1. INTRODUCCIÓN

Las nuevas generaciones son cada vez más exigentes, las tecnologías de la información y las comunicaciones están presentes en todos los ámbitos de la vida diaria.

Según varios autores [1] [2] en los últimos años se ha despertado un gran interés por el diseño de materiales educativos.

El uso de herramientas computacionales educativas ha generado nuevas formas de impartir el conocimiento, creando nuevos contextos de aprendizaje en donde los estudiantes pueden aplicar los conocimientos adquiridos en entornos tradicionales (aulas, laboratorios), y lo que propicia mayor interactividad, creatividad de los canales para el aprendizaje.

Software Educativo por su rol que cumple en el proceso de aprendizaje, es considerado como parte del material educativo, enmarcándose como Material Educativo Computarizado (MEC). [3]

1.1 ANTECEDENTES

La programación orientada a objetos (POO) es un conjunto importante de técnicas que pueden utilizarse para hacer el desarrollo de programas más eficiente, a la par que mejora su fiabilidad. En la POO los objetos son los elementos principales. Sin embargo la simple comprensión de lo que es un objeto, o bien el uso de objetos en un programa, no significa que se está programando en un modo orientado a objetos. Lo que cuenta es el sistema en el cual los objetos se interconectan e interactúan entre sí. [4]

La enseñanza de la POO es una actividad no fácil de realizar de manera efectiva. La aparición del enfoque de objetos y el surgimiento de nuevos lenguajes de programación, herramientas y tecnologías, han introducido nuevos factores de dificultad a esta labor docente. El proceso de enseñanza-aprendizaje comprende un conjunto de actividades que van desde la explicación teórica, partiendo desde conceptos específicos, hasta los más generales, llegando a la realización de ejercicios sencillos que buscan representar la implementación de lo aprendido, basándose en ejemplos.
Se intenta comunicar los conceptos de una mejor manera. Los recursos que los maestros utilizan para la enseñanza son: Manuales de prácticas, clases y algunas herramientas de desarrollo con énfasis didáctico, como Bluej. [5]
El software educativo también ha jugado un papel muy importante en la enseñanza de diversas disciplinas, ya que tiene la finalidad específica de ser utilizado como medio para facilitar el proceso de aprendizaje, ofreciendo al usuario un ambiente propicio para la construcción del conocimiento.

Otros aspecto importante es considerar los conceptos que deben ser aprendidos; estudios realizados por Ben-Ari [6] han demostrado que una de principales causas que dificultan la comprensión de la POO es el aprendizaje de los conceptos clase y objeto, debido a que los estudiantes no los diferencian entre sí, así como el estado de un objeto, el envío de mensajes, el hecho de que un método puede cambiar el estado de un objeto, etc.

1.2 PLANTEAMIENTO DEL PROBLEMA

¿Cómo se puede apoyar para mejorar la efectividad en el aprendizaje de los conceptos de la programación orientada a objetos?

Dificultades en la trasmisión de los conocimientos, por parte del docente. Existe la necesidad de una comprensión adecuada en los temas relacionados con la Programación Orientada a Objetos (POO).

El profesor que imparte la asignatura de la POO, necesita recursos didácticos adicionales a los que se emplean en clases, principalmente porque el tiempo disponible para interactuar con los alumnos se limita al horario de clases.

Para una mayor efectividad en el aprendizaje se requiere que el alumno pueda reforzar los conocimientos en otro horario, tal vez en casa. Esto es debido a la dificultad para aprender los conceptos de la POO. Para ello se le debe de proveer de algún medio de apoyo.

Existen herramientas de software didácticas normalmente están apegadas a un lenguaje de programación especifico esto distrae al alumno del concepto básico que se desea enseñar, haciendo que se pierda en el mundo de la sintaxis del lenguaje.

La baja comprensión de la POO ha llevado a muchos alumnos a perder la motivación en la programación, y ha que exista un alto índice de reprobación.

Datos estadísticos nos muestran que la mayoría de los estudiantes de la POO, no conocen herramientas de apoyo para una mejor comprensión de los conceptos, el 60% de los estudiantes tienen dificultades para comprender los conceptos básicos de la POO (Clases, Objetos, Herencia, Abstracción). [Anexo 1]

1.3 ESTADO DEL ARTE

1.3.1 HERRAMIENTAS DE APOYO A LA ENSEÑANZA DE LA POO

Existe una serie de herramientas que ayudan a visualizar los objetos y su interacción de forma que se puedan entender mejor los conceptos abstractos de la Programación Orientada a Objetos. La elección de las herramientas analizadas se ha basado en la búsqueda de aplicaciones desarrolladas o en proceso de desarrollo que mantuviesen objetivos afines a la problemática planteada en esta investigación.

1.3.1.1 ALICE

Es un entorno de programación 3D bajo la filosofía de “arrastrar y soltar” objetos, que permite crear ambientes 3D, animaciones y vídeo para compartir en la web, con el propósito de entender cómo se crean y se relacionan estos objetos en un ambiente 3D.

- Arrastrar y soltar un objeto y hacer otros ajustes necesarios con el ratón.
- Fijar sus posiciones y hacer los ajustes manualmente de manera interactiva, así como la invocación de métodos sobre los objetos.
- Se puede escribir código para configurar puntos de vista de los objetos después de que el programa comienza a correr, pero antes de que empiece la animación.

En la figura 1, se muestra el entorno de trabajo de Alice, con un caso de estudio de una animación básica.

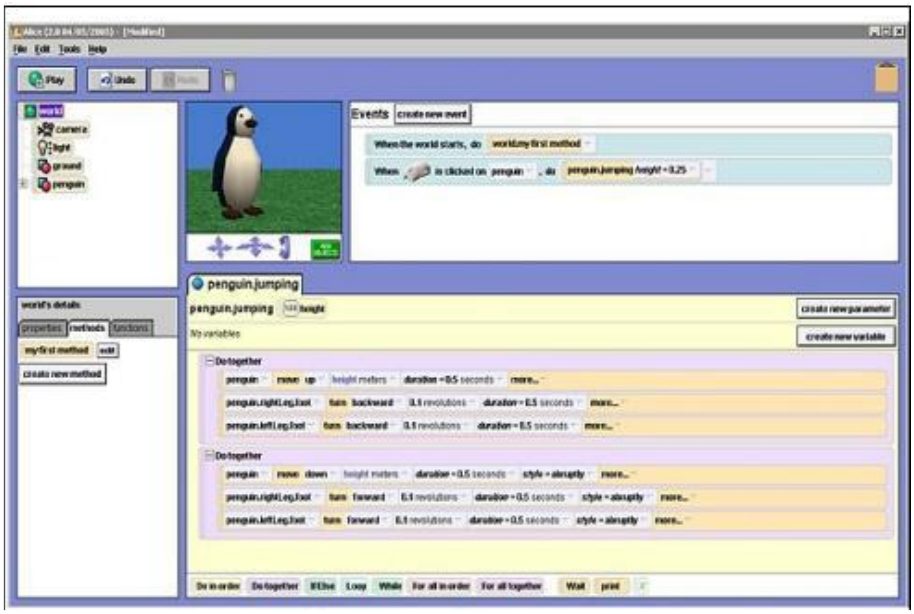


Figura 1: Entorno Integrado Alice.

Uno de los problemas de esta herramienta es la interacción con los objetos. Estos son creados y añadidos al mundo 3D, a su vez tienen una serie de propiedades que pueden ser inspeccionadas y cambiadas antes de que empiece la animación, sin embargo, una vez iniciada ésta, no se puede inspeccionar el estado de los objetos ni interactuar con ellos, lo que no le da mucha flexibilidad a la creación de animaciones.

1.3.1.2 JELIOT 3

Es un entorno para la visualización de programas Java. Los programas son ejecutados en una pantalla en forma de animación, donde se pueden seguir paso a paso el intercambio de mensajes, los atributos, métodos y ejecución de programa, lo que la convierte en una herramienta adecuada para la iniciación a la programación con Java.

Es un entorno adecuado para la comprensión de la sintaxis Java, mas no para aprender los conceptos básicos de la POO.

En la figura 2, se muestra el entorno de trabajo de Jeliot 3, durante el proceso de ejecución de un programa Java básico.

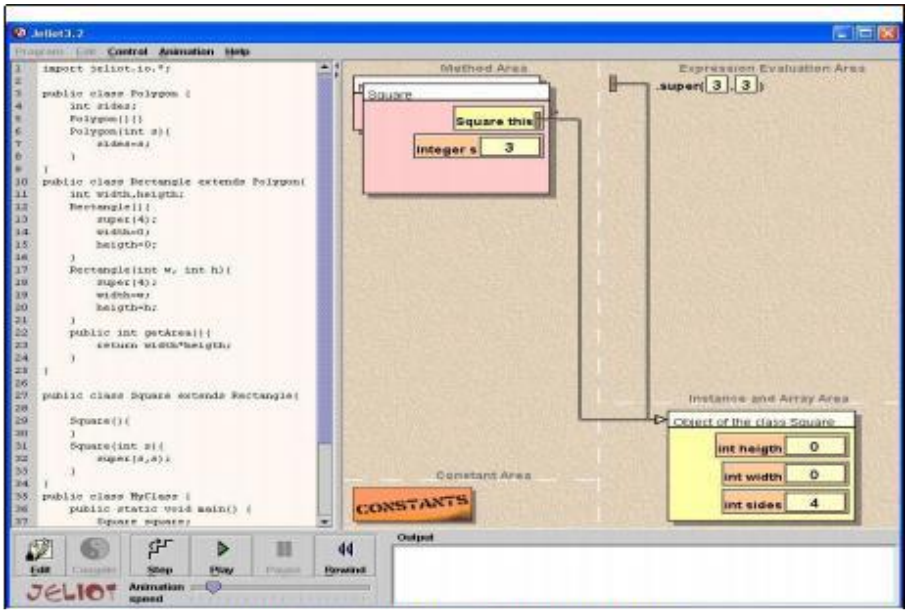


Figura 2: Entorno Integrado Jeliot 3

Una de las mayores deficiencias de esta herramienta es la interacción con los objetos, pues una vez compilado y ejecutado el código, el usuario no ejerce ningún tipo de interacción, únicamente observa el resultado e intenta comprender, lo que hace que el usuario no ejerce ningún tipo de interacción, únicamente observa el resultado e intenta comprender, lo que hace que el uso de la herramienta sea menos llamativo para los usuarios. Se trata de una herramienta que permite más una comprensión de la sintaxis Java, que de los principios de la POO. [7]

1.3.1.3 GREENFOOT

Es una combinación de un framework de trabajo para la creación de contenido multimedia en 2D y un entorno integrado de desarrollo (IDE) con las funciones de compilación y ejecución de programas en Java, explorador de proyectos, editor, etc. Muy útil para los ejercicios en Java donde se desee detallar la aparición de objetos y la comunicación entre éstos de una forma visual.

En la figura 3, se muestra el entorno de trabajo de Greenfoot, durante el proceso de ejecución del programa en Java robotWorld.

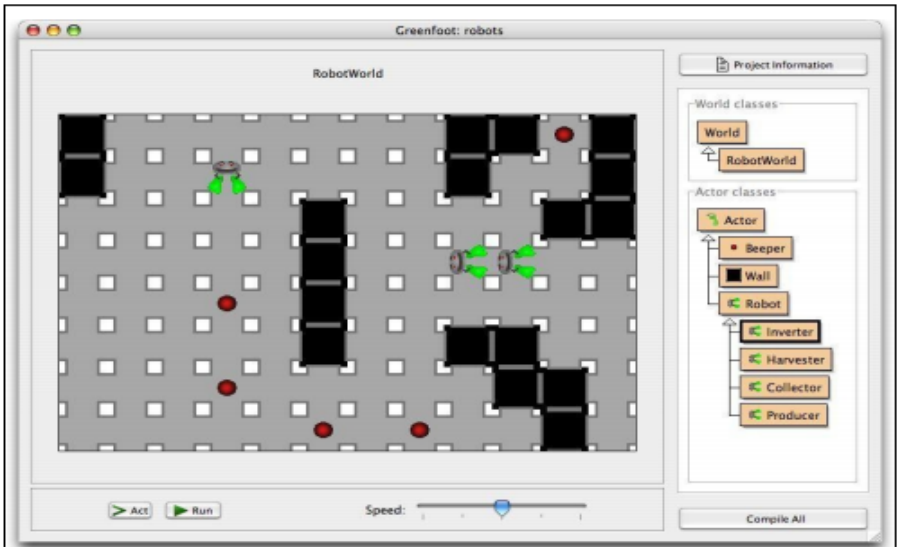


Figura 3: Entorno Integrado GreenFoot.

Además de diferenciar los conceptos de clase y objeto, la invocación a métodos y el acceso al estado y comportamiento de los objetos, ayuda a los usuarios a afianzar.

1.3.2 ANÁLISIS DE HERRAMIENTAS

Las herramientas explicadas anteriormente presentan especial énfasis en la visualización de los objetos y envío de mensajes entre ellos, no todas ofrecen las mismas posibilidades ni todas son igual de intuitivas.

Nótese que las tres herramientas analizadas introducen en mayor o menor grado los conceptos fundamentales de la POO, la creación de objetos, atributos y métodos y el envío de mensajes. Sin embargo, ninguna de ellas está en español y todas están apegadas a un lenguaje de programación. Nuestro software proporcionará los fundamentos de la POO (Objetos, clases, atributos, métodos), visualización de métodos y atributos en tiempo de ejecución, Y no estará ligado a ningún lenguaje de programación pero si anotaciones de UML.

2. JUSTIFICACIÓN

El uso de este software permitirá al alumno reforzar los conocimientos vistos en clases, El profesor podrá delegar parte del esfuerzo de la enseñanza al uso de esta herramienta, ya que el alumno podrá ocupar el tiempo extra clase necesario para aprender los conceptos de la POO, con el software le permitirá apoyarse y disipar las dudas que hayan quedado en clases, con esto evitará reprobar la materia e ir arrastrando la falta de conocimiento lo que conllevara a una deficiencia al emplear el paradigma.

El software al ser interactivo, el usuario se verá interesado en utilizarlo y podrá ser capaz de aprender en su casa o en el laboratorio de clases, esto se logrará porque el software podrá funcionar de manera normal en computadoras con hardware no tan potente como existen en la actualidad.

3. OBJETIVOS

3.1 OBJETIVO GENERAL

Desarrollar un software educativo multimedia que sirva como material de apoyo para mejorar el aprendizaje de la programación orientada a objetos.

3.2 OBJETIVOS ESPECÍFICOS.

- ✓ Definir cómo se pueden aplicar los conceptos de la programación orientada a objetos en un software educativo.
- ✓ Investigar los conceptos claves de la Programación Orientada Objetos (POO).
- ✓ Crear el software educativo que permitan al estudiante comprender mejor los conceptos estudiados en las materias programación orientada objetos.
- ✓ Probar la efectividad del software, aplicándolo a una muestra de estudiantes.

4. CARACTERIZACIÓN DEL ÁREA QUE SE PARTICIPÓ.

Nombre: Instituto Tecnológico de Tuxtla Gutiérrez

4.1 MISIÓN

Formar de manera integral profesionistas de excelencia en el campo de la ciencia y la tecnología con actitud emprendedora respeto al medio ambiente y apego a los valores éticos.

