

Introducción a la Ingeniería de Software

Unidad 3. Diseño, codificación, pruebas y mantenimiento



Docente en línea: *Mtra. Adriana Álvarez Gutiérrez*

3.2. Codificación

3.2.1. Traducción de diseño a código

3.2.2. Codificación de la interfaz

3.2.3. Herramientas de desarrollo: gestión de la configuración

Introducción

Cuando se ha finalizado la fase del diseño, el siguiente paso es la codificación.

La codificación se traduce de los algoritmos dando inicio con la construcción de programa mediante la sintaxis del lenguaje de programación que ha resolver las peticiones del cliente.

La estructuración de esta fase requiere ser desarrollado por módulos con el fin de ser integrados en un solo bloque.

Competencia general

Conocer los lineamientos de codificación, y sus principales características de acuerdo con un caso de estudio donde se relacione con el módulo establecido del diseño del sistema de software.

Competencia específica

- Analizar e identificar el proceso de la fase de codificación mediante un caso de estudio para resolver los requerimientos del cliente que integrarán el software.

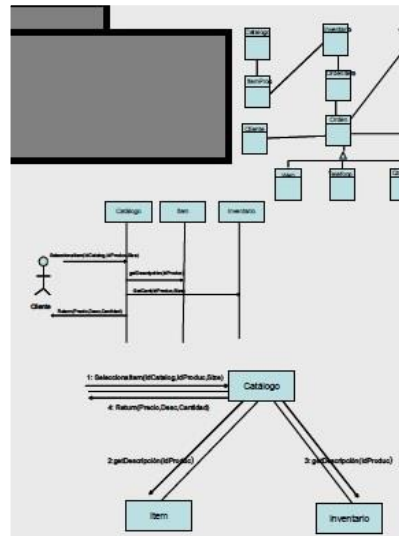
3.2.1. Traducción de diseño a código

Función que realiza el programador con base a los lineamientos establecidos en la **fase de diseño** tomando en cuenta los requisitos funcionales y no funcionales definidos en la **fase de Análisis**.

El software contará con un módulo de ayuda de cada uno de los procesos y ventanas.
También se puede disminuir la ambigüedad del requerimiento indicando que el personal será capacitado por un cierto periodo de tiempo estimado para que pueda aprender a utilizarlo.

¿Qué?

Análisis



¿Cómo?

Diseño



Nuevo teclado
profesional para
programadores



Codificación

Codificación

Diseño

Herramientas

Entorno

**Lenguaje de
programación**

Criterios de selección del lenguaje

1. Petición del cliente.
2. Tipo de aplicación:
 - Orientado a Objetos: Java, C++
 - Orientados a la Web: Perl, PHP, JavaScript
 - Dirigido por eventos: Java, HTML, Visual Basic
 - Inteligencia artificial: Lisp y Prolog
 - Ciencias: Fortran, Pascal y C
3. Clasificación de lenguaje: bajo nivel y alto nivel
4. Recursos y entorno de desarrollo:
 - Flexibilidad en el diseño y desarrollo de aplicaciones
 - Flexibilidad en el mantenimiento de los elementos.
 - Factibilidad Programación
 - Factibilidad Conversión.
 - Requerimientos de lenguaje
5. Experiencia en el manejo del lenguaje de programación
6. Reutilización.
7. Compatibilidad de compiladores

Recomendaciones



Retomar Planteamiento del problema



Revisar el diseño

```
/**
 * @author Magus
 */
public class Main {

    /** Creates a new instance of Main */
    public Main() {
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here
    }
}
```

Documentar y estructurar el código de programa

Variables en java

los:

	Declaración e inicialización separadas	Declaración e inicialización en una sola instrucción
Variable de tipo char (carácter) llamada k	char k; k = 's';	char k = 's';
Variable de tipo boolean (lógico) llamada m	boolean m; m=true;	boolean m = true;
Variable de tipo double (real) llamada salario	double salario; salario = 100.5;	double salario = 100.5;

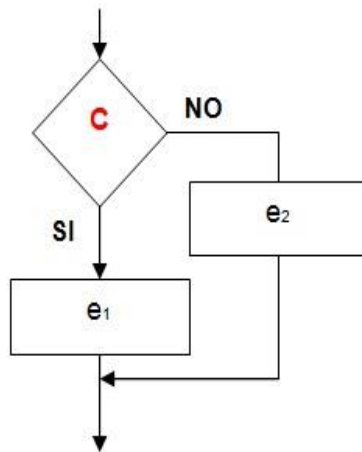
Identificadores y Variables

Licencia de Software

Recursos

3.2.2. Codificación de la interfaz

Diagrama de Flujo



```
clase plantilla ordenador{
    c1,c2,c3,.....cn //son celdas de memoria
    ve //variable de estado
    n //números leídos
públicos:
    plantilla_ordenador(): //es el constructor
        Inicio_pantalla
        c1=c2=c3=.....=cn=0
        ve=-1
        n=0
        Fin_plantilla
    Leer_numeros():
        Inicio_leer
        l=0
        Mientras Haya_numeros
            l=l+1
            Leer c1
        Fin_mientras
```

Manual de Procedimientos

MANUAL DE PROCEDIMIENTOS [Técnico].

1. INTRODUCCIÓN

Este manual está dirigido a la empresa, con la finalidad de proporcionarle herramientas de solución en caso de que el sistema falle, así como también bases técnicas para futuras actualizaciones y mejoras del mismo.

