Universidad de San Carlos de Guatemala
Escuela de Ingeniería en Ciencias y Sistemas, Facultad de Ingeniería
Introducción a la programación y computación 2, 1er. Semestre 2021.

BITACORAS EN UN SERVIDOR WEB

201801146 – Daniel Enrique Coti Peñate

Resumen

Comúnmente los softwares contienen problemas o generan problemas en la ejecución del mismo, por lo cual es necesario detectar los problemas que comete y la cantidad de ellos; para lo cual se debe de generar un reporte (bitácora) con los datos del usuario que lo reporta, entre dichos datos se puede comentar: las fechas de ejecución, el usuario que reporta, los usuarios afectados y los errores que han sido registrados (código de error y su descripción).

Para lo cual se solicito crear una pagina web en la cual se puedan cargar documentos con formato xml que lleven dicha información, depurarla y con la misma generar un reporte con formato xml que contenga la información recaba de forma óptima y simplificada para el usuario que trabajará dicho software y a su vez se podrá visualizar en la misma página la obtención de datos para garantizar la transparencia de este.

Palabras clave

Abstracción

Se basa en poder describir un objeto con propiedades y métodos principales sin pensar en detalle. La abstracción separa el comportamiento específico de un objeto, a esta división que se realiza se le conoce como la barrera de abstracción. (Lara, 2017)

Summary

Commonly the softwares contain problems or cause problems in the execution of the same, so it is necessary to detect the problems it commits and the amount of them; for which a report (log) must be generated with the data of the user reporting it, among that data can be commented: the execution dates, the user reporting, the affected users and the errors that have been logged (error code and its description).

For which I request to create a web page in which you can upload xml-formatted documents that carry such information, debug it and with it generate an xml-formatted report containing the information collected in an optimal and simplified way for the user who will work such software and in turn you can view on the same page the obtaining of data to ensure the transparency of the same page.

Keywords

Abstraction

It is based on being able to describe an object with main properties and methods without thinking in detail. Abstraction separates the specific behavior of an object, this division that is performed is known as the abstraction barrier.

Clases

Una clase (en programación) es una agrupación de datos (variables o campos) y de funciones (métodos) que operan sobre esos datos. A estos datos y funciones pertenecientes a una clase se les denominan variables y métodos o funciones miembro. (Anonimo, s.f.)

Nodo

es un componente que forma parte de una red. En otras palabras, tanto si se trata de Internet como de Intranet (utilizada en ámbitos cerrados, con acceso limitado a los usuarios autorizados), cada servidor u ordenador constituye un nodo y se encuentra conectado a otro u otros nodos. (Gardey, 2009)

XML

proviene de extensible Markup Language ("Lenguaje de Marcas Extensible"). Se trata de un metalenguaje (un lenguaje que se utiliza para decir algo acerca de otro) extensible de etiquetas que fue desarrollado por el Word Wide Web Consortium (W3C), una sociedad mercantil internacional que elabora recomendaciones para la World Wide Web. (Gardey, Definicion.de, 2010)

Backend

Es un el sistema corporativo que se utilizan para dirigir una web o empresa, tales como sistemas de gestión de pedidos, inventario y procesamiento de suministro. Este sistema recoge información de los usuarios u otros sistemas de tratamiento de datos en la compañía. Es el encargado de gestionar la información que proporciona el usuario recogido por el sitio web.

Classes

A class (in programming) is a grouping of data (variables or fields) and functions (methods) that operate on that data. This data and functions belonging to a class are called variables and member methods or functions. (Anonymous, s.f.)

Node

is a component that is part of a network. In other words, whether it is the Internet or Intranet (used in closed scopes, with limited access to authorized users), each server or computer is a node and is connected to another or other nodes.(Gardey, 2009)

XML

comes from extensible Markup Language. It is a label-extendable metalanguage (a language used to say something about another) that was developed by the Word Wide Web Consortium (W3C), an international trading company that develops recommendations for the World Wide Web. (Gardey, Definicion.de, 2010)

Backend

Thisis a corporate system used to run a website or company, such as order management systems, inventory, and supply processing. This system collects information from users or other data processing systems in the company. It is responsible for managing the information provided by the user collected by the website

Introducción

Basándose en el Paradigma de Programación Orientado a Objetos, específicamente en los Tipos de Datos Abstractos (TDA) y de forma particular en este caso se trabajará por medio de Listas Enlazadas las cuales adoptarán el papel de recibir los datos que se estarán enviando desde un archivo XML; dichos datos a su vez serán almacenados en nodos que crean dichas listas enlazadas.

