PROCESO DIRECCIÓN DE FORMACIÓN PROFESIONAL

INTEGRAL FORMATO GUÍA DE APRENDIZAJE

IDENTIFICACIÓN DE LA GUÍA DE APRENDIZAJE

- Denominación del Programa de Formación: Técnico en Programación de Software
- Código del Programa de Formación: 233104 Versión 1
- Nombre del Proyecto (si es formación Titulada): CODIFICACION DE APLICACIONES DE SOFTWARE PARA EL SECTOR EDUCATIVO Y EMPRESARIAL DEL DEPARTAMENTO DEL QUINDIO
- Fase del Proyecto (si es formación Titulada): Analisis
- Actividad de Proyecto (si es formación Titulada): Elaborar los diagramas de clase, de casos de uso y de secuencias de UML aplicados el al proyecto (herencia, asociación, dependencia, etc...).
- Competencia: Codificar la Solución de software
- Resultados de Aprendizaje Alcanzar: Interpretar la información técnica de diseño para la codificación del software.
- Duración de la Guía: 40 Horas

2. PRESENTACIÓN

- Motivar hacia la actividad de aprendizaje en consideración a las fortalezas que aportará en el desarrollo de habilidades y destrezas
- Guiar y organizar el aprendizaje de manera que se oriente al desarrollo integral del aprendiz
- Motivar a la acción, al trabajo autónomo sistemático y organizado
- Relacionar conocimientos previos con los nuevos para la construcción significativa de los mismos
- Promover el aprendizaje colaborativo y el crecimiento integral del grupo

INTRODUCCIÓN

El lenguaje unificado de modelado (UML, por sus siglas en inglés, Unified Modeling Language) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el Object Management Group (OMG).

Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos, funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y compuestos reciclados.

Es importante remarcar que UML es un "lenguaje de modelado" para especificar o para describir métodos o procesos. Se utiliza para definir un sistema, para detallar los artefactos en el sistema y para documentar y construir. En otras palabras, es el lenguaje en el que está descrito el modelo

3. FORMULACION DE LAS ACTIVIDADES DE APRENDIZAJE

- Descripción de la(s) Actividad(es)
- Ambiente Requerido
- Materiales

Tomando como referencia la planeación pedagógica y las orientaciones para elaborar guías de aprendizaje citado en la guía de desarrollo curricular

3.1. Actividades de Reflexión inicial.

"¿Te comprarías una casa nueva en construcción sin haber visto los planos? Seguramente la respuesta es no. De hecho, seguro que al vendedor le exiges que te muestre los planos a escala de lo que va a ser tu nueva vivienda. Los planos te van a servir para que puedas hacerte la idea de su distribución.