4.2 VISIÓN

Ser una institución de excelencia en la educación superior tecnológica del sureste comprometido con el desarrollo socioeconómico sustentable de la región.

4.3 UBICACIÓN GEOGRÁFICA DE LA INSTITUCIÓN

Dirección: carretera Panamericana Km.1080 Terán, 29050 Tuxtla Gutiérrez, Chiapas.

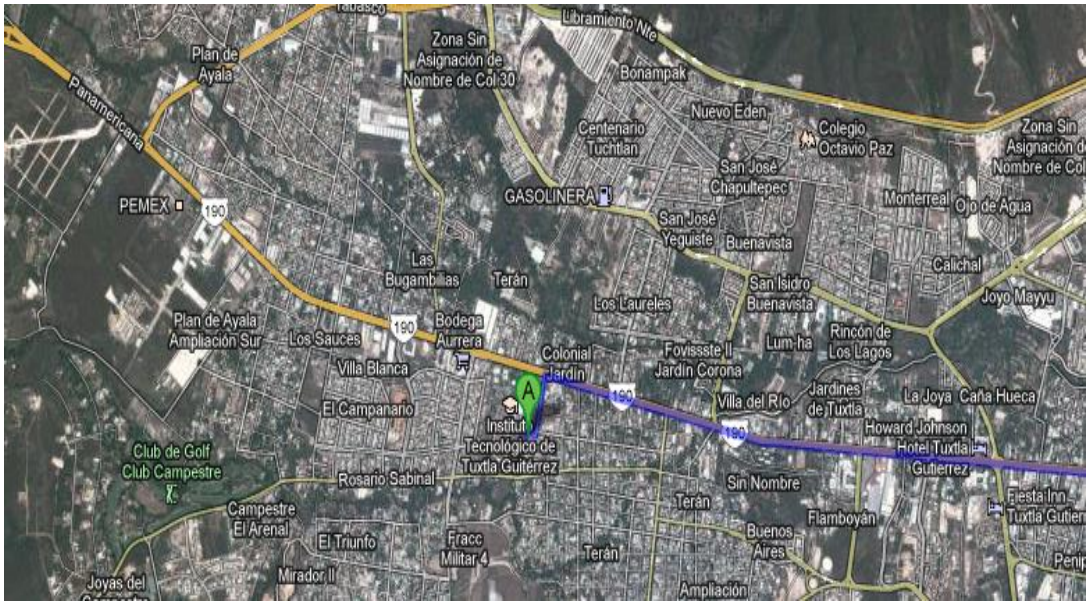


Figura 4: croquis Instituto Tecnológico de Tuxtla Gutiérrez, Chiapas

4.4 ORGANIGRAMA DE LA INSTITUCIÓN

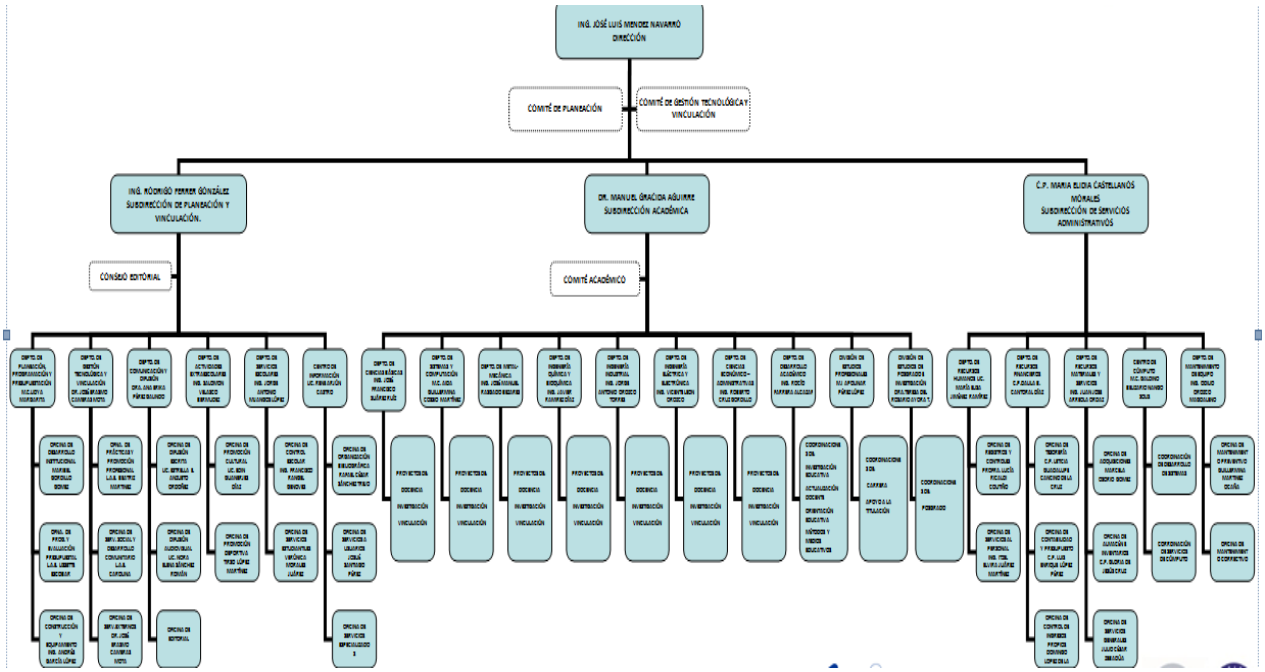
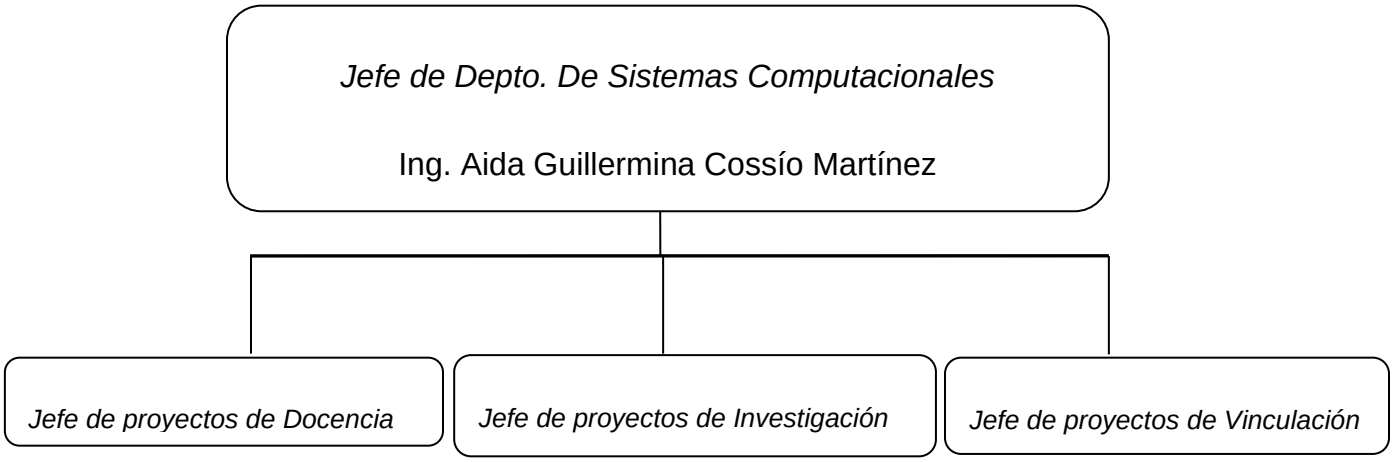


Figura 5: Organigrama de la Institución

4.5 DEPTO. INGENIERÍA EN SISTEMAS COMPUTACIONALES

4.5.1 ORGANIGRAMA



4.5.2 INFRAESTRUCTURA DEL DEPARTAMENTO DE INGENIERÍA EN SISTEMAS COMPUTACIONALES

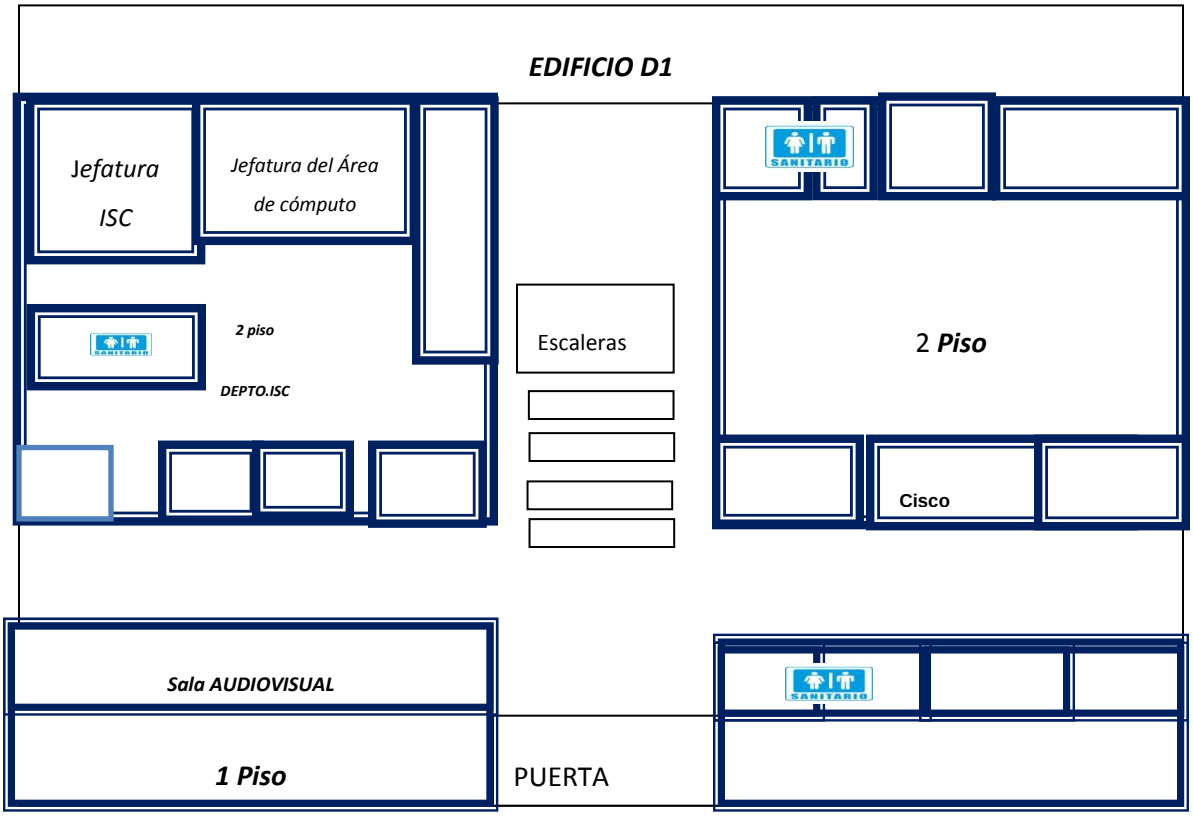


Figura 6: Infraestructura del Departamento de ISC

5. PROBLEMAS A RESOLVER PRIORIZÁNDOLOS

- Seleccionar los temas de la materia programación orientada a objetos de la retícula ISC-2010-224 del Instituto Tecnológico de Tuxtla Gutiérrez Chiapas.
- Seleccionar los objetos convenientes para dar a entender los conceptos.
- Diseñar el guion del software educativo, para cada tema se contara con una escena en los diferentes temas.
- Dar movimiento a los personajes.

6. ALCANCES Y LIMITACIONES

Con el software educativo pretendemos lograr el refuerzo del aprendizaje obtenido en el salón de clases con una estrategia diferente mediante el uso del computador y la interacción con el software.

Para desarrollo del Software Educativo, los temas de la POO que han sido considerados son Clase, Objetos, Abstracción y Herencia.

Cabe destacar, que esta herramienta será un complemento a la comprensión de la materia y que en ningún momento pretenderá ser sustituto en el aprendizaje guiado por parte del docente, es por ello que desea convertirse en un refuerzo y recurso libre y abierto a nuevos cambios y opiniones de las partes por las cuales será utilizado.

El software educativo será una herramienta que contara con escenarios atractivos, el estudiante podrá interactuar con la herramienta por medio de botones, contara con imágenes animadas y pequeños textos para una mejor comprensión de lo que se quiere explicar, será una *herramienta multimedia.*

7. MARCO TEÓRICO

7.1 MARCO TEORICO CONCEPTUAL

7.1.1 ADOBE FLASH

Los programas para animar son una herramienta insustituible en la producción de animación a gran escala permitiendo a un ahorro de tiempo de producción, así como de materiales de pintura y dibujo.

Adobe Flash Es una aplicación que utiliza gráficos vectoriales e imágenes ráster. Es un sistema muy empleado en la creación de páginas Web. Permite animar fotograma por fotograma, o interpolar movimientos o formas, a partir de dibujos clave. Es empleado para realizar animación limitada, full animación, ortoscopia, animación en sucio. [8]

¿Qué es Flash Player?

Flash Player es un programa creado por Adobe que permite visualizar contenido y aplicaciones interactivas en la Web. El contenido compatible con Flash Player (archivos SWF y FLV) está en toda la Web y, por lo tanto, Flash Player está instalado en el 98% de los equipos de escritorio todo el mundo. Visite el sitio web de Adobe para visualizar increíbles presentaciones creadas con Flash o para obtener más información sobre la creación de contenido SWF o FLV.

Flash Player dispone de información limitada sobre la capacidad de su equipo y permite que el contenido SWF o FLV guarde los datos en una ubicación específica del mismo. Sin embargo, como usuario de Flash Player, usted controla la privacidad y la configuración de almacenamiento de sitios web concretos o de todos los sitios web en general.

7.1.2 ACTIONSCRIPT

El ActionScript es el lenguaje de programación que ha utilizado Flash desde sus comienzos, y que por supuesto, emplea Flash CS5. A grandes rasgos, podemos decir que el ActionScript nos permitirá realizar con Flash CS5 todo lo que nos propongamos, ya que nos da el control absoluto de todo lo que rodea a una película Flash. Absolutamente de todo. [9]

7.1.3 ADOBE PHOTOSHOP

Es uno de los programas para animar para crear animaciones basadas en cuadros, modificando capas de imágenes, para crear la ilusión de movimiento y cambios de aspecto de las figuras. También puede crear imágenes para emplearlas en videos.
Cuando la edición se ha completado, se almacena como archivo GIF animado o como archivo PSD, y puede editarse en programas de video. También se puede importar archivos de video y secuencias de imágenes, para editarlas y retocarlas. Es posible crear animaciones basadas en líneas de tiempo, y exportarlas luego. [10]

7.1.4 UML

Desarrollo Orientado a Objetos con UML Xavier Ferré Grau, María Isabel Sánchez Segura El objetivo principal cuando se empezó a gestar UML era posibilitar el intercambio de modelos entre las distintas herramientas CASE orientadas a objetos del mercado. Para ello era necesario definir una notación y semántica común.

Hay que tener en cuenta que el estándar UML no define un proceso de desarrollo específico, tan solo se trata de una notación. En este curso se sigue el método propuesto por Craig Larman [Larman99] que se ajusta a un ciclo de vida iterativo e incremental dirigido por casos de uso.En la parte II de este texto se expone la notación y semántica básica de UML, en la parte III se presentan conceptos avanzados de la notación UML, mientras que en la parte IV se presenta el método de desarrollo orientado a objetos de Larman, que se sirve de los modelos de UML que se han visto anteriormente.