Así mismo se hace la salvedad que en dicho (XML) se están enviando lo que son Matrices con dimensiones nxm las cuales serán recibidas en las listas que se mencionan anteriormente. Las cuales tendrán la intención de poder convertir dichos datos en una matriz binaria y a su vez creando una matriz patrón, la cual tiene el fin de poder comparar filas y sumar las mismas que cumplan con el patrón de x fila. Para final con la escritura de un archivo (XML) enviando los datos de las sumas entrantes.

Desarrollo del tema

Para poder entrar en contexto se deberá de conocer como temas fundamentales los siguientes:

a) Programación Orientada a objetos
b) Tipos de Datos Abstractos
c) Listas Enlazadas
    a. Listas Enlazadas Simples
    b. Lista Doblemente Enlazada
    c. Listas Circulares
d) Vectores y Matrices
e) Que es XML
    a. Lectura y Escritura de archivos XML
f) Backend y Frontend

Para lo cual se empezará con:

PROGRAMACION ORIENTADA A OBJETOS:

Para hablar de la Programación Orientada a Objetos se debe de hablar de primero que es un Paradigma, lo cual se puede mencionar que un Paradigma es un modelo o patrón el cual se refiere a la forma que un programador pueda leer e interpretar el código de distintos programadores, dándole solución a los problemas que se le plantean al mismo.

A su vez se puede mencionar que hay distintos Paradigmas de Lenguajes de programación entre ellos:

- Paradigma Estructural
- Paradigma Funcional
- Paradigma Modular
- Paradigma Lógico
- Paradigma Procedimental
- Paradigma Orientado a Eventos
- Paradigma Orientado a Objetos

Mas en nuestro caso se hablará del Paradigma Orientado a Objetos o también llamado Programación Orientada a Objetos (POO).

La programación orientada a objetos se puede mencionar que se refiere a una manera especifica de poder organizar el código por medio de pequeños fragmentos del mismo llamadas *clases*, de las cuales se puede obtener *Objetos*¸ los cuales podrán interactuar con el resto del código para obtener el funcionamiento adecuado de la aplicación (APP) y/o software (Programa).

La intención de este paradigma es el poder programar de una forma especial, y que se observe como objetos de la vida real; con lo cual se puede aprender a resolver los problemas de maneras

distintas a las que se conocían anteriormente en la programación Estructural.

Para poder hablar de dicho paradigma se debe de conocer de igual manera los conceptos de clases y objetos los cuales se describirán a continuación:

a) *Clases: es una agrupación de datos (variables o campos) y de funciones (métodos) que operan sobre esos datos. A estos datos y funciones pertenecientes a una clase se les denominan variables y métodos o funciones miembro. (Anónimo, s.f.)*
b) *Objetos: consta de un ente perteneciente al POO la cual consta de datos y tareas que puede realizar en el transcurso del programa; los cuales se puede instanciar a partir de una clase.*

TIPOS DE DATOS ABSTRACTOS (TDA)

Para poder hablar de los tipos de datos abstractos debemos de hacer referencia al concepto de *Abstracción o Encapsulamiento*, el cual es un tema fundamental para dicho de tipos de datos.

- *Abstracción: se puede mencionar que la abstracción o separación de la especificación de un objeto o algoritmo de su implementación, en base a la utilización que se le dará en el programa y solo depende de una interfaz explícitamente definida (la especificación) y no de los detalles de su representación física (la implementación) ya que están ocultos.*

Conociendo el concepto de abstracción se puede mencionar que los Tipos de Datos Abstractos son Tipos de Datos que puede crear el usuario (programador) los cuales se realizan a partir del análisis y la abstracción de los datos y recursos que se manejaran en las mismas, así mismo se puede mencionar varios tipos de TDA'S entre ellas se mencionan:

- Colas
- Pilas
- Listas enlazadas

Se menciona que cada TDA constara de su especificación la cual será independiente de su implementación y a su vez nos proporcionara las operaciones que podrá realizar las cuales nos determinaran su utilización.

COLAS:

Colección de elementos homogéneos (del mismo tipo: Tipo Elemento) ordenados cronológicamente (por orden de inserción) y en el que sólo se pueden añadir elementos por un extremo (final) y sacarlos sólo por el otro (frente). Es una estructura FIFO (First In First Out):

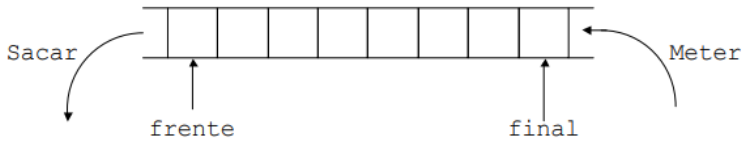

Ilustración I. TDA Cola

Fuente: Departamento de Lenguajes y Ciencias de la Computación.

