se definió como un lenguaje de programación orientado a la simulación de procesos, con el que se podían definir distintos tipos de actividades. En este lenguaje aparecen por primera vez los conceptos de clases y objetos.

c) ¿A qué se conoce como la crisis del Software?

Se conoce como la crisis del software a la potencia de cálculo y disminución de los costes de esa nueva generación de ordenadores, más y más empresas adquirían esas máquinas demandando cada vez más programadores. El software se convirtió en un producto, y aparecieron las primeras empresas de desarrollo.
La rápida expansión de la demanda de software superó a la oferta de profesionales
especializados en el ámbito de la Informática. Ello obligó a las empresas de la época a contratar rápidamente personal sin formación, tras un proceso de selección y formación previo, pasaban a cubrir la mayoría de puestos de trabajo en desarrollo de software en empresas, administración, y servicios de procesamiento de datos.
Estos nuevos profesionales se senúan cómodos usando la técnica "codificar-probar-corregir". Sin embargo, esta estrategia tuvo nefastas consecuencias.

Conducía inevitablemente a códigos muy enrevesados, dando lugar a lo que se denominó "código
spaguetti"; los proy~s software no se acababan a tiempo. Este fenómeno que presentaba el software en esos años fue denominado crisis del software

d) ¿Cuáles fueron las causas de la Crisis del Software?

Las causas de la crisis del software fueron:

- ✓ Contratación de personas sin formación en el ámbito de la informática
- ✓ El software desarrollado no se deterioraba, pero quedaba obsoleto tras las nuevas tecnologías que iban apareciendo a lo largo de los años
- ✓ Se sobrepasaban los presupuestos
- ✓ No se prestaba atención a la calidad del producto final

e) ¿Posibles soluciones para la crisis del software?

La solución a la crisis del software se hizo posible con la llegada de la ingeniera del software lo cual permitió que se desarrollara una gran cantidad de modelos, lenguajes, métodos, técnicas y herramientas, todos orientados a la obtención de un software fiable, robusto, eficiente y de alta calidad

Se Construyen modelos y aprenden de ellos: analizan, diseñan, crean una maqueta o prototipo, testean y, finalmente, construyen y mantienen el producto construido. Este proceso aplicado en la ingeniería tradicional sirvió de base para establecer las diferentes fases implicadas en el desarrollo y gestión de un proyecto software:

- ❖ Análisis y especificación de requisitos (qué debe hacerse y cómo debe hacerse),
- ❖ Diseño (construcción del modelo),
- ❖ Codificación (construcción del producto)
- ❖ (verificación del producto) y mantenimiento.

f) ¿Herramientas de la POO para crisis del software?

La modularización, la abstracción y la herencia son herramientas para abordar la complejidad. Una modularización eficiente requiere que se generen módulos que satisfagan: „ Representen la abstracción de algo útil, relevante, fácil de conceptualizar y con una funcionalidad bien definida. „ Tengan una entidad propia e independiente del resto del sistema. „ De lugar a una estructura del sistema que sea simple, natural y poco sensible a los cambios evolutivos. „ Permita reducir la complejidad jerarquizando los módulos de forma que cada característica solo se describa una sola vez.

El análisis de un problema o el diseño de la aplicación software que lo resuelve debe llevarse a cabo mediante una eficiente modularización, lo cual requiere que el criterio que se utilice conduzca a la generación de módulos auténticos, esto es,

- o Representen la abstracción de algo significativo y útil: y como consecuencia fácil de conceptualizar y de describir mediante una interfaz sencilla.
- o Tengan una entidad propia e independiente del resto del sistema: y que por ello, se puedan diseñar, implementar, probar y utilizar de forma independiente y descentralizada.

- De lugar a una estructura del sistema que sea simple, natural y poco sensible a los cambios evolutivos.
- Permita organizar los componentes de forma que no se repita la información común.

REFERENCIAS

- https://rua.ua.es/dspace/bitstream/10045/4042/1/tema11.pdf
- http://villarod.blogspot.com/2007/02/la-crisis-del-software-y-sus-posibles.html
- https://www.ctr.unican.es/asignaturas/MC_OO/Doc/OO_08_I1_OrientadoObjeto.pdf