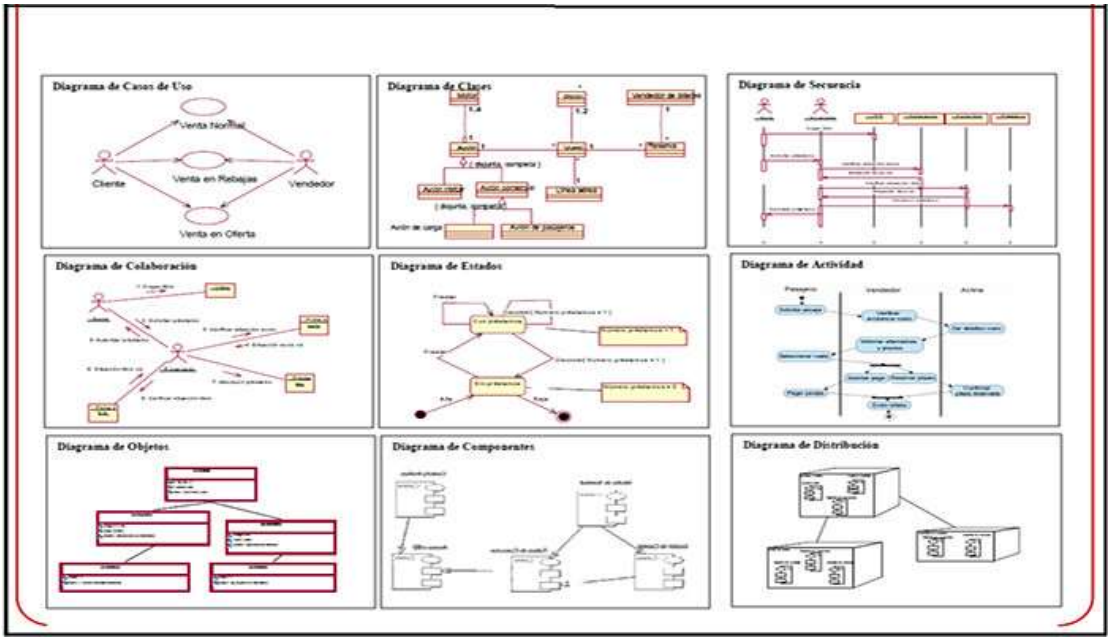


Sin embargo, hasta hace poco la situación con las aplicaciones software no era así. Aunque en algunos casos la inversión en un sistema software podía ser superior al precio de una vivienda, a los ingenieros de software no se nos exigía que entregásemos los planos de la aplicación antes de empezar a construirla. Sólo unos cuantos documentos textuales donde se describían los requisitos y la funcionalidad esperada. Era difícil saber qué iba a hacer la aplicación exactamente, e imposible el que los ingenieros de software pudiéramos razonar sobre ellos. Aun peor, una vez desarrollada la aplicación era muy difícil resolver las diferencias en caso de interpretaciones diferentes sobre lo escrito en esos documentos. Por no citar que el cliente quedaba totalmente en manos del proveedor cuando era preciso realizar alguna modificación en la aplicación, o sin posibilidad de irse a otro "constructor" para que hiciera alguna ampliación en caso de necesitarla.

Afortunadamente la situación va cambiando, y los informáticos usan ya modelos del software a la hora de comprender, diseñar, analizar y mantener una aplicación. Incluso más, los modelos

pueden usarse para generar el código final en muchos casos. Ya no es sólo el código Java, C++ o Python el único artefacto que compone una aplicación, sino que también hay modelos de alto nivel. Y al igual que ocurre con las ingenierías tradicionales, dichos modelos permiten razonar sobre la aplicación y deducir muchas de sus propiedades antes de desarrollar nada de código. Imagínense un ingeniero civil que no usase planos y modelos matemáticos para construir un puente y determinar el peso que soportará, o las cargas que ha de soportar un edificio, sino que construyera el puente y luego lo probase…

¿Y cómo son los modelos del software? Pues al igual que ocurre en un edificio o en una obra civil, los hay de muchos tipos, cada uno centrado en un aspecto concreto del sistema. Por ejemplo, el modelo de estructura describe con detalle los datos que maneja la aplicación. Otros modelos se encargan de especificar el comportamiento y las interacciones con los usuarios. Hay modelos para diseñar e implementar la apariencia de la aplicación (pantallas, diseño, etc.). Otros se encargan de la arquitectura software. En el caso de las aplicaciones distribuidas, hay modelos que expresan cómo los componentes que conforman la aplicación han de desplegarse en los nodos (ordenadores) en donde han de correr para que lo hagan de la forma más eficiente. Otros modelos permiten razonar sobre las prestaciones de la aplicación (rendimiento, tiempo de respuesta, uso de recursos compartidos, etc.).



Si no fuera usando modelos de alto nivel, cada uno centrado en un aspecto, sería imposible razonar sobre el sistema completo, analizar su comportamiento antes y después de construirlo, mantenerlo, o incluso desarrollarlo garantizando que va a cumplir con los requisitos del usuario (que, por cierto, también se recogen usando modelos).”

Tomado de: https://elpais.com/tecnologia/2016/11/22/actualidad/1479834209_075442.html

De la lectura anterior

¿cree usted que son necesarios los planos para crear software?
¿Qué pasaría si solamente creo software sin usar los planos?

¿Cuál es la ventaja de tener planos en la construcción de software?

3.2. Actividades de contextualización e identificación de conocimientos necesarios para el aprendizaje.

Se tiene que desarrollar el software de facturación para una empresa, indague acerca de que información hace parte de un sistema de facturación e indague como podría construir los planos para fabricar el software y que herramientas necesita para hacerlo.

3.3. Actividades de apropiación del conocimiento (Conceptualización y Teorización).

Diagramar los Modelos UML: Diagrama de clases, diagrama de objetos, casos de uso, secuencia, colaboración, estados, actividades, componentes y despliegue mediante una herramienta de diseño
(argoUML, StarUML, Dia).

Se establecerá un caso que permitirá realizar una simulación en el caso de construcción de un software para un restaurante y una biblioteca, se asignaran roles que permitirán realizar un trabajo simulando un entorno real de una restaurante y una biblioteca, se analizaran los procesos que intervienen y en equipos de trabajo, los aprendices construirán los Diagramas asignados por el instructor:

- Diagrama de clases
- Diagrama de objetos
- Casos de uso
- Secuencia
- Colaboración
- Estados
- Actividades
- Componentes
- Despliegue

La actividad se realizará en grupos de 3 aprendices y posteriormente se hará un conversatorio de cada tema, se designará un moderador y se concluirá en un informe las definiciones unificadas de cada tema, conseguidas mediante la intervención de los aprendices y las aclaraciones del instructor.

Una vez terminada la actividad se extraerán las conclusiones finales, se realizara la retroalimentación pertinente y se despejaran las dudas que surjan sobre la temática.

3.4. Actividades de transferencia del conocimiento.

Construir el prototipo del sistema de información a través de un Informe Final de Diseño del Proyecto aplicando todas las Técnicas y Modelos vistos durante la Formación. El instructor conformará los grupos de trabajo y después de realizar el informe se socializará a través de una Mesa Redonda(utilizando herramientas TIC), cada grupo deberá formular tres preguntas a los demás compañeros.