PILAS:

Colección de elementos homogéneos (del mismo tipo: Tipo Elemento) ordenados cronológicamente (por orden de inserción) y en el que sólo se pueden añadir y extraer elementos por el mismo extremo, la cabeza. Es una estructura LIFO (Last In First Out):

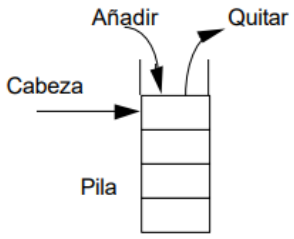

Ilustración II. TDA Pila

Fuente: Departamento de Lenguajes y Ciencias de la Computación.

LISTAS ENLAZADAS

Colección de elementos homogéneos (del mismo tipo: Tipo Elemento) con una relación LINEAL establecida entre ellos. Pueden estar ordenadas o no con respecto a algún valor de los elementos y se puede acceder a cualquier elemento de la lista.

- Hay que tener en cuenta que la posición de la inserción no se especifica, por lo que dependerá de la implementación. No obstante, cuando la posición de inserción es la misma que la de eliminación, tenemos una subclase de lista denominada pila, y cuando insertamos siempre por un extremo de la lista y eliminamos por el otro tenemos otra subclase de lista denominada cola.
- En el procedimiento Eliminar el argumento elem podría ser una clave, un campo de un registro Tipo Elemento.

Implementación

La implementación o representación física puede variar:

- Representación secuencial: el orden físico coincide con el orden lógico de la lista.
- Ejemplo: arrays. Para insertar o eliminar elementos podría exigirse que se desplazaran los elementos de modo que no existieran huecos. Por otra parte, tiene la desventaja de tener que dimensionar la estructura global de antemano, problema propio de las estructuras estáticas.

Ilustración III. Ejemplo de Listas

- Representación enlazada: da lugar a la lista enlazada ya estudiada, en la que el orden físico no es necesariamente equivalente al orden lógico, el cual viene determinado por un campo de enlace explícito en cada elemento de la lista. Ventaja: se evitan movimientos de datos al insertar y eliminar elementos de la lista.

Podemos implementar una lista enlazada de elementos mediante variables dinámicas o estáticas (el campo enlace será un puntero o un índice de array):

a) Variables de tipo puntero: Esta representación permite dimensionar en tiempo de ejecución (cuando se puede conocer las necesidades de memoria). La implementación de las operaciones es similar a las ya vista sobre la lista enlazada de punteros.

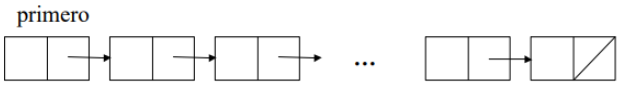

Ilustración IV. Punteros

b) Variables estáticas: se utilizan arrays de registros con dos campos: elemento de la lista y campo enlace (de tipo índice del array) que determina el orden lógico de los elementos, indicando la posición del

siguiente elemento de la lista (simulan un puntero). Esta representación es útil cuando el lenguaje de programación no dispone de punteros o cuando tenemos una estimación buena del tamaño total del array.

a. Ejemplo: Una lista formada por A, B, C, D, E (en este orden) puede representarse mediante el siguiente array de registros:

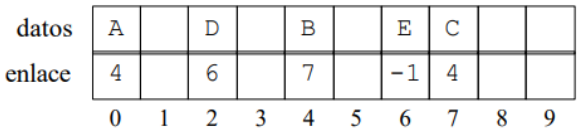

Ilustración V. composición de Listas

LISTA DOBLEMENTE ENLAZADA:

Son listas en las que cada nodo, además de contener los datos información propios del nodo, contiene un enlace al nodo anterior y otro al nodo siguiente:

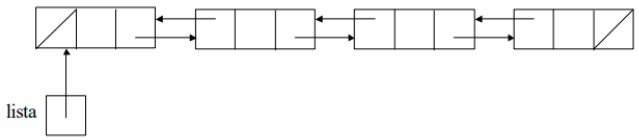

Ilustración VI. Lista Doblemente Enlazada 1

Aparte de que ocupan más memoria (el tamaño de un puntero por cada nodo más), los algoritmos para implementar operaciones para listas dobles son más complicados que para las listas simples porque requieren manejar más punteros. Para insertar un nodo, por ejemplo, habría que cambiar cuatro punteros (o índices si se simulan con un array):