Modelos

Un modelo representa a un sistema software desde una perspectiva específica. Al igual que la planta y el alzado de una figura en dibujo técnico nos muestran la misma figura vista desde distintos ángulos, cada modelo nos permite fijarnos en un aspecto distinto del sistema. [14]

7.2 MARCO TEÓRICO ESPECÍFICO

7.2.1 PROGRAMACION ORIENTADA A OBJETOS (*POO)*

Los conceptos de la programación orientada a objetos tienen origen en simula 67, un lenguaje diseñado para hacer simulaciones, creado por Ole-Johan Dahi y Kristen Nygaard del centro de cómputo Noruego en Oslo.

Más tarde en Smaltak, Desarrollado en Simal en Xerox Parc (cuya primera versión fue escrita sobre Basic) pero diseñado para ser un sistema completamente dinámico en el cual los objetos se podrían crear y modificar “. [11]

7.2.2 SIGINIFICADO DE LA (POO).

La programación orientada objetos (POO). Es un paradigma de programación que usa los objetos en sus interacciones, para diseñar aplicaciones y programas informáticos.

7.2.3 BENEFICIOS Y USOS

Ventajas

- → Minimiza y neutraliza la extensión de código.
- → Facilita el mantenimiento de software.
- → Permite crear Sistemas más complejos.
- → Agiliza el Desarrollo de Software.

7.2.4 CONCEPTOS

7.2.4.1 OBJETO

Un objeto no es más que un conjunto de variables (o datos) y métodos (o funciones) relacionados entre sí. Los objetos en programación se usan para modelar objetos o entidades del mundo real. Un objeto es, por tanto, la representación en un programa de un concepto y contiene toda la información necesaria para abstraerlo: datos que describen sus atributos y operaciones que pueden realizarse sobre los mismos. La siguiente figura 7 [15] muestra una representación visual de un objeto.

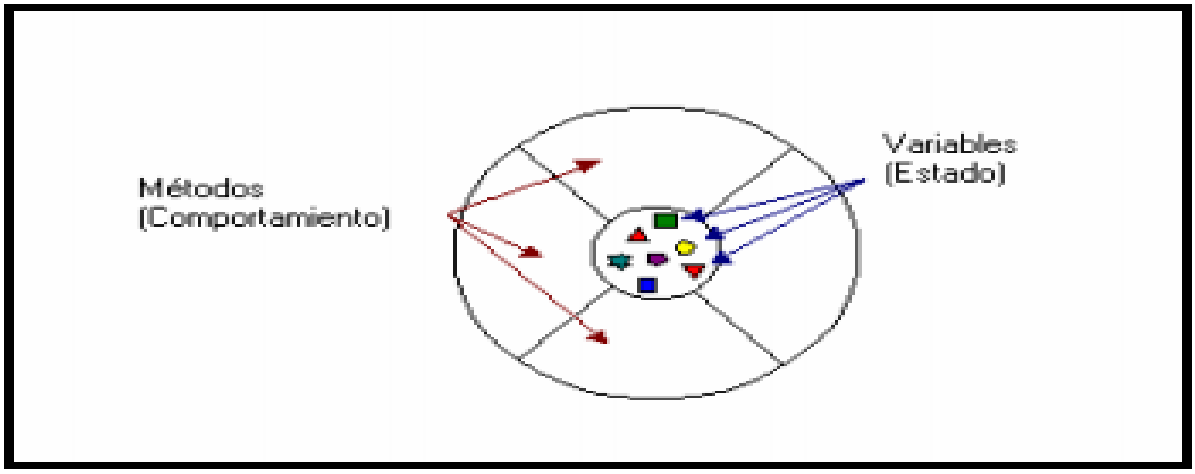


Figura 7: Representación visual

Los atributos del objeto (estado) y lo que el objeto puede hacer (comportamiento) están expresados por las variables y los métodos que componen el objeto respectivamente.

Por ejemplo un objeto que modelase una bicicleta en el mundo real tendría variables que indicara rían el estado actual de la bicicleta; su velocidad es de 20 km/h, su cadencia de pedaleo de 90. Estas variables se conocen formalmente como variables instancia porque contienen el estado de un objeto bicicleta particular y, en programación orientada a objetos, unos objetos particular se denomina instancia.

Además de estas variables, el objeto bicicleta podría tener métodos para frenar, cambiar la cadencia de pedaleo. (la bicicleta no tendría que tener un método para cambiar su velocidad pues esta función de la cadencia de pedaleo en la que está. Estos métodos se denominan formalmente métodos instancia. Ya que cambian el estado de una instancia u objeto bicicleta particular. La figura 8 [15] muestra una bicicleta modelada como un objeto.

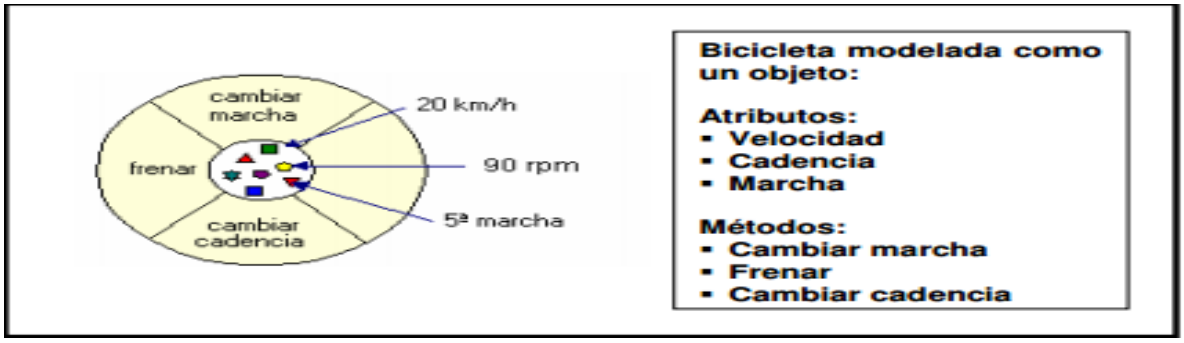


Figura 8: Representación visual

El diagrama del objeto bicicleta muestra las variables objeto en el núcleo o centro del objeto y los métodos rodeando el núcleo y protegiéndolo de otros objetos del programa.

Este hecho de empaquetar o proteger la variable miembro con el método miembro se denomina encapsulación. Este dibujo conceptual que muestra el núcleo de variables miembro del objeto protegido por una membrana protectora de métodos o función miembro es la representación ideal de un objeto y es el ideal que los programadores de objetos suelen buscar.

Sin embargo, debemos matizarlo. A menudo, por razones prácticas, es posible que un objeto desee exponer alguna de sus variables miembro, o proteger otras de sus propios métodos o función miembro. Por ejemplo, Java permite establecer 4 niveles de protección de las variables y de la función miembro para casos como éste. Los niveles de protección determinan qué objetos y clases pueden acceder a qué variables o a qué métodos.

De cualquier forma, el hecho de encapsular las variables y las funciones miembro relacionadas proporciona dos importantes beneficios a los programadores de aplicaciones:

- **Capacidad de crear módulos**: El código fuente de un objeto puede escribirse y mantenerse independiente del código fuente del resto de los objetos. De esta forma, un objeto puede pasarse fácilmente de una parte a otra del programa. Podemos dejar nuestra bicicleta a un amigo, y ésta seguirá funcionando.
- **Protección de información:** Un objeto tendrá una interfaz pública Perfectamente definida que otros objetos podrán usar para comunicarse con él. De esta forma, los objetos pueden mantener información privada y pueden cambiar el modo de operar de sus funciones miembros sin que esto afecte a otros objetos que usen estas funciones miembro. Es decir, no necesitamos entender cómo funciona el mecanismo de cambio de marcha para hacer uso de él.

7.2.4.2 CLASE

Normalmente en el mundo real existen varios objetos de un mismo tipo, o como diremos enseguida, de una misma clase. Por ejemplo, mi bicicleta es una de las muchas bicicletas que existen en el mundo.

Usando la terminología de la programación orientada a objetos, diremos que mi bicicleta es una instancia de la clase de objetos conocida como bicicletas.

Todas las bicicletas tienen algunos estados o atributos (color, marcha actual**,** cadencia actual, dos ruedas) y algunos métodos (cambiar de marcha, frenar) en común. Sin embargo, el estado particular de cada bicicleta es independiente del estado de las demás bicicletas. La particularización de estos atributos puede ser diferente. Es decir, una bicicleta podrá ser azul, y otra roja, pero ambas tienen en común el hecho de tener una variable “color”. De este modo podemos definir una plantilla de variables y métodos para todas las bicicletas. Las plantillas para crear

objetos son denominadas clases. **Una clase es una plantilla que define las variables y los métodos que son comunes para todos los objetos de un cierto tipo.**

En nuestro ejemplo, la clase bicicleta definiría variables miembro común a todas las bicicletas, como la marcha actual, la cadencia actual, etc. Esta clase también debe declarar e implementar los métodos o función miembro que permiten al ciclista cambiar de marcha, frenar, y cambiar la cadencia de pedaleo, como se muestra en la figura 9[15]**.**

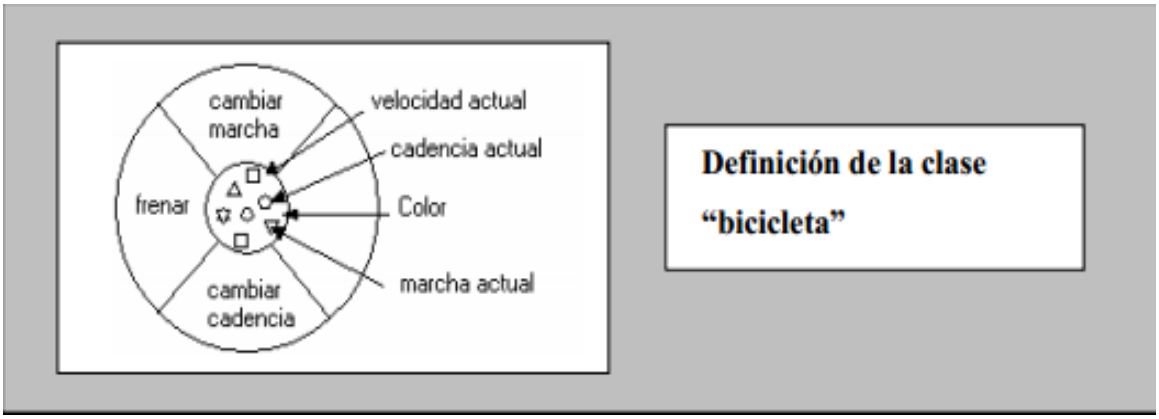


Figura 9: Representación visual

Después de haber creado la clase bicicleta, podemos crear cualquier número de objetos bicicleta a partir de la clase. Cuando creamos una instancia de una clase, el sistema reserva suficiente memoria para el objeto con todas sus variables miembro. Cada instancia tiene su propia copia de las variables miembro definidas en la clase.

7.2.4.3 HERENCIA

Una vez que hemos visto el concepto de clase y el de objeto, estamos en condiciones de introducir otra de las características básicas de la programación orientada a objetos: el uso de la herencia.

El mecanismo de herencia permite definir nuevas clases partiendo de otras ya existentes. Las clases que derivan de otras heredan automáticamente todo su

comportamiento, pero además pueden introducir características particulares propias que las diferencian.

Como hemos visto, los objetos se definen a partir de clases. Con el mero hecho de conocer a qué clase pertenece un objeto, ya se sabe bastante sobre él. Puede que no sepamos lo que es la “Espada”, pero si nos dicen que es una bicicleta, ya sabremos que tiene dos ruedas, manillar, pedales...

La programación orientada a objetos va más allá, permitiéndonos definir clases a partir de otras clases ya construidas. Por ejemplo, las bicicletas de montaña, las de carretera son todos, en definitiva, bicicletas. En términos de programación orientada a objetos, son subclases o clases derivadas de la clase bicicleta. Análogamente, la clase bicicleta es la clase base o superclase de las bicicletas de montaña, las de carretera. Esta relación se muestra en la figura 10[15].

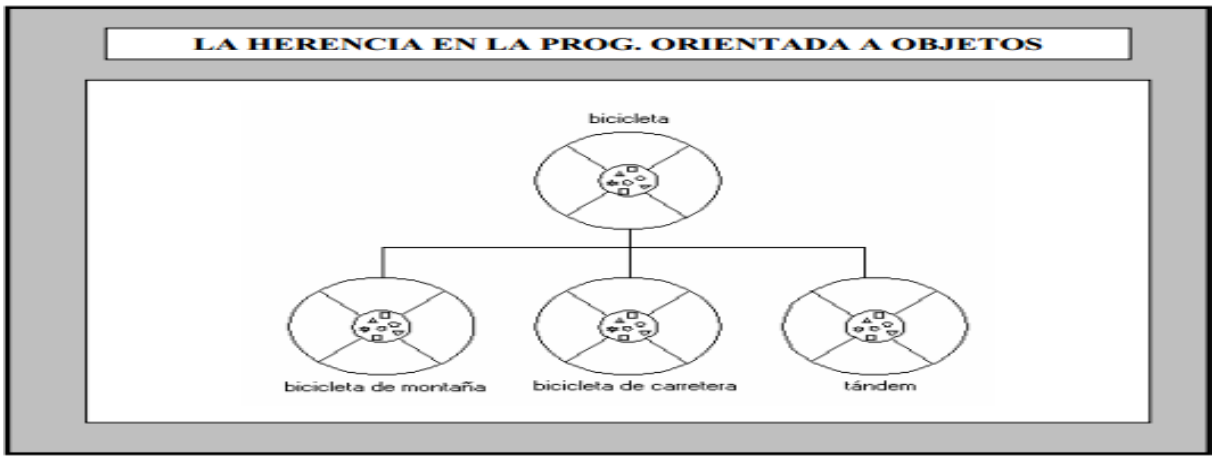


Figura 10: Representación visual Herencia

Cada subclase hereda los estados (en forma de declaración de variables) de la superclase de la cual deriva. Las bicicletas de montaña, las de carretera y los tándems comparten algunos estados: cadencia, velocidad... Además, cada subclase hereda los métodos de su superclase.

Las bicicletas de montaña, las de carretera y los tándems comparten algunos comportamientos: frenar y cambiar la cadencia de pedaleo. Sin embargo, las clases

derivadas no se encuentran limitadas por los estados y comportamientos que heredan de su superclase. Muy al contrario, estas subclases pueden añadir variables y métodos a aquellas que han heredado. Los tándems tienen dos asientos y dos manillares; algunas bicicletas de montaña tienen una catalina adicional con un conjunto de marchas con relaciones de transmisión mucho más cortas.

Las clases derivadas pueden incluso sobrescribir los métodos heredados y proporcionar implementaciones más especializadas para esos métodos.

Por ejemplo, si nuestra bicicleta de montaña tuviera una catalina extra, podríamos sobrescribir el método “CambiarDeMarcha” para poder usar esas nuevas marchas.