- Desarrolle una o más clases para las siguientes aplicaciones de software. Tenga en cuenta en definir atributos y métodos de la clase:

- Desarrolle las clases necesarias que permita dar solución a el proyecto de software que está desarrollando, recuerde que por cada clase debe definir sus atributos, propiedades y métodos necesarios.
- Codificar cada una de las clases aplicando los conceptos de Orientación a Objetos vistos anteriormente.

4. ACTIVIDADES DE EVALUACIÓN

Tome como referencia la técnica e instrumentos de evaluación citados en la guía de Desarrollo Curricular

Evidencias de Aprendizaje	Criterios de Evaluación	Técnicas e Instrumentos de Evaluación
Evidencias de Conocimiento :	**Respuestas a Cuestionario**	**Cuestionario**
Evidencias de Desempeño:	Elaboración de los diagramas de clase, de casos de uso y de secuencias, bien sea a mano o usando herramientas de software como ArgoUML o StarUML	Lista de Chequeo
Evidencias de Producto:	Construcción de diagrama de clases, de casos de uso y de secuencias, bien sea a mano o usando herramientas de software como ArgoUML o StarUML	Lista de Chequeo

5. GLOSARIO DE TERMINOS

Programación orientada a objetos: es un paradigma de programación que usa los objetos en sus interacciones, para diseñar aplicaciones y programas informáticos.

Objeto: Instancia de una clase. Entidad provista de un conjunto de propiedades o atributos (datos) y de comportamiento o funcionalidad (métodos), los mismos que consecuentemente

reaccionan a eventos. Se corresponden con los objetos reales del mundo que nos rodea, o con objetos internos del sistema (del programa). Es una instancia a una clase.

Clase: Definiciones de las propiedades y comportamiento de un tipo de objeto concreto. La instanciación es la lectura de estas definiciones y la creación de un objeto a partir de ellas.

Abstracción: Denota las características esenciales de un objeto, donde se capturan sus comportamientos. Cada objeto en el sistema sirve como modelo de un "agente" abstracto que puede realizar trabajo, informar y cambiar su estado, y "comunicarse" con otros objetos en el sistema sin revelar cómo se implementan estas características.

Tipo de dato: Tipo de dato informático es un atributo de una parte de los datos que indica al ordenador (y/o al programador) algo sobre la clase de datos sobre los que se va a procesar. Esto incluye imponer restricciones en los datos, como qué valores pueden tomar y qué operaciones se pueden realizar. Tipos de datos comunes son: enteros, números de coma flotante (decimales), cadenas alfanuméricas, fechas, horas, colores, etc.

Parámetros: Los parámetros se usan para mandar valores a los métodos. Un método trabajará con los parámetros para realizar las acciones. Por decirlo de otra manera, los parámetros son los valores de entrada que recibe un método.

Clase: una clase es una construcción que se utiliza como un modelo (o plantilla) para crear objetos de ese tipo. El modelo describe el estado y el comportamiento que todos los objetos de la clase comparten. Un objeto de una determinada clase se denomina una instancia de la clase. La clase que contiene (y se utilizó para crear) esa instancia se puede considerar como del tipo de ese objeto

Objeto: un objeto se define como la unidad que en tiempo de ejecución realiza las tareas de un programa. También a un nivel más básico se define como la instancia de una clase.

Atributos: un atributo es una especificación que define una propiedad de un Objeto.

Método: un método es una subrutina asociada exclusivamente a una clase (llamados métodos de clase o métodos estáticos) o a un objeto (llamados métodos de instancia).

6. REFERENTES BILBIOGRAFICOS

Construya o cite documentos de apoyo para el desarrollo de la guía, según lo establecido en la guía de desarrollo curricular

- http://msdn.microsoft.com/es-es/library/ms173114(v=vs.90).aspx (Consulta 06/03/2015)

 Introducción a la programación orientada a objetos. En línea. Disponible en:

http://luis.izqui.org/resources/ProgOrientadaObjetos.pdf (Consulta 06/03/2015)

 Que es la programación orientada a objetos. En línea. Disponible en:

http://www.desarrolloweb.com/articulos/objetos-lenguaje-informatico-introduccion.html

http://www.ciberaula.com/articulo/tecnologia_orientada_objetos/ (Consulta 06/03/2015)

7. CONTROL DEL DOCUMENTO

	Nombre	Cargo	Dependencia	Fecha
Autor (es)	Paula Andrea García Juan Manuel Cardona Jonathan Stick Campos Carolina Henao Otoniel Suarez Jorge Ivan Sierra Sebastian López Oscar Ramírez	Instructor Instructor Instructor Instructor Instructor Instructor Instructor Instructor	Articulación con la media centro de Comercio y turismo	Febrero de 2021

8. CONTROL DE CAMBIOS (diligenciar únicamente si realiza ajustes a la guía)

	Nombre	Cargo	Dependencia	Fecha	Razón del Cambio


SENA