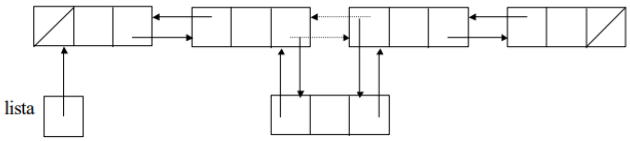

Ilustración VII. Lista Doblemente Enlazada 2

LISTAS CIRCULARES:

Son listas en las que el último elemento está enlazado con el primero, en lugar de contener el valor NULL o NULO. Evidentemente se implementarán con una representación enlazada (con punteros o variables estáticas). Con punteros sería:

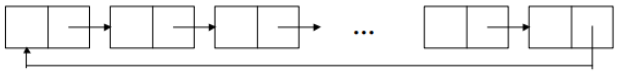

Ilustración VIII. Lista Circular

UTILIZACION DE LISTAS:

Las listas se utilizan en casi todo tipo de software, especialmente su representación enlazada con punteros. Un ejemplo de aplicación del TAD lista son las tablas hash con encadenamiento de sinónimos. La tabla sería un array de listas de longitud variable: un nodo por cada colisión, lo que evitaría sobredimensionar la tabla y además su posible desborde:

```
/* TIPOS */
  typedef ¿? TipoElemento /* un registro p. ej. */
  TipoLista TablaHash[NmEmpl];
```

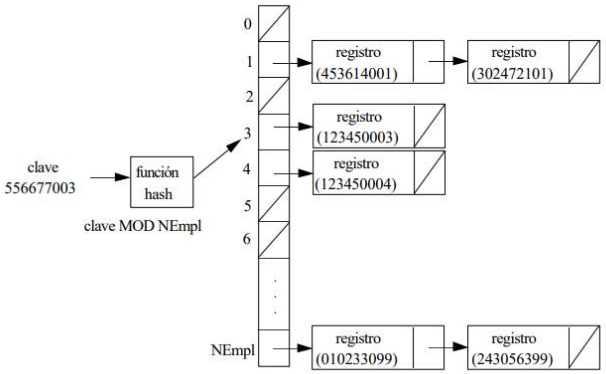

Ilustración IX. Ejemplo de la utilización de Listas

VECTORES Y MATRICES

En programación, una matriz o vector (llamados en inglés arrays) es una zona de almacenamiento continuo, que contiene una serie de elementos del mismo tipo, los elementos de la matriz. Desde el

punto de vista lógico una matriz se puede ver como un conjunto de elementos ordenados en fila (o filas y columnas si tuviera dos dimensiones).

En principio, se puede considerar que todas las matrices son de una dimensión, la dimensión principal, pero los elementos de dicha fila pueden ser a su vez matrices (un proceso que puede ser recursivo), lo que nos permite hablar de la existencia de matrices multidimensionales, aunque las más fáciles de imaginar son los de una, dos y tres dimensiones.

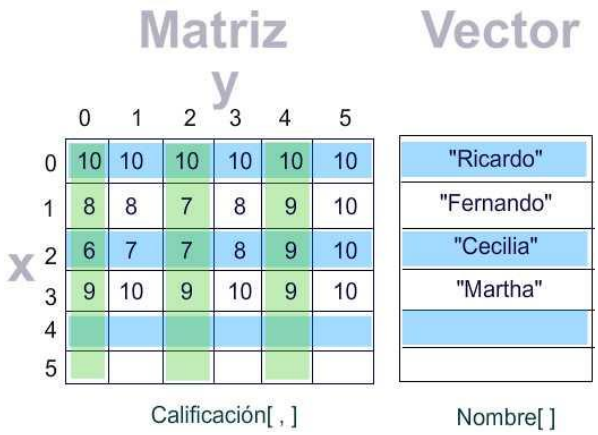

Ilustración X. Vectores y Matrices

QUE ES XML

XML es un subconjunto de SGML (Estándar Generalised Mark-up Language), simplificado y adaptado a Internet

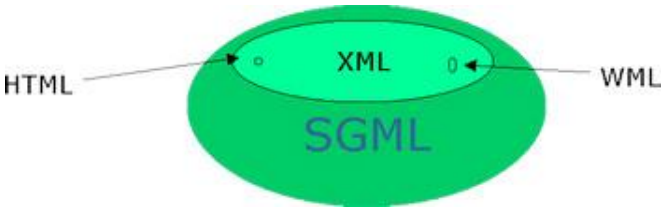

Ilustración XI. SGML

- XML no es, como su nombre puede sugerir, un lenguaje de marcado.
- XML es un metalenguaje que nos permite definir lenguajes de marcado adecuados a usos determinados.