Además, no estamos limitados a un único nivel de herencia. El árbol de herencias o jerarquía de clases puede ser tan extenso como necesitemos. Los métodos y las variables miembro se heredarán hacia abajo a través de todos los niveles de la jerarquía. Normalmente, cuanto más abajo está una clase en la jerarquía de clases, más especializado es su comportamiento. En nuestro ejemplo, podríamos hacer que la clase bicicleta derivase de una superclase de vehículos.

La herencia es una herramienta clave para abordar la resolución de un problema de forma organizada, pues permite definir una relación jerárquica entre todos los conceptos que se están manejando. Es posible emplear esta técnica para descomponer un problema de cierta magnitud en un conjunto de problemas subordinados a él. La resolución del problema original se consigue cuando se han resuelto cada uno de los problemas subordinados, que a su vez pueden contener otros.

Por consiguiente, la capacidad de descomponer un problema o concepto en un conjunto de objetos relacionados entre sí cuyo comportamiento es fácilmente identificable puede ser extraordinariamente útil para el desarrollo de programas informáticos.

La herencia proporciona las siguientes ventajas:

- Las clases derivadas o subclases proporcionan comportamientos especializados a partir de los elementos comunes que hereda de la clase base. A través del mecanismo de herencia los programadores pueden reutilizar el código de la superclase tantas veces como sea necesario.
- Los programadores pueden implementar las llamadas superclases abstractas, que definen comportamientos genéricos. Las clases abstractas definen e implementan parcialmente comportamientos, pero gran parte de estos comportamientos no se definen ni se implementan totalmente. De esta forma, otros programadores pueden hacer uso de estas superclases detallando esos comportamientos con subclases especializadas.
- El propósito de una clase abstracta es servir de modelo base para la creación de otras clases derivadas, pero cuya implantación depende de las características particulares de cada una de ellas. Un ejemplo de clase abstracta podría ser en nuestro caso la clase vehículos. Esta clase sería una clase base genérica, a partir de la cual podríamos ir creando todo tipo de clases derivadas.

7.2.4.4 MENSAJES Y MÉTODOS

La Programación Orientada a Objetos no solo tiene en consideración los Objetos, sino también sus interrelaciones.

- **Mensaje**

  Los objetos interactúan enviándose mensajes unos a otro. Tras la recepción de un mensaje el objeto actuar. La acción puede ser el envió de otros mensajes, el cambio de su estado o la ejecución de cualquier otra tarea que se requiera que haga el Objeto.
- **Método**

  Un método se implementa en una Clase, y determina como tiene que actuar el objeto cuando recibe un mensaje.

Cuando un objeto A necesita que el Objeto B ejecute alguno de sus métodos, el objeto A le manda un mensaje al objeto B. [12]

7.2.5 ASPECTOS PEDAGÓGICOS

7.2.5.1 SOFTWARE EDUCATIVO. .

La asignación del término educativo a los programas para computadora, se debe a que estos son elaborados con un sólo propósito y con características propias que determinan su carácter educacional.

Investigadores de esta nueva disciplina, definen como “cualquier programa computacional que cuyas características estructurales y funcionales le permiten servir de apoyo a la enseñanza, el aprendizaje y la administración educacional” (Sánchez, 1995). “las expresiones de software educativo, programas educacionales y programas didácticos como sinónimos para designar genéricamente todo tipo de programas para computador creados con la finalidad específica de ser utilizado como medio didáctico”, esta última definición involucra a todo los programas que son diseñados con el fin de apoyar la labor del profesor, como es el caso de los programas conductistas para la

- Enseñanza Asistida por Computador (E.O.A.), y los programas de Enseñanza
- Inteligente Asistida por Computador (E.I.A.O.). (Márquez, 1995).

Software Educativo por su rol que cumple en el proceso de aprendizaje, es considerado como parte del material educativo, enmarcándose como Material Educativo Computarizado (MEC). (Galvis, 1994).

Características del software educativo.

En el mercado existen diversos programas que son considerados como “software educativo”, pero que requieren ser diferenciados por sus características propias

considerando que estos deben cumplir con fines educativos. Siendo las principales las siguientes:

- El software educativo es concebido con un propósito específico: apoyar la labor del profesor en el proceso de aprendizaje de los estudiantes.
- Además de sus características computacionales, estas deben contener elementos metodológicos que orienten el proceso de aprendizaje.
- Son programas elaborados para ser empleados por computadores, generando ambientes interactivos que posibilitan la comunicación con el estudiante.
- La facilidad de uso, es una condición básica para su empleo por parte de los estudiantes, debiendo ser mínimos los conocimientos informáticos para su utilización.
- Debe ser un agente de motivación para que el alumno, pueda interesarse en este tipo de material educativo e involucrarlo
- Poseer sistemas de retroalimentación y evaluación que informen sobre los avances en la ejecución y los logros de los objetivos educacionales que persiguen.

Componentes del software educativo.

Estos como todo material que tienen una finalidad educativa, están conformado por diversos componentes, siendo aquellos que realizan el proceso de comunicación entre la computadora y el usuario (interfaz), los que contienen la información y los procesos metodológicos (pedagógico) y los que orientan las secuencias y acciones del sistemas (computacional).

A) Componente de comunicación o interfaz: Es aquel que posibilita la interacción entre los usuarios y el programa, en el cual intervienen los tipos de mensajes entendibles por el usuario y por el programa así como los dispositivos de entrada y salida de datos y las zonas de comunicación disponibles para el intercambio de mensajes, comprendiendo
Dos niveles:

- *Programa-usuario*, esta relación posibilita la transmisión de la información desde la computadora al usuario, a través de diversos periféricos como la pantalla, principal componente que presenta la información al usuario, así como las impresoras.
- *Usuario-programa*, relación que permite la comunicación del usuario con la computadora. En este proceso se involucra el empleo principalmente del teclado, así como de los apuntadores (mouse, lápiz óptico), para la introducción de información, comandos y respuestas. Así mismo se puede considerar el empleo de otros periféricos como: micrófonos, pantallas táctiles, lectores ópticos.

B) Componente pedagógico o instruccional: Es el que determina los objetivos de aprendizaje que se lograrán al finalizar el empleo del software, los contenidos a desarrollar con el programa en función a los objetivos educacionales, las secuencias de la instrucción, los tipos de aprendizajes que se quieren lograr, sistemas de evaluación que se deben considerar para determinar los logros y los sistemas de motivación extrínseca e intrínseca que se deben introducir.

C) Componente computacional o técnico*:* Que permite establece la estructura lógica para la interacción para que el software cumpla con las acciones requeridas por el usuario, así como ofrecer un ambiente al estudiante para que pueda aprender lo deseado y servir de entorno. A la estructura lógica del programa se liga íntimamente la estructura de datos, que organiza la información necesaria para que el software pueda cumplir con sus objetivos instrucionales. El algoritmo que se emplee determinará el tipo de ambiente de aprendizaje, y la interacción del programa. [13]

8. PROCEDIMIENTOS Y DESCRIPCIÓN DE LAS ACTIVIDADES REALIZADAS.

Primeramente el software educativo cuenta con la escena de bienvenida donde se encuentran el título del proyecto que es (Software Educativo para el aprendizaje de la programación orientada a objetos).

También contiene un menú interactivo donde se encuentran los temas como (Clase, clase y Objeto, Abstracción y Herencia).

Realizamos búsquedas referentes a software educativo para el aprendizaje de la *POO.* Recopilamos información para tener en cuenta con lo que ya existe teniendo una ayuda para la realización del software, también buscamos sobre los fundamentos de los conceptos de la *POO.*

1. **Diseño del prototipo.**
   Diseñamos el software educativo para tener una imagen clara que queríamos hacer y hasta donde vamos a llegar. Tomando en cuenta las necesidades del cliente.
2. **Crear el diseño en Flash (imágenes usar).**
   Diseñamos las imágenes a utilizar y los escenarios que el software va contener.
3. **Desarrollar la primer Modulo del software.**
   Usamos nuestros Diseños y comenzamos a crear la primera parte del software, la de clases y Objetos, creamos todos nuestros objetos a usar y las clases.

   Clase: en esta escena aparece la definición del concepto y nos enlaza a menú de clase donde se encuentra los ejemplos de automóvil, persona.

   - Escenario automóvil: en esta sección se encuentra un rompe-cabeza e instrucciones donde se muestra que es la clase y el usuario debe de armar esa clase automóvil.

- Escenario persona: en esta sección se encuentra un rompe-cabeza e instrucciones donde se muestra que es la clase y el usuario debe de armar esa clase persona

Clase y Objetos: en esta escena aparece la definición del concepto “objeto” y de ahí nos enlaza a menú de clase y objetos, donde se encuentra automóvil, persona y bicicleta.

- Escenario automóvil: en esta sección el usuario interactuar con el objeto automóvil donde el usuario puede cambiarle de atributos al objeto automóvil y también selecciona el método que tiene dicho objeto.
- Escenario persona: en esta sección el usuario interactuar con el objeto persona donde el usuario puede cambiarle de atributos al objeto persona y también selecciona el método que tiene dicho objeto.
- Escenario bicicleta: en esta sección el usuario interactuar con el objeto bicicleta donde el usuario puede cambiarle de atributos al objeto bicicleta y también selecciona el método que tiene dicho objeto.

4. **Hacer las pruebas necesarias para el primer modulo.**
   En esta parte de la actividad llevamos a cabo las pruebas necesarias del primer modulo ***clase y objetos.***

5. **Desarrollar el segundo modulo del software.**

   Usamos nuestros Diseños y comenzamos a crear el segundo modulo del software, la de **Abstracción** creamos todos nuestros objetos a interactuar.

   Abstracción: En esta escena aparece la definición del concepto “abstracción” y de ahí nos enlaza al escenario donde se muestra el objeto principal que es un gato y diferentes personajes abstraen al gato de manera diferente, al darle clic al personaje que está observando el gato nos muestra de que manera abstrae al gato.

6. **Desarrollar el tercer modulo**

   En este modulo de la actividad llevamos a cabo las pruebas necesarias del tercer modulo ***Herencia.***

   Herencia: en esta escena aparece la definición del concepto “herencia” y de ahí nos enlaza a menú herencia donde se encuentra herencia simple y herencia múltiple.

   Escena de herencia simple: aquí se muestra un ejemplo de herencia donde se encuentra la clase padre que es trasporte y las clases hija como automóvil, avión, motocicleta. Donde el usuario observara cuales son los atributos y métodos que hereda las clases hijas.

- Escena de herencia múltiple: aquí se muestra un ejemplo de herencia donde se encuentra la clase padre que es persona y las clases hija como profesor, estudiante donde también estas clase se convierten en clase padre ya que de esas clase se hereda la clase investigador que es clase hija. Donde el usuario observara cuales son los atributos y métodos que hereda las clases hijas.

7. **Probar el software Completo.**

   Hicimos las pruebas con estudiantes que llevaron la materia Programación Orientada a Objetos en la cual realizamos unas encuestas [anexo 2] antes de Probar el software y otra después en la cual probamos el software completo con sus 4 módulos :***Clases, Objetos, Abstracción, Herencia***

9. RESULTADOS, PLANOS, PROTOTIPO Y PROGRAMAS

9.1 MÉTODOS APLICADOS

El propósito de esta fase es crear una vista general de los objetivos del proyecto, establecer que es lo qué se debe cubrir con el desarrollo del sistema. En la primera instancia se inicia con una descripción de la metodología utilizada y de las necesidades educativas que deben ser atendidas.

La creación del software educativo para el aprendizaje de los conceptos de la programación Orientada a Objetos, se desarrolló empleando. El Modelo Incremental combina elementos del MLS con la filosofía interactiva de construcción de prototipos.

En una visión genérica, el proceso se divide en 4 partes: Análisis, Diseño, Código y Prueba. Sin embargo, para la producción del Software, se usa el principio de trabajo en cadena o “Pipeline”, utilizado en muchas otras formas de programación. Con esto se mantiene al cliente en constante contacto con los resultados obtenidos en cada incremento.

9.2 CARACTERISTICAS ESPERADAS

Las especificaciones de los requisitos funcionales contienen los aspectos del contenido del comportamiento del software que es requerido por el usuario.

- Apoyo la labor del profesor reduciendo su carga de la materia POO.
- El software debe tener animaciones y mecanismos de interacción.
- El sistema de mostrar ejemplos sencillos audiovisuales con animaciones textos.
- La aplicación debe ofrecer al usuario facilidades en cuanto a su uso y manejo.
- Permitir al usuario una rápida asimilación de los conceptos.
- Debe contar con un manual de usuario.

9.3 DIAGRAMA DE CASO DE USO

El modelo caso de uso permite establecer las condiciones y posibilidades que debe cumplir el software educativo, es decir los casos de usos modelos los requisitos funcionales.

9.3.1 ACTORES DEL SISTEMA

Los actores del sistema representan cualquier elemento que intercambian información con el sistema POO y que esta fuera.

Actores	Descripción
	Representa a la persona que va interactuar con el software educativo, con la finalidad de adquirir conocimientos.

Figuran 11: Actores del sistema

9.3.2 CASO DE USO GENERAL

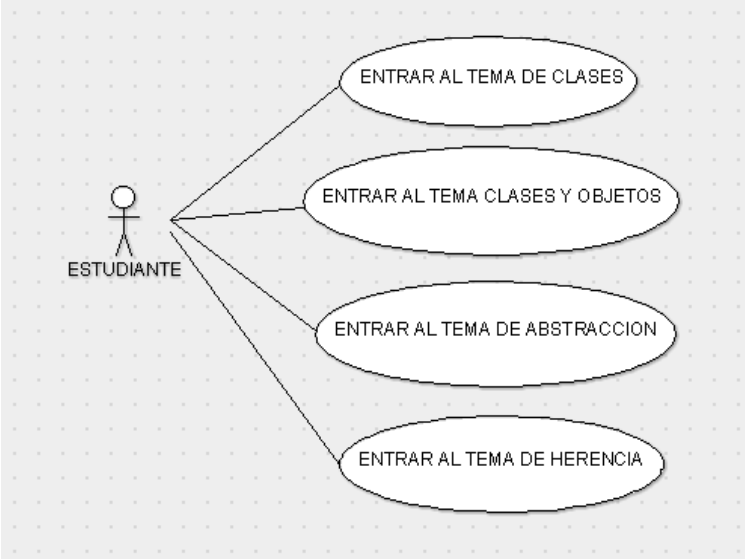


Figura 12: Diagrama de Caso de Uso

Plantillas de los Casos de Usos

Nombre de Caso de Uso: ***CLASES***	
Actor: *ESTUDIANTE*	**Descripción**: Que el estudiante pueda entrar al tema de clases e interactuar con él para comprender mejor el concepto de clases..
Precondiciones: Activa la opción **Tema de clase**	
Flujo Normal	
Usuario del sistema 1.Entra al sistema 2. Selecciona el tema clases. 3.clic en entrar 4. Selección *automóvil o Persona* 5. Selecciona actualizar o Regresar	**Respuesta Sistema** 1.Muestra la pantalla principal 2. Muestra una pantalla inicial con el concepto clase 3. entrar al tema clases. 4. comienza a interactuar con el usuario. 5. Actualiza o Regresa al menú de clases. .

.