Sirve como manual de consulta, para personas con conocimientos de computación y programación, cuando el sistema tenga algún fallo o error para su solución.

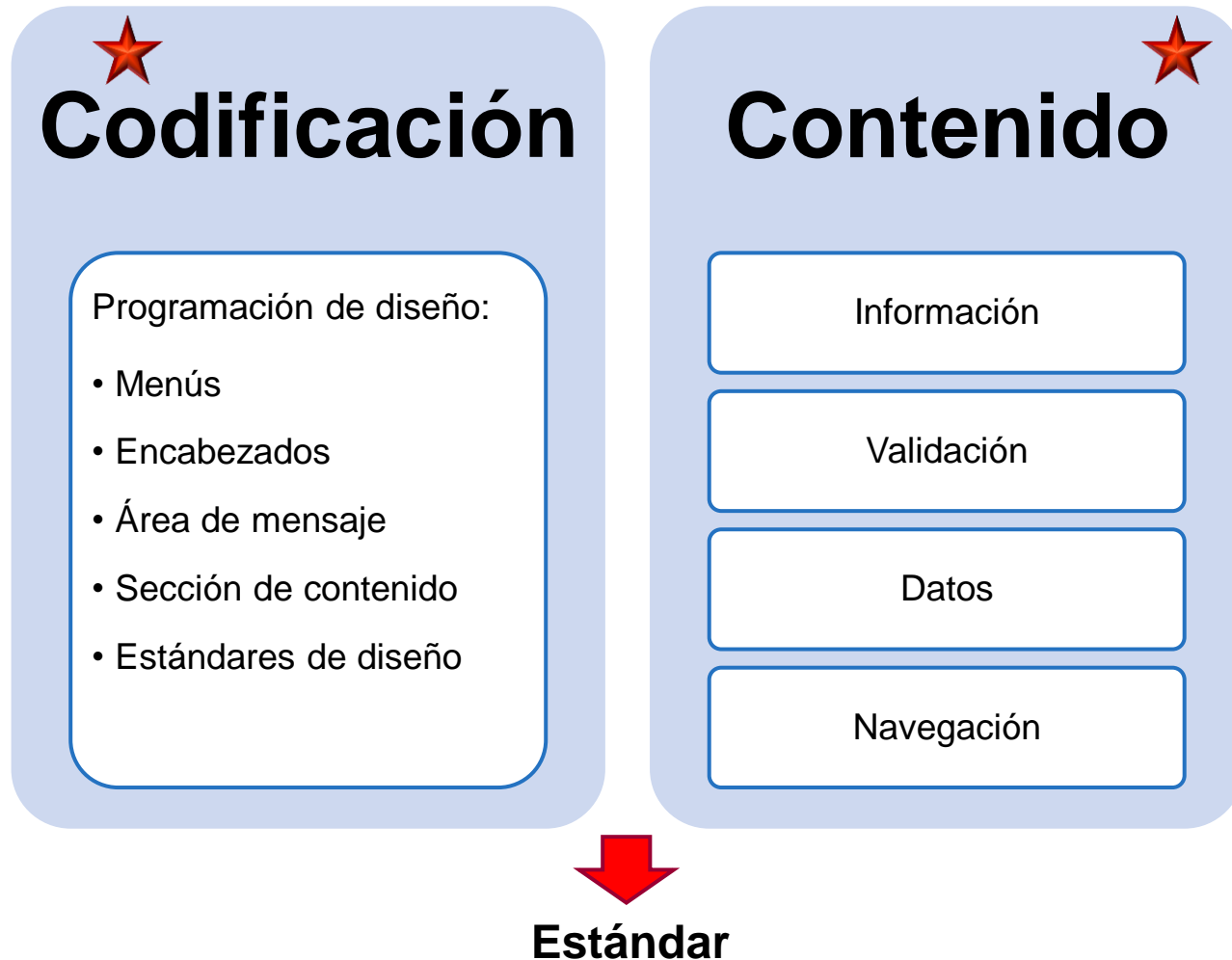
Es la implementación del diseño y la aplicación de estándares

3.2.2. Codificación de la interfaz

Uso de generadores automáticos de interfaces. Por ejemplo: *WYSIWYG Web Builder* que significa “Lo que ves es lo que obtienes”.

WYSIWYG Web Builder **11**

Construcción de interfaces



3.2.3. Herramientas de desarrollo: gestión de la configuración

Las herramientas de gestión de la configuración agilizan y mantienen la integridad de la organización del proyecto para evitar pérdidas de información, duplicidad de archivos y errores. Por ejemplo una administración configurada en:

- 1.Administración del cambio
- 2.Gestión de versiones
- 3.Construcción del sistema
- 4.Gestión de entregas

CVS es una herramientas de software que genera repositorios para administrar los documentos de lo proyectos y favoreciendo la administración de versiones.

Otras **herramientas** son:

- Software Configuration Management (SCM)
- Microsoft Visual Studio ALM
- SmartBear
- CA Harvest Software Change Manager

Referencias

EcuRed. (2016). *Gestión de la configuración*. Recuperado de http://www.ecured.cu/Gesti%C3%B3n_de_la_Configuraci%C3%B3n

López Islas, N. (2011). *Análisis y Diseño de sistemas*. D.F., México: MC Graw Hill.

PhpBB Group. (2016). *WYSIWYG Web Builder 11*. Recuperado de <http://www.wysiwygwebbuilder.com/>

Adelante con las actividades.

!Gracias!