CARACTERISTICAS

- XML es un subconjunto de SGML que incorpora las tres características más importantes de este:
  - Extensibilidad
  - Estructura
  - Validación
- Basado en texto.
- Orientado a los contenidos no presentación.
- Las etiquetas se definen para crear los documentos, no tienen un significado preestablecido.
- No es sustituto de HTML.
- No existe un visor genérico de XML.

ESTRUCTURA DE UN DOCUMENTO XML

Un documento XML está formado por datos de caracteres y marcado, el marcado lo forman las etiquetas:

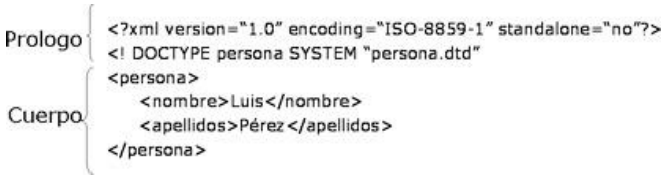

Ilustración XII. Estructura de un Archivo XML

ESTRUCTUCA DE ETIQUETAS

Ilustración XIII. Estructura de Etiquetas en XML

COMPONENTES DE UN DOCUMENTO XML

- En un documento XML existen los siguientes componentes:
    - Elementos: Pieza lógica del marcado, se representa con una cadena de texto(dato) encerrada entre etiquetas. Pueden existir elementos
    - vacíos (
). Los elementos pueden contener atributos.
    - Instrucciones: Ordenes especiales para ser utilizadas por la aplicación que procesa
    - <?xml-stylesheet type= “text/css” href= “estilo.css”>
    - Las instrucciones XML. ¿Comienzan por <? ¿Y terminan por?>.
    - Comentarios: Información que no forma parte del documento. ¡Comienzan por <! -- y terminan por -- >.
    - Declaraciones de tipo: Especifican información acerca del documento:
    - <!DOCTYPE persona SYSTEM “persona.dtd”>
    - Secciones CDATA: Se trata de un conjunto de caracteres que no deben ser interpretados por el procesador:
    - <! [CDATA [Aquí se puede meter cualquier carácter, como <, &, >, ... Sin que sean interpretados como marcación]]>

BACKEND Y FRONTED

Un backend es un el sistema corporativo que se utilizan para dirigir una web o empresa, tales como sistemas de gestión de pedidos, inventario y procesamiento de suministro. Este sistema recoge información de los usuarios u otros sistemas de tratamiento de datos en la compañía. Es el encargado de gestionar la información que proporciona el usuario recogida por el sitio web.

Un sistema de backend es cualquier sistema que soporta aplicaciones de “back office”. Estos sistemas se utilizan como parte de la gestión social y funcionan mediante la obtención de los datos de la entrada del usuario en el sitio y reunir las aportaciones de otros sistemas para proporcionar una salida de respuesta.

El Front end es la parte de una web que conecta e interactúa con los usuarios que la visitan. Es la parte visible, la que muestra el diseño, los contenidos y la que permite a los visitantes navegar por las diferentes páginas mientras lo deseen. Es una de las dos mitades en las que se divide la estructura de cualquier página web.

Junto a esta se encuentra el Back end, que es el polo completamente opuesto, la capa que accede a datos y software en general para su comunicación. Ambas se reúnen en cualquier site que visite una persona y son las que, trabajando, hacen que funcione en todo momento tal y como lo hace.

Conclusiones

A. Se presento el análisis de los temas utilizados para el proyecto en función, con el fin de tener una orientación para la ejecución de este.
B. Se conoció los conceptos básicos de la Programación Orientación a Objetos con el fin de utilizar de forma adecuada las TDA en la operación del proyecto.
C. La utilización de documentos XML como base para la estructuración de datos es de utilidad ya que a partir de cada etiqueta se podrá obtener diversos datos que se utilizará en el mismo proceso.

Referencias

Anonimo. (s.f.). Obtenido de https://www.ecured.cu/Clases_(programaci%C3%B3n)#:~:text=Una%20clase%20%28en%20programaci%C3%B3n%29%20es%20una%20agrupaci

%C3%B3n%20de,Objetos%20se%20basa%20en%20la%20programaci%C3%B3n%20de%20clases.

Gardey, J. P. (2009). *definicion.de*. Obtenido de https://definicion.de/nodo/

Gardey, J. P. (2010). *Definicion.de*. Obtenido de https://definicion.de/xml/

Lara, D. (6 de julio de 2017). *Laraveles*. Obtenido de https://laraveles.com/series/poo/la-abstraccion-programacion-orientada-objetos/