Nombre de Caso de Uso: ***Objetos***	
Actor: *ESTUDIANTE*	**Descripción**: Que el estudiante pueda entrar al tema de Objetos. Al darle clic a la segunda sección del sistema. Se explicara los conceptos de objetos
Precondiciones: Activa la opción **Tema de Objetos**	
Flujo Normal	
Usuario del sistema 1.Entra al sistema 2. Selecciona el tema Objetos 3.clic en entrar 4. Selección *El objeto deseado* 5. Selecciona construir el objeto.	**Respuesta Sistema** 1.Muestra la pantalla principal 2. Muestra una pantalla inicial con el concepto Objetos 3. entrar al tema Objetos 4. comienza a interactuar con el usuario.

.

Nombre de Caso de Uso: ***ABSTRACCION***	
Actor: *ESTUDIANTE*	**Descripción**: Que el estudiante pueda entrar al tema de *abstracción,* Al darle clic a la tercera sección del sistema. Se explicara los conceptos de Abstracción
Precondiciones: Activa la opción **Tema de Abstracción**	
Flujo Normal	
Usuario del sistema 1.Entra al sistema 2. Selecciona el tema Abstracción. 3.clic en entrar 4. Selecciona al personaje deseado 5. Selecciona Regresar	**Respuesta Sistema** 1.Muestra la pantalla principal 2. Muestra una pantalla inicial con el concepto Abstracción 3. entrar al tema clases. 4. comienza a interactuar con el usuario 5. Regresar al menú principal. .

Nombre de Caso de Uso: Herencia	
Actor: *ESTUDIANTE*	**Descripción**: Que el estudiante pueda entrar al tema de *Herencia,* Al darle clic a la cuarta sección del sistema. Se explicara los conceptos de Herencia
Precondiciones: Activa la opción **Tema Herencia**	
Flujo Normal	
Usuario del sistema 1.Entra al sistema 2. Selecciona el tema Herencia. 3.clic en entrar 4. Selecciona herencia simple o herencia múltiple. 5. Selecciona Regresar	**Respuesta Sistema** 1.Muestra la pantalla principal 2. Muestra una pantalla inicial con el concepto Herencia 3. entrar al tema herencia. 4. comienza a interactuar con el usuario 5. Regresar al menú principal. .

9.4 DIAGRAMA DE ESTADO

En este diagramas de estado muestran el conjunto de estados por los cuales pasa un objeto durante su vida del software Educativo en respuesta a eventos.

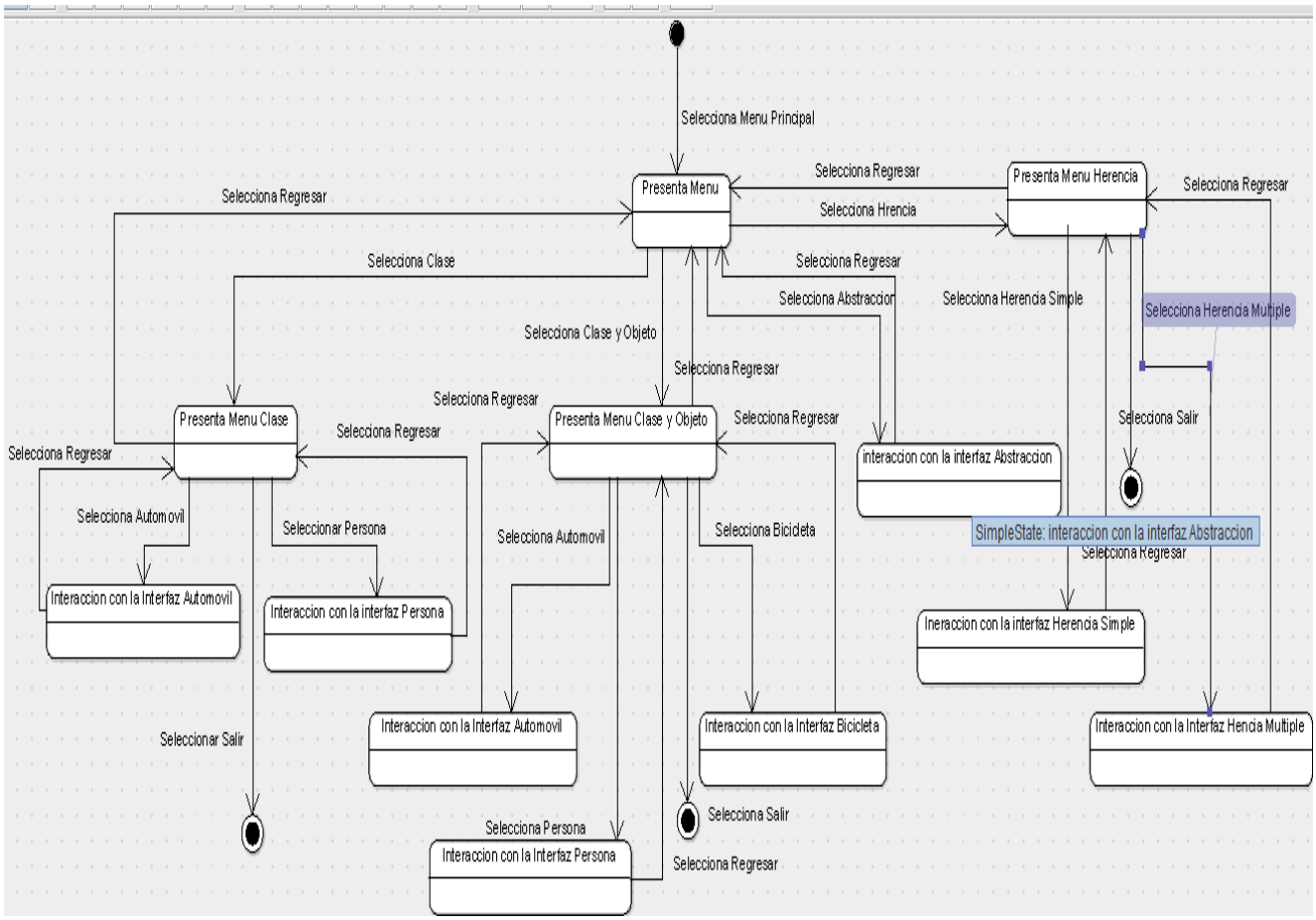


Figura 13: Diagrama de Estado

9.5 PROTOTIPO

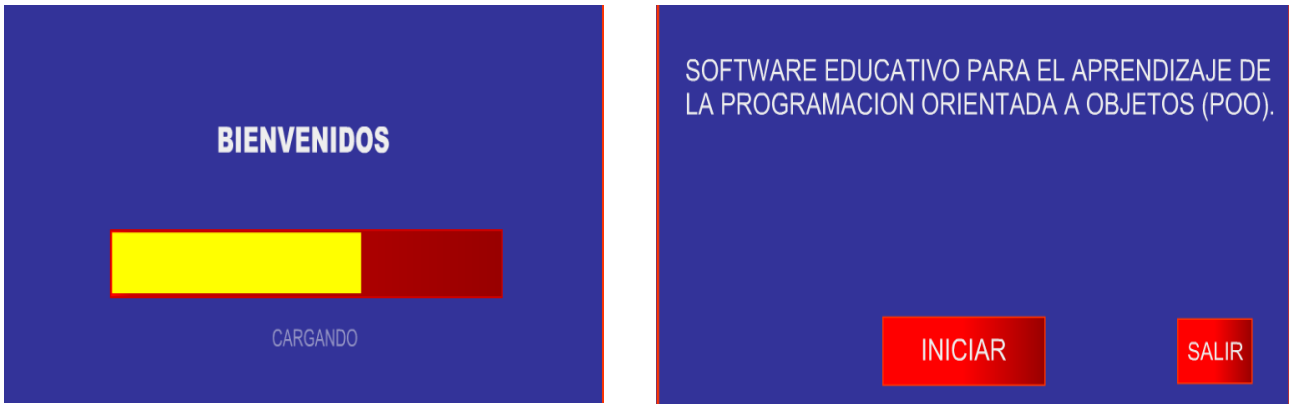


Figura 14-Pantallas de Bienvenida

Figura 15-Menu Principal

Menú Principal: Cada manzana tendrá el nombre del tema: CLASES Y OBJETOS-INTERACCION ENTRE OBJETOS-HERENCIA dándole clic se expandirá y nos mandara a la escena deseada.

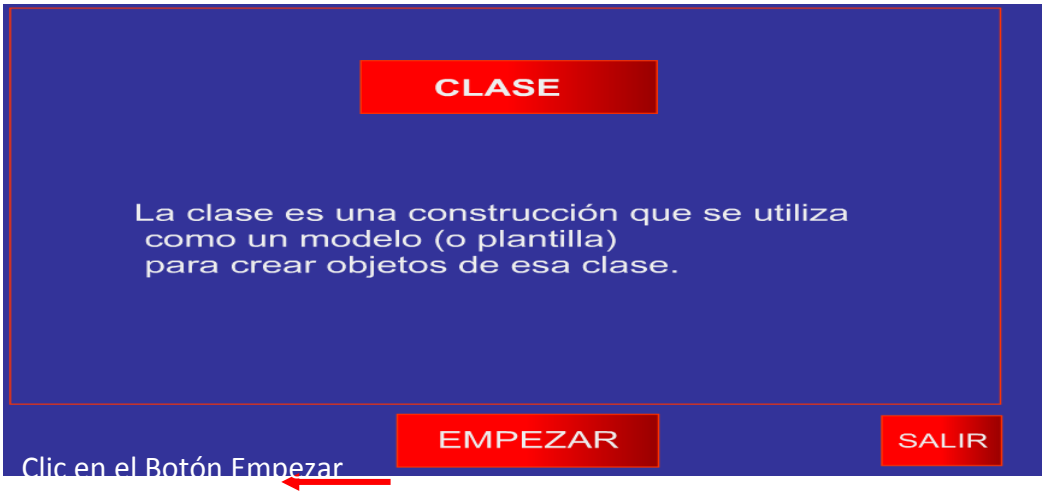


Figura16-Concepto Clase

1. Clic en el botón Empezar que nos enviara al ***menú clases.***

Figura 17-Menu Clases

2. En esta pantalla se mostrara EL MENU CLASE que serán botones que al darle clic a cada uno ya sea automóvil o persona mandara a otra escena.

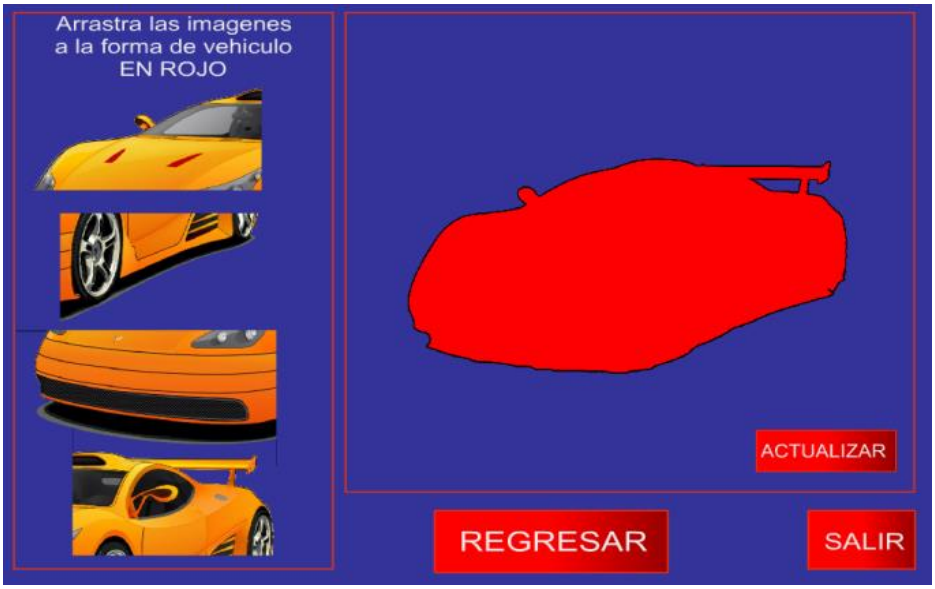


Figura 18-Clase Automóvil

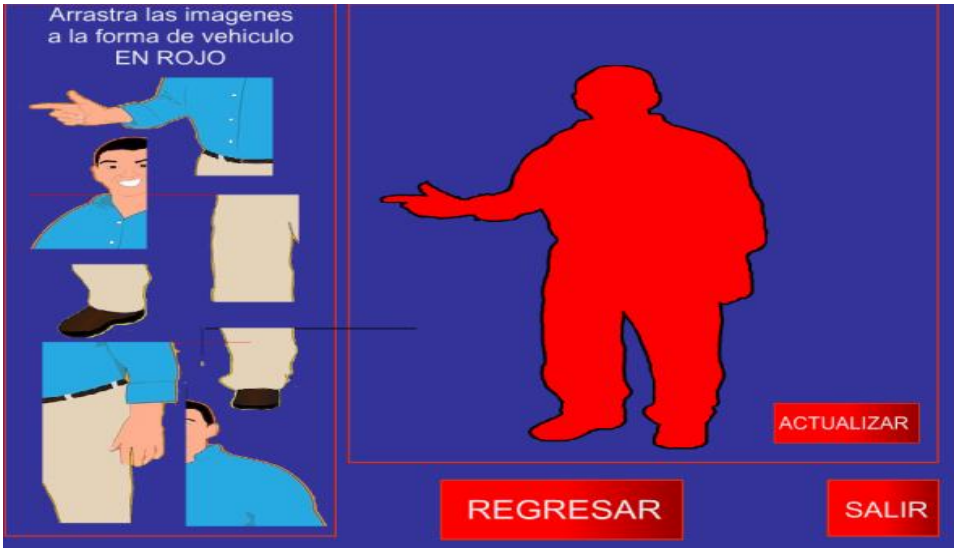


Figura 19-Clase Automóvil

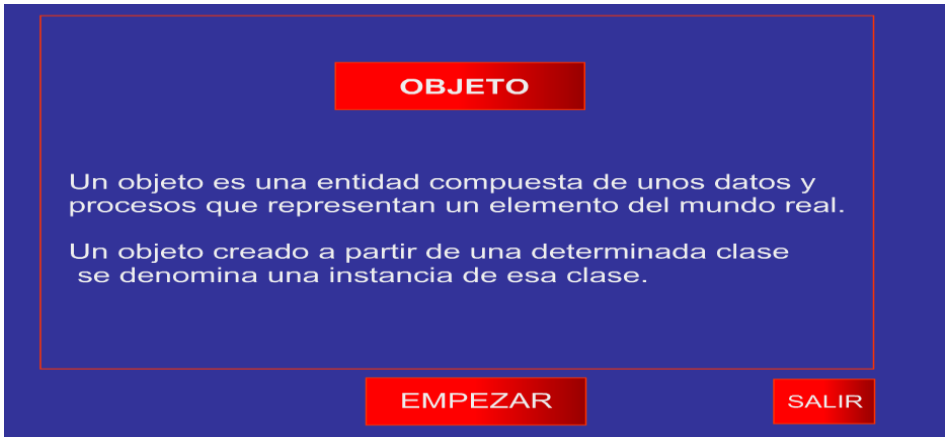


Figura 19-Concepto Herencia

Figura 20- Menú Clase

- En la figura 20 encontramos un menú que al darle clic nos enviara a cada clase ya sea: Automóvil-Persona

Figura 21-Clase Automóvil

- En la figura 21 Mostramos un automóvil el cual puedes cambiar el color y tiene métodos: Avanzar- Arrancar.

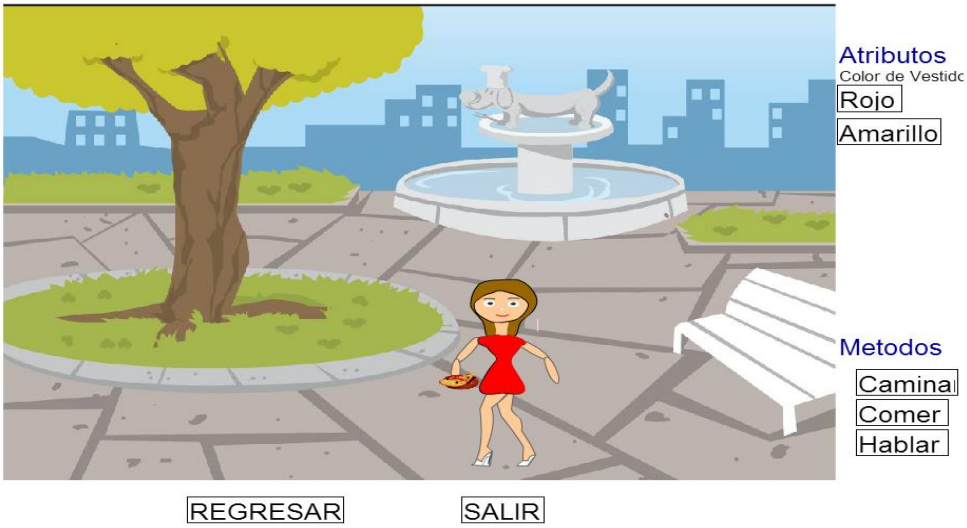


Figura 22-Clase Persona

- En la figura 22 Mostramos un Objeto Personal cual puedes cambiar el color de vestido y tiene métodos: Caminar-Comer-Hablar.

Figura 23-Abstraccion

La PANTALLA será la que nos enviara al tema ***Abstracción*** Al darle clic en entrar que será un botón que nos enviara al tema menú Abstracción.

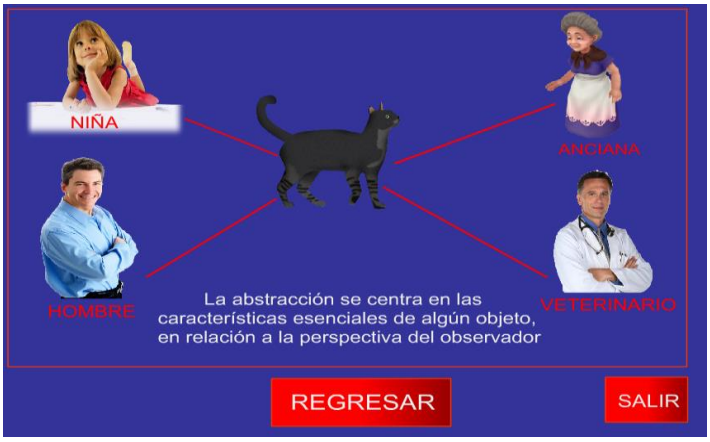


Figura 24-Abstraccion

En esta pantalla explicamos el concepto de abstracción al hora de darle clic a cada Mostrara la forma de observar al gato.

Figura25-Observador Niña

Figura 26-Observador Abuelita

Figuras 27-Observador Veterinario

Figura 28-Observadot Hombre

Figura 29-MenU herencia

Figura 30-Herencia simple

Figura 31-Herencia multiple

10. CONCLUSIÓN

La enseñanza de la POO no es una tarea simple, sin embargo muchos docentes preocupados por el aprendizaje de sus estudiantes, buscan nuevas formas de enseñar este paradigma.

Con la utilización del software educativo, se pretende que los estudiantes puedan comprender más claramente los conceptos relacionados con la POO y su posterior implementación en comparación con los métodos tradicionales de enseñanza.

Realizamos Dos encuestas [anexo 2] para probar el software educativo la cual se realizo a estudiantes del Instituto Tecnológico de Tuxtla Gutiérrez de la materia de programación orientada objetos. Llegando a una conclusión donde los estudiantes tuvieron mejor resultados después de haber usado el software, la encuesta trato sobre los conceptos básicos de la POO.

11. REFERENCIAS BIBLIOGRAFICAS

[1] Martin Laborada Roció 2005. Las nuevas tecnologías en la Educación. Fundación AUNA.

[2] Área Moreira Manuel 2009. Las nuevas tecnologías Educativas. Universidad de la Laguna (España).

[3] Galvis P. “Ingeniería de Software Educativo". Ediciones Unidas. Primera Edición. Santa fe de Bogotá Colombia.1992.

[4] Importancia de la POO Extraída el 1 de octubre del 2011 http://boards4.melodysoft.com/2004BFDP0201/importancia-de-la-poo-7.html.

[5] Primero Objetos con Java: introducción Usando Bluej, David J. Barnes, Michael Kolling, 4, Prentice Hall, 2008.

[6] Ben-Ari, N. Ragonis, R. Ben-Bassat Levy. (2008) A Vision of Visualization in Teaching Object-Oriented Programming, Department of Science Teaching, Weizmann Institute of Science, Israel

[7] Universidad los Andes, proyecto Cupi2 Extraída 22 de octubre 2011 http://cupi2.uniandes.edu.co/sitio.

[8] William Heldman. “Flash Professional CS6 “.Primer Edición. Serious Skills.2000.

[9] Mariano Makedonsky. ”Flash Extremo”. Primera Edición. Lomas de zamora.2009

[10] Programas de animación, Extraída 3 noviembre del 2011 http://www.dibujosinfantiles.org/animacion/programas-para-animar.php

[11] Gayo, G. Agustín Cernuda del Río, Juan Manuel Cueva Lovelle, Marián Díaz Fondón, Ma PilarAlmudena García Fuente, José Manuel RedondoLópez. 2002. Reflexiones y experiencias sobre la enseñanza de POO como único paradigma. Departamento de Informática, Universidad de Oviedo.

[12] Jacobson, I. Booch G. y Rumbaugh J y G. Booch. “El Lenguaje Unificado de Modelo”. Primera Edición. Addison-Wesley. Madrid, España. 2000.

[13] Good T. “Psicología Educativa Contemporánea”. McGraw Hill. (5ta. Ed.). Madrid España. (2001).

[14] Galvis P. “Ingeniería de Software Educativo". Ediciones Unidas. Primera Edición. Santa fe de Bogotá Colombia.1992.

[15] Luis R. Izquierdo “Introducción a la Programación Orientada Objetos” http://luis.izqi.org/resources/ProgOrientadaObjetos.pdf

12. ANEXOS

Anexo [1] Encuesta 30-Nov-2012 se realizo a Alumnos del Instituto tecnológico de Tuxtla Gutiérrez de la carrera de Ingeniería en Sistemas Computacionales tomando a 40 alumnos que han cursado la materia de Programación Orientada Objetos.

1. **Dificultad en la TEORIA de la Programación orientada a Objetos**
    - Difícil
    - Similar
    - Fácil

2. **Dificultad en la Práctica de la Programación orientada a Objetos**
    - Difícil
    - Similar
    - Fácil
3. **¿Qué Factores crees que influyen para tener un buen conocimiento de la Programación Orientada a Objetos?**
    - Lenguaje de programación
    - Falta de herramientas
    - Tiempo en clases
    - Introducción de POO

4. **¿Con que lenguajes te enseñaron programación Orientada a Objetos?**
    - C++
    - C#
    - Java
    - Python
    - Ruby
    - Php

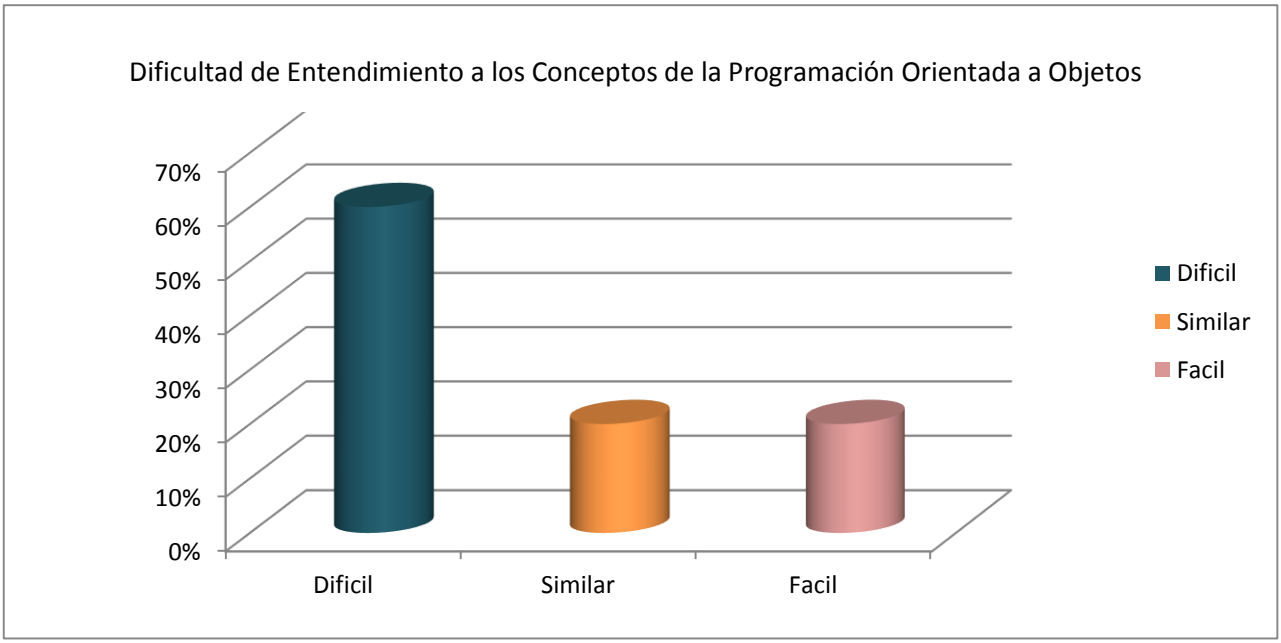
5. **¿Conoces alguna herramienta para el aprendizaje de la Programación Orientada Objetos?**
    - Alice
    - Greenfoot
    - Jeliot 3
    - Bluej
    - Otra Mencionala_____

6. **¿Cuál es la descripción que crees que define mejor el concepto clase en la Programación Orientada a Objetos?**
    - Es un concepto similar al array
    - Es un tipo particular de variable
    - Es un modelo o plantilla a partir de la cual creamos objetos
    - Es una categoría de datos ordenada secuencialmente

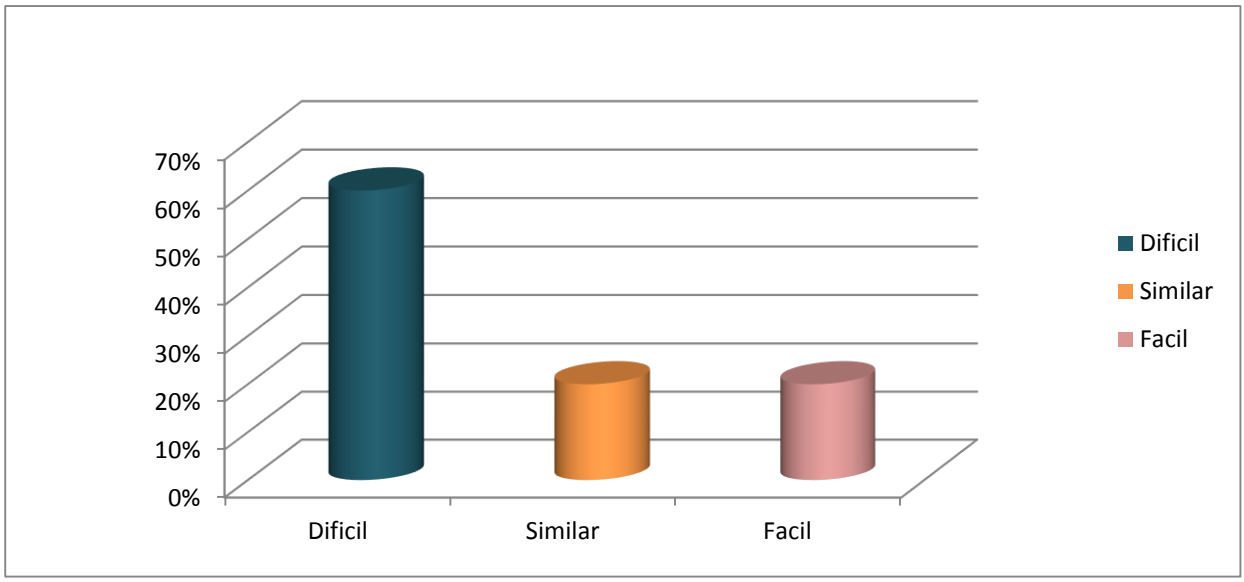
7. **¿Qué elementos crees que define a un objeto?**
    - Su cardinalidad y su tipo
    - Sus atributos y sus métodos
    - La forma que establece comunicación e intercambia mensajes

Resultados de la encuesta realizada alumnos.

- En las Encuestas aplicadas se encontró que un 70% de muestra se encuentra dificultad en entender los Conceptos Básicos de la Teoría POO y un 18% menciono que se le facilita.

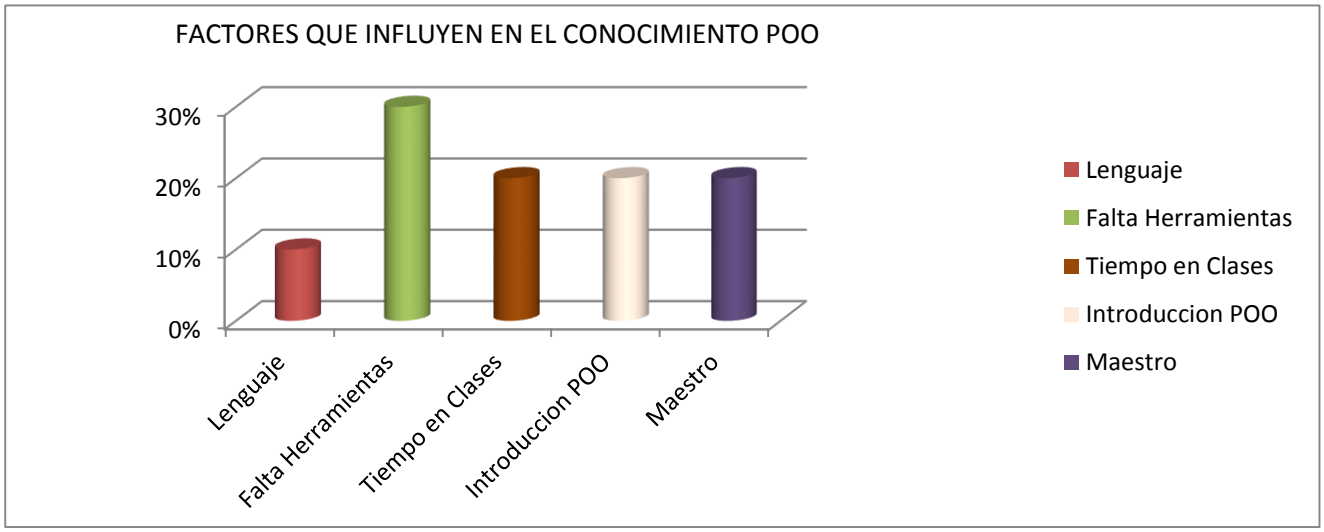


- Concluimos que un 60% de muestra se encuentra en dificultad en la Práctica de POO y un 20% menciona que se le facilita.

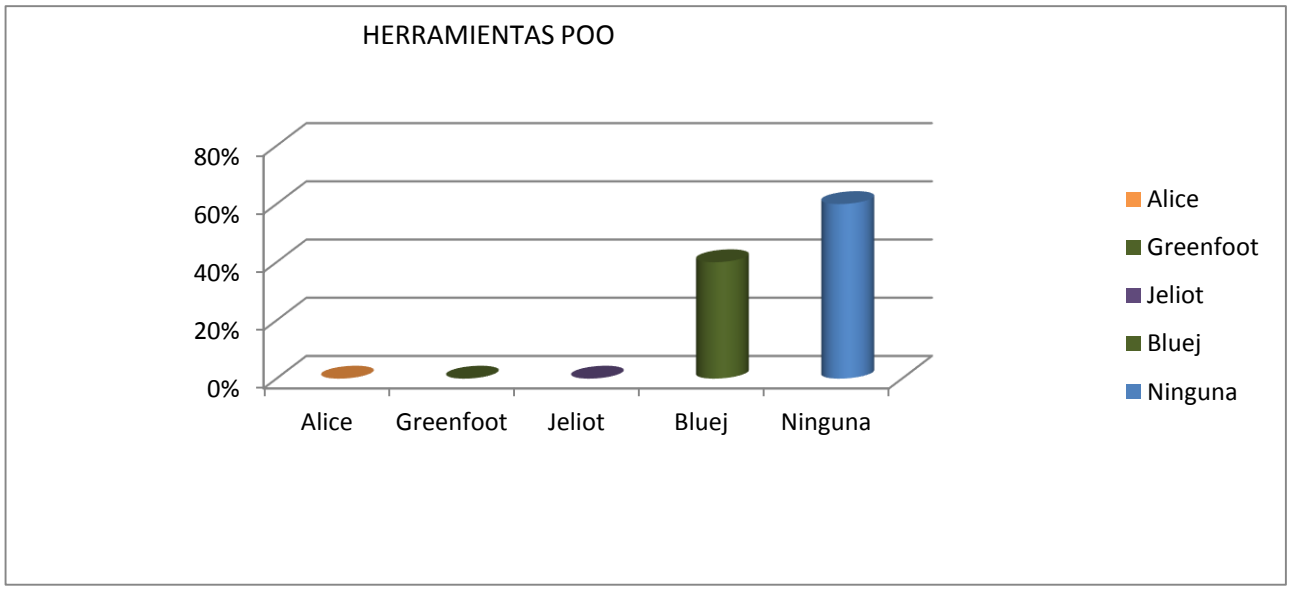


- Los factores que influyen en el conocimiento POO, para los estudiantes existen distintos factores los cuales afectan a la falta comprensión de los conceptos de POO.

Lenguaje	**10%**
Falta Herramientas	**30%**
Tiempo en Clases	**20%**
Introducción POO	**20%**
Maestro	**20%**



- Concluimos que un 60% de los estudiantes no conocen una herramienta de apoyo para el conocimiento de POO ya que solo el 40% conocen la herramienta **bluej.**



Anexo 2

Anexo [2] Encuesta 31-Jul-2013 se realizo a Alumnos del Instituto tecnológico de Tuxtla Gutiérrez de la carrera de Ingeniería en Sistemas Computacionales tomando una muestra de 20 alumnos que cursaron la materia de Programación Orientada Objetos.

Es una agrupación con características semejantes

- ⌘ Atributo
- ⌘ Método
- ⌘ Objeto
- ⌘ Clase

1. Las características fundamentales de un “objeto” son:
   - ⌘ Estado
   - ⌘ Comportamiento
   - ⌘ Identificación

2. Los paradigmas de la Programación Orientada Objetos son:
    - ⌘ Herencia
    - ⌘ Extensibilidad
    - ⌘ Abstracción
    - ⌘ Encapsulamiento

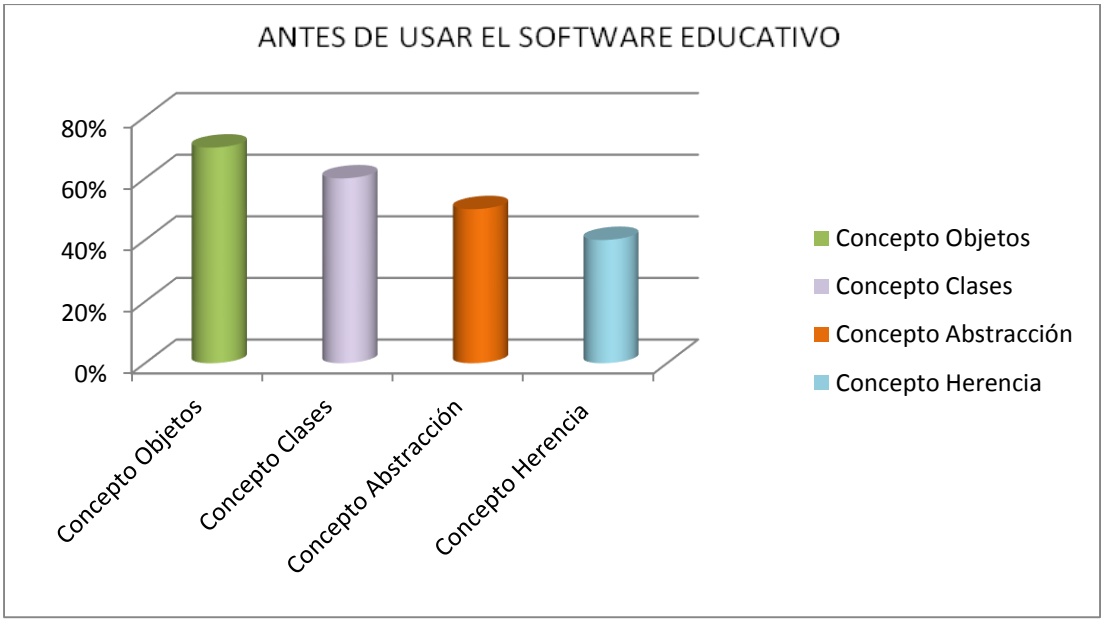
3. ¿Cuál es la descripción que crees que define mejor el concepto “clase” en la Programación Orientada a Objetos?
    - ⌘ Es un concepto similar al de array
    - ⌘ Es un tipo particular de variable
    - ⌘ Es un modelo o plantilla a partir de la cual creamos objetos.

4. ¿Qué elementos crees que define a un objeto?
    - ⌘ Su cardinalidad y su tipo
    - ⌘ Sus atributos y métodos
    - ⌘ La forma en que se establece la comunicación e intercambia mensajes.

5. ¿Qué es Abstracción?
    - ⌘ Expresa las características esenciales de una clase.
    - ⌘ Expresa las características esenciales de un objeto.
    - ⌘ Es la capacidad que tienen los objetos de una clase de responder al mismo mensaje.
6. ¿Que permite La herencia?
    - ⌘ Que los objetos sean creados a partir de otros ya existentes
    - ⌘ Que los objetos obtengan características (métodos y atributos).
    - ⌘ La instanciación y la creación de un objeto.

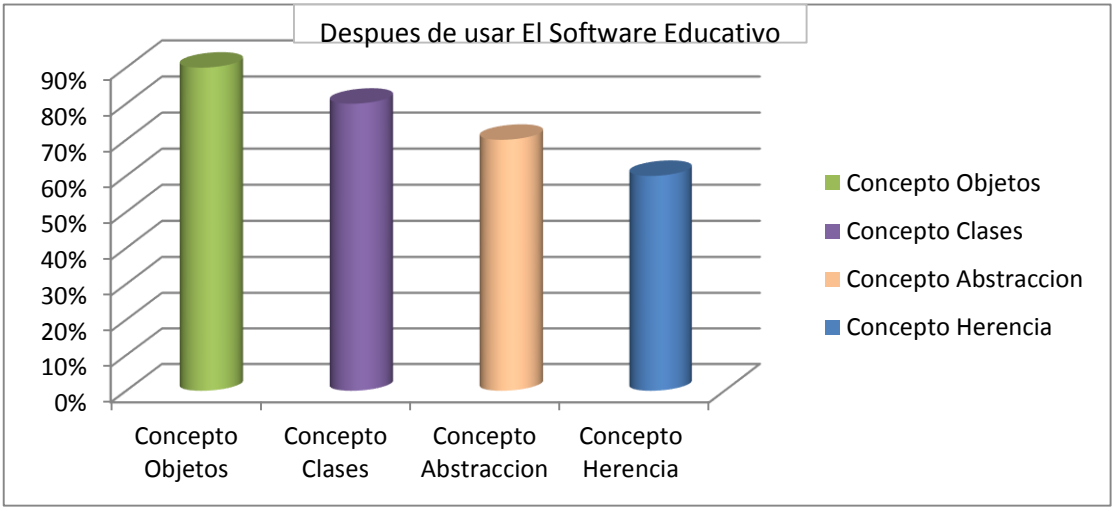
- En la siguiente Grafica mostramos los resultados obtenidos de la encuesta realizada antes de usar la herramienta

Antes de Usar la Herramienta	
Concepto Objetos	70%
Concepto Clases	60%
Concepto Abstracción	50%
Concepto Herencia	40%



- En la siguiente Grafica Mostramos los resultados obtenidos de la encuesta realizada después de usar el software. Llegando a una conclusión donde los estudiantes tuvieron mejor resultados después de haber usado el software, la encuesta trató sobre los conceptos básicos de la POO

Después de Usar la Herramienta	
Concepto Objetos	90%
Concepto Clases	80%
Concepto Abstracción	70%
Concepto Herencia	60%



Observaciones de los alumnos que Utilizaron el software Educativo para el aprendizaje de la Programación Orientada a Objetos.

- → Muy funcional y Practica.
- → Se da entender los conceptos Claves de la Programación.
- → A mí me atrajo lo de herencia simple sobre las utilidades de la clase con sus métodos y atributos que utilizaron
- → Es una muy buena herramienta para poder comprender mejor la programación orientada a objetos de manera interactiva
- → Es un buen programa muy educativo podría implementar se en un nivel de dificultad para el usuario, es fácil de usar es un buen software.
- → Tal vez un poco más de animaciones pero básicamente es sencillo y es lo que lo hace entendible.
- → Creo que es una herramienta que puede ser útil para los estudiantes ya que ayuda a comprender los conceptos de programación que en ocasiones pueden resultar difíciles de entender.
- → La herramienta es muy buena para comprender mejor los conceptos de la programación orientada al objeto.
- → Siento que falto más explicación de lo que se maneja ya que se complementaria bien con lo ya animado para una mejor comprensión.
- → Es un buen programa muy educativo podría implementar se un nivel de dificultad para el usuario fácil de usar es un buen software

INSTITUTO TECNOLÓGICO DE TUXTLA GUTIÉRREZ

Departamento: GESTION TEC. Y VINC
No. de Oficio: DGTyV /349
Fecha: 18/02/13

ASUNTO: **PRESENTACIÓN DEL ALUMNO Y AGRADECIMIENTO**

M.C.A. José Luis Méndez Navarro
Director
Instituto Tecnológico de Tuxtla Gutiérrez
Tuxtla Gutiérrez, Chiapas
P R E S E N T E

El Instituto Tecnológico de Tuxtla Gutiérrez, tiene a bien presentar a sus finas atenciones al (la) alumno (a): **Gloria Guadalupe Mayorga Vidal** número de control: **09270232** carrera de: **Ing. En sistemas Computacionales** quien desea desarrollar en ese organismo el proyecto de Residencias Profesionales denominado **Software educativo para el aprendizaje de la programacion orientada a objetos** cubriendo un total de 640 horas, en un período de cuatro a seis meses, en el periodo Febrero-Junio 2013.

Es importante hacer de su conocimiento que todos los alumnos que se encuentran inscritos en esta institución cuentan con un seguro contra accidentes personales con la empresa **MetLife,** Según póliza **No. AE1489**, e inscripción en el IMSS.

Así mismo, hacemos patente nuestro sincero agradecimiento por su buena disposición y colaboración para que nuestros alumnos, aún estando en proceso de formación, desarrollen un proyecto de trabajo profesional, donde puedan aplicar el conocimiento y el trabajo en el campo de acción en el que se desenvolverán como futuros profesionistas.

Al vernos favorecidos con su participación en nuestro objetivo, sólo nos resta manifestarle la seguridad de nuestra más atenta y distinguida consideración.

A T E N T A M E N T E

SECRETARÍA DE EDUCACIÓN PÚBLICA
Instituto Tecnológico de Tuxtla Gutiérrez
Departamento de Gestión Tecnológica y Vinculación

ING. RODRIGO FERRER GONZÁLEZ
JEFE DEL DEPARTAMENTO DE GESTIÓN TECNOLÓGICA Y VINCULACIÓN

C.c.p. Archivo
C.c.p. Alumno

Recibí

Carretera Panamericana Km. 1080, Tuxtla Gutiérrez, Chiapas. C. P. 29050, apartado Postal 599
Teléfonos: (961) 615-0380, 615-0461 Fax: (961) 615-1687
www.ittuxtlagutierrez.edu.mx

SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA

SEP Institutos Tecnológicos

Subsecretaría de Educación Superior
Dirección General de Educación Superior Tecnológica
Instituto Tecnológico de Tuxtla Gutiérrez

Tuxtla Gutiérrez, Chiapas; 26/Febrero/2013

DSC/019/2013

ASUNTO: CARTA DE ACEPTACION

C. ING. RODRIGO FERRER GONZALEZ
JEFE DEPTO. GESTION TECNOLOGICA Y VINCULACION
PRESENTE.

Por este medio me permito informarle que la **C. Gloria Guadalupe Mayorga Vidal,** estudiante de la carrera de: **Ingeniería en Sistemas Computacionales,** con núm. de Control: **09270232**, ha sido aceptado para realizar su Residencia Profesional en este Departamento denominado**: "Software educativo para el aprendizaje de la programación orientada a objetos"**, con fecha de inicio a partir del 18 de Febrero de 2013, cubriendo un período mínimo de seis meses y no mayor a dos años, haciendo un total de **640 horas**.

Sin mas por el momento quedo de Usted.

ATENTAMENTE
"Ciencia y Tecnología con Sentido Humano"

M.C. AIDA GUILLERMINA COSSIO MARTINEZ
JEFA DEL DEPTO. SISTEMAS Y COMPUTACION

SECRETARIA DE EDUCACION PUBLICA
INSTITUTO TECNOLOGICO TUXTLA GUTIERREZ
DEPARTAMENTO DE SISTEMAS Y COMPUTACION

RECIBIDO
12 MAR 2013
INSTITUTO TECNOLOGICO DE TUXTLA GUTIERREZ
Oficina de Prácticas y Promoción Profesional

C.p. Archivo
AGCM/fylc.

INSTITUTO TECNOLÓGICO DE TUXTLA GUTIÉRREZ

SUBDIRECCIÓN ACADÉMICA
DEPARTAMENTO DE
SEGUIMIENTO DE PROYECTO DE RESIDENCIAS PROFESIONALES

ALUMNO: MAYORGA VIDAL GLORIA GUADALUPE No. DE CONTROL: 09270232

NOMBRE DEL PROYECTO: SOFTWARE EDUCATIVO PARA EL APRENDIZAJE DE LA PROGRAMACION ORIENTADA A OBJETOS. EMPRESA: ITTG

ASESOR EXTERNO: M.C.AIDA GUILLERMINA COSSIO MARTINEZ ASESOR INTERNO: M.C.OCTAVIO ARIOSTO RIOS TERCER

PERIODO DE REALIZACIÓN: ENERO-JUNIO DEL 2013

		SEMANAS														
ACTIVIDAD		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1.Ampliar el Estado de Arte	P	■														
	R	X														
2.Ampliar el Marco Teórico	P		■													
	R		X													
3.Diseño del prototipo	P			■	■											
	R			X	X											
4. Elaborar el primer le reporte	P				■											
	R				X											
5.Aprender usar Flash	P					■	■	■								
	R					X	X	X								
6.Desarrollar la ventana principal	P								■							
	R								X							
7.Detallar el Prototipo	P									■						
	R									X						
8. Crear el diseño en Flash (imágenes a usar)	P										■	■	■			
	R										X					
9.Elaborar el segundo reporte	P										■					
	R										X					

INSTITUTO TECNOLÓGICO DE TUXTLA GUTIÉRREZ

10. Desarrollar la Primera sección Del software.	P																
	R																
11. Hacer las pruebas de la Primera sección.	P																
	R																
12. Desarrollar la Segunda sección Del software.	P																
	R																
13. Hacer las pruebas necesarias de la segunda sección	P																
	R																
14. Desarrollar la Tercera sección Del software.	P																
	R																
15 Hacer las pruebas necesarias de la tercera sección	P																
	R																
16.Probar el software Completo	P																
	R																
17. Implantación	P																
	R																
18. Elaborar el tercer reporte	P																
	R																

OBSERVACIONES		Febrero 25-26	Abril 8-9	Mayo 8-9
ENTREGA DE REPORTES	**DOCENTE**: M.C.OCTAVIO ARIOSTO RIOS TERCER			
	ALUMNO: MAYORGA VIDAL GLORIA GUADALUPE			
	JEFE DE DEPTO. M.C.AIDA GUILLERMINA COSSIO MARTINEZ			

SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA

Subsecretaría de Educación Superior
Dirección General de Educación Superior Tecnológica
Instituto Tecnológico de Tuxtla Gutiérrez

Tuxtla Gutiérrez, Chiapas, 12 Agosto 2013

DSC/082/2013

ASUNTO: CARTA DE LIBERACION

C. C.D. JOSE ERASMO CAMERAS MOTA
JEFE DEPTO. GESTION TECNOLOGICA Y VINCULACION
PRESENTE.

Por medio de la presente me dirijo a Usted, con la finalidad de informarle que la **C. Gloria Guadalupe Mayorga Vidal,** con número de control **09270232**, de la carrera de **Ingeniería en Sistemas y Computación**, ha concluido satisfactoriamente su residencia profesional realizado en el proyecto: "**Software educativo para el aprendizaje de la programación orientada a objetos**", con fecha de inicio a partir del 18 de Febrero de 2013 y finalizando el 09 de Agosto de 2013, cubriendo un total de **640 horas**, bajo el asesoramiento del **C. M.C. Octavio Ariosto Ríos Tercero**.

No teniendo otro particular que tratar, me despido enviándole un cordial saludo

ATENTAMENTE
"Ciencia y Tecnología con Sentido Humano"

M.C. AIDA GUILLERMINA COSSIO MARTINEZ
JEFA DEL DEPTO. SISTEMAS Y COMPUTACION

ESTADOS UNIDOS MEXICANOS
SECRETARIA DE EDUCACION PUBLICA
INSTITUTO TECNOLOGICO TUXTLA GUTIERREZ
DEPARTAMENTO DE SISTEMA Y COMPUTACION
RECIBIDO
12 AGO 2013
INSTITUTO TECNOLOGICO DE TUXTLA GUTIERREZ
Oficina de Prácticas y Promoción Profesional

C.p. Archivo
AGCM/fylc.

CONSTANCIA DE LIBERACIÓN Y EVALUACIÓN DE PROYECTO DE RESIDENCIA PROFESIONAL

MC. Aida Guillermina Cossío Martínez
Jefe del Dpto. de Sistemas Computacionales

Por medio de la presente me permito informarle que se ha concluido la asesoría y revisión del proyecto de Residencia Profesional cuyo título ***"SOFTWARE EDUCATIVO PARA EL APRENDIZAJE DE LA PROGRAMACION ORIENTADA A OBJETOS"*** desarrollado por el C. ***GLORIA GUADALUPE MAYORGA VIDAL*** estudiante de la carrera de Ingeniería en Sistemas Computacionales, Con número de Control ***09270232***, desarrollado en el presente periodo ENERO – JUNIO 2013.

Por lo que se emite la presente **Constancia de Liberación y Evaluación del proyecto** a los 27 días del mes de Junio de 2013

ATENTAMENTE

"CIENCIA Y TECNOLOGÍA CON SENTIDO HUMANO"

M.C. OCTAVIO AUGUSTO RIOS TERCERO
Asesor del proyecto

M.C. IMELDA VALLES LOPEZ
Revisor del proyecto

LIC. JOSE MANUEL SANTIAGO CALVO
Revisor del proyecto

C.c.p.- Alumno
C.c.p.- Archivo

INSTITUTO TECNOLÓGICO DE TUXTLA GUTIÉRREZ

Departamento: GESTION TEC. Y VINC
No. de Oficio: DGTyV /350
Fecha: 18/02/13

ASUNTO: **PRESENTACIÓN DEL ALUMNO Y AGRADECIMIENTO**

M.C.A. José Luis Méndez Navarro
Director
Instituto Tecnológico de Tuxtla Gutiérrez
Tuxtla Gutiérrez, Chiapas
P R E S E N T E

El Instituto Tecnológico de Tuxtla Gutiérrez, tiene a bien presentar a sus finas atenciones al (la) alumno (a): **Maritza Vera Zavala** número de control: **09270260** carrera de: **Ing. En sistemas Computacionales** quien desea desarrollar en ese organismo el proyecto de Residencias Profesionales denominado **Software educativo para el aprendizaje de la programacion orientada a objetos** cubriendo un total de 640 horas, en un período de cuatro a seis meses, en el periodo Febrero-Junio 2013.

Es importante hacer de su conocimiento que todos los alumnos que se encuentran inscritos en esta institución cuentan con un seguro contra accidentes personales con la empresa **MetLife,** Según póliza **No. AE1489**, e inscripción en el IMSS.

Así mismo, hacemos patente nuestro sincero agradecimiento por su buena disposición y colaboración para que nuestros alumnos, aún estando en proceso de formación, desarrollen un proyecto de trabajo profesional, donde puedan aplicar el conocimiento y el trabajo en el campo de acción en el que se desenvolverán como futuros profesionistas.

Al vernos favorecidos con su participación en nuestro objetivo, sólo nos resta manifestarle la seguridad de nuestra más atenta y distinguida consideración.

A T E N T A M E N T E

SECRETARIA DE EDUCACION PUBLICA
Instituto Tecnológico de Tuxtla Gutiérrez
Departamento de Gestión Tecnológica y Vinculación

ING. RODRIGO FERRER GONZÁLEZ
JEFE DEL DEPARTAMENTO DE GESTIÓN TECNOLÓGICA Y VINCULACIÓN

C.c.p. Archivo
C.c.p. Alumno

Carretera Panamericana Km. 1080, Tuxtla Gutiérrez, Chiapas. C. P. 29050, apartado Postal 599
Teléfonos: (961) 615-0380, 615-0461 Fax: (961) 615-1687
www.ittuxtlagutierrez.edu.mx

SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA

SEP Institutos Tecnológicos

Subsecretaría de Educación Superior
Dirección General de Educación Superior Tecnológica
Instituto Tecnológico de Tuxtla Gutiérrez

Tuxtla Gutiérrez, Chiapas; 26/Febrero/2013

DSC/020/2013

ASUNTO: CARTA DE ACEPTACION

C. ING. RODRIGO FERRER GONZALEZ
JEFE DEPTO. GESTION TECNOLOGICA Y VINCULACION
PRESENTE.

Por este medio me permito informarle que la **C. Maritza Vera Zavala,** estudiante de la carrera de: **Ingeniería en Sistemas Computacionales,** con núm. de Control: **09270260**, ha sido aceptado para realizar su Residencia Profesional en este Departamento denominado**: "Software educativo para el aprendizaje de la programación orientada a objetos",** con fecha de inicio a partir del 18 de Febrero de 2013, cubriendo un período mínimo de seis meses y no mayor a dos años, haciendo un total de **640 horas**.

Sin mas por el momento quedo de Usted.

ATENTAMENTE
"Ciencia y Tecnología con Sentido Humano"

M.C. AIDA GUILLERMINA COSSIO MARTINEZ
JEFA DEL DEPTO. SISTEMAS Y COMPUTACION

SECRETARIA DE EDUCACION PUBLICA
INSTITUTO TECNOLOGICO TUXTLA GUTIERREZ
DEPARTAMENTO DE SISTEMAS Y COMPUTACION

RECIBIDO
12 MAR 2013
INSTITUTO TECNOLOGICO DE TUXTLA GUTIERREZ
Oficina de Prácticas y Promoción Profesional

C.p. Archivo
AGCM/fylc.

INSTITUTO TECNOLÓGICO DE TUXTLA GUTIÉRREZ

SUBDIRECCIÓN ACADÉMICA
DEPARTAMENTO DE
SEGUIMIENTO DE PROYECTO DE RESIDENCIAS PROFESIONALES

ALUMNO: VERA ZAVALA MARITZA — No. DE CONTROL: 09270260

NOMBRE DEL PROYECTO: SOFTWARE EDUCATIVO PARA EL APRENDIZAJE DE LA PROGRAMACION ORIENTADA A OBJETOS. — EMPRESA: ITTG

ASESOR EXTERNO: M.C.AIDA GUILLERMINA COSSIO MARTINEZ — ASESOR INTERNO: M.C.OCTAVIO ARIOSTO RIOS TERCER

PERIODO DE REALIZACIÓN: ENERO-JUNIO DEL 2013

		SEMANAS														
ACTIVIDAD		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1.Ampliar el Estado de Arte	P	■														
	R	x														
2.Ampliar el Marco Teórico	P		■													
	R		x													
3.Diseño del prototipo	P			■	■											
	R			x	x											
4. Elaborar el primer le reporte	P				■											
	R				x											
5.Aprender usar Flash	P					■	■	■								
	R					x	x	x								
6.Desarrollar la ventana principal	P								■							
	R								x							
7.Detallar el Prototipo	P									■						
	R									x						
8. Crear el diseño en Flash (imágenes a usar)	P										■	■	■			
	R										x					
9.Elaborar el segundo reporte	P										■					
	R										x					

INSTITUTO TECNOLÓGICO DE TUXTLA GUTIÉRREZ

10. Desarrollar la Primera sección Del software.	P																
	R																
11. Hacer las pruebas de la Primera sección.	P																
	R																
12. Desarrollar la Segunda sección Del software.	P																
	R																
13. Hacer las pruebas necesarias de la segunda sección	P																
	R																
14. Desarrollar la Tercera sección Del software.	P																
	R																
15 Hacer las pruebas necesarias de la tercera sección	P																
	R																
16.Probar el software Completo	P																
	R																
17. Implantación	P																
	R																
18. Elaborar el tercer reporte	P																
	R																

OBSERVACIONES		Febrero 25-26	Abril 8-9	Mayo 8-9
ENTREGA DE REPORTES	**DOCENTE:** M.C.OCTAVIO ARIOSTO RIOS TERCER			
	ALUMNO: VERA ZAVALA MARITZA			
	JEFE DE DEPTO. M.C.AIDA GUILLERMINA COSSIO MARTINEZ			

SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA

SEP
Institutos Tecnológicos

Subsecretaría de Educación Superior
Dirección General de Educación Superior Tecnológica
Instituto Tecnológico de Tuxtla Gutiérrez

Tuxtla Gutiérrez, Chiapas; 12/Agosto/2013

DSC/083/2013

ASUNTO: CARTA DE LIBERACION

C. C.D. JOSE ERASMO CAMERAS MOTA
JEFE DEPTO. GESTION TECNOLOGICA Y VINCULACION
PRESENTE.

Por medio de la presente me dirijo a Usted, con la finalidad de informarle que la **C. Maritza Vera Zavala,** con número de control **09270260**, de la carrera de **Ingeniería en Sistemas y Computación**, ha concluido satisfactoriamente su residencia profesional realizado en el proyecto: "**Software educativo para el aprendizaje de la programación orientada a objetos**", con fecha de inicio a partir del 18 de Febrero de 2013 y finalizando el 09 de Agosto de 2013, cubriendo un total de **640 horas**, bajo el asesoramiento del **C. M.C. Octavio Ariosto Ríos Tercero.**

No teniendo otro particular que tratar, me despido enviándole un cordial saludo

ATENTAMENTE
"Ciencia y Tecnología con Sentido Humano"

M.C. AIDA GUTIÉRREZ COSSIO MARTINEZ
JEFA DEL DEPTO. SISTEMAS Y COMPUTACION

ESTADOS UNIDOS MEXICANOS
SECRETARIA DE EDUCACION PUBLICA
INSTITUTO TECNOLOGICO TUXTLA GUTIERREZ
DEPARTAMENTO DE SISTEMAS Y COMPUTACION

RECIBIDO
12 AGO 2013
INSTITUTO TECNOLOGICO DE TUXTLA GUTIERREZ
Oficina de Prácticas y Promoción Profesional

C.p. Archivo
AGCM/fylc.

CONSTANCIA DE LIBERACIÓN Y EVALUACIÓN DE PROYECTO DE RESIDENCIA PROFESIONAL

MC. Aida Guillermina Cossío Martínez
Jefe del Dpto. de Sistemas Computacionales

Por medio de la presente me permito informarle que se ha concluido la asesoría y revisión del proyecto de Residencia Profesional cuyo título ***"SOFTWARE EDUCATIVO PARA EL APRENDIZAJE DE LA PROGRAMACION ORIENTADA A OBJETOS"*** desarrollado por el C. ***MARITZA VERA ZAVALA*** estudiante de la carrera de Ingeniería en Sistemas Computacionales, Con número de Control ***09270260***, desarrollado en el presente periodo ENERO – JUNIO 2013.

Por lo que se emite la presente **Constancia de Liberación y Evaluación del proyecto** a los 27 días del mes de Junio de 2013

A T E N T A M E N T E

"CIENCIA Y TECNOLOGÍA CON SENTIDO HUMANO"

M.C. OCTAVIO ARIOSTO RIOS TERCERO
Asesor del proyecto

M.C. IMELDA VALLES LOPEZ
Revisor del proyecto

LIC. JOSE MANUEL SANTIAGO CALVO
Revisor del proyecto

C.c.p.- Alumno
C.c.p.- Archivo