Revisión

Incremento de la funcionalidad relacionada con la manipulación de eventos del mouse y los elementos gráficos en el sistema de visualización Polkaw

Increase of the functionalities related to the handling of events of the mouse and graphical elements in the Polkaw visualization system

Ing. Yanetsys González Mojena, Profesora asistente, Universidad de Granma,
ymojenay@udg.co.cu , Cuba.
Ing. Yamira Medel Viltres, Profesora asistente, Universidad de Granma,
ymedelv@udg.co.cu , Cuba.
MSc. Ibet de los Ángeles Pascual Sánchez, Profesora asistente, Universidad de Granma,
ipascuals@udg.co.cu , Cuba.

Recibido: 13/04/2018 Aceptado: 9/07/2018

RESUMEN

Dada la importancia que para el desarrollo de la sociedad actual tiene la formación de profesionales capacitados en la rama de la informática y la computación se hace necesario que los estudiantes adquieran de forma sólida las habilidades para solucionar problemas complejos de forma óptima, apoyados en las propias ventajas de los sistemas informáticos que tienen a su alcance. Se presenta la mejora de una herramienta que se construyó para el apoyo del aprendizaje de la asignatura de Programación II, el Sistema de Visualización de Programas en el lenguaje SubC (SVP-SubC). Este utiliza, para realizar las animaciones, el Sistema de Visualización PolkaW, el cual no ofrece un ambiente gráfico adecuado para la visualización de programas dinámicos. Se evidenció que era factible la inclusión de funcionalidades relacionadas con la manipulación de eventos del Mouse y la incorporación de elementos gráficos adecuados para la animación de la dinámica del comportamiento de estructuras de datos en la ejecución de un programa en SubC. Estas aportaciones son la salida fundamental de la investigación.

Palabras clave: herramienta; aprendizaje; visualización; estructura de datos.

ABSTRACT

Given the importance the formation of professionals qualified in the area of informatics and computing sciences has for the development of modern society, it is necessary that the students

solidly acquire the skills to solve in an optimum way complex problems, supported by the advantages of the informatics´ systems they have at hand. In this paper, an enhancement of a tool built for the learning support of the Introduction to Programming II subject is presented, the Program Visualization System SubC (SVP-SubC). It uses the PolkaW Visualization System to build animations. PolkaW does not supply a suitable graphic environment for the visualization of dynamic programs. It was shown that the inclusions of functionalities related to the handling of events of the mouse and that of graphical elements more suitable for the animation of the dynamics of the behavior of data structures in the execution of a program in SubC was feasible. These contributions are the main results of the investigation.

Key words: tool; learning; visualization; data structures.

INTRODUCCIÓN

En todo el mundo desarrollado y no desarrollado, la educación es una necesidad constante y vital porque muchas personas tienen más de un ciclo profesional a lo largo de su vida.

El desarrollo de las tecnologías ha tenido gran impacto en la educación y sobre todo el desarrollo de software educativos que permiten visualizar determinados procesos facilitando una mejor comprensión de los contenidos mostrados.

La creciente popularidad de la visualización de algoritmos y programas ha llevado al desarrollo de diversas herramientas, muchas de las cuales están diseñadas para tratar una cierta área de la visualización.

En el mundo existen muchos sistemas de visualización de programas entre ellos tenemos los sistemas Zeus, StarLite, Samba y Xtango, Axoft (2005). Algunas de estas aplicaciones muestran de forma gráfica las operaciones básicas con ciertas estructuras de datos, visualizando el proceso que se ejecuta, como Balsa, propuesta por Ruiz (1996).

Almeida (2003), propone la herramienta EDApplets, una aplicación Web orientada a la enseñanza-aprendizaje de la programación y de la algorítmica en las ingenierías, basada en la tecnología de Applets Java y está orientada a la animación y visualización mediante trazas de algoritmos y estructuras de datos.

Stasko (1998) desarrolló una herramienta de visualización denominada Polka, este es un sistema de animación de propósito general, particularmente para la construcción de algoritmos y animaciones de programas. Es descendiente, y a la vez más potente que el sistema XTango. Provee sus propios conceptos abstractos de alto nivel para crear animaciones más fácil y más

rápido que muchos otros sistemas. Los programadores no necesitan ser expertos en diseño gráfico para desarrollar sus propias animaciones.

Los desarrolladores del sistema Polka realizaron una nueva versión para la plataforma Windows llamada PolkaW. El proceso básico de animación en PolkaW consiste en la implementación del algoritmo que será visualizado, luego se especifican los eventos importantes a ser interpretados durante la ejecución del algoritmo. Estos eventos activan rutinas de animación implementadas en el paquete de animación de XTango, donde se crean y manipulan objetos (círculos, cuadrados, líneas, y otros). Las transiciones sobre objetos incluyen movimiento, cambio de color, de tamaño.

En Cuba se han realizado investigaciones en la esfera de informática educativa. En La Universidad Central de las Villas se crearon los sistemas SESE, SEP y Progen, desarrollados por Lezcano (1998), con el objetivo de elevar la calidad del proceso docente educativo.

Debido a las dificultades que presentan los estudiantes de la carrera de informática en el aprendizaje de la asignatura de Estructura de Datos, Frías (2007), en la Universidad de Granma realizó un Sistema de Visualización de Programas (SVP-SubC), que cuenta con amplias facilidades permitiendo visualizar el proceso de ejecución de los programas implementados en el lenguaje SubC. Este en general posibilita la creación de habilidades en los alumnos para hacer una selección adecuada de las estructuras de datos a utilizar frente a un determinado problema logrando algoritmos más eficientes y comprendiendo de forma más clara las operaciones básicas que con estos tipos de datos se realizan y además, es de gran utilidad para la comprensión de algoritmos que presentan procedimientos recursivos. También utiliza el sistema de visualización PolkaW para la creación de animaciones, a través de la llamada a procedimientos y funciones que este incluye, utilizando una función controladora que se va a encargar de ejecutar los eventos del usuario.

El paquete de animación implementa clases para la creación, transformación y localización de los objetos, que son utilizados por SVP-SubC para la representación de los tipos de datos abstractos como pilas, colas, listas, árboles, grafos y otros, así como las operaciones que con tales datos se pueden realizar. Se muestra además la ventana de visualización del PolkaW que es donde se realizan las representaciones de los elementos estáticos y dinámicos que intervienen en la ejecución de un programa.

PolkaW presenta una serie de ineficiencias que le restan funcionalidad y apariencia manifestado en el sistema de visualización de programas SVP-SubC. El mismo no cuenta con documentación del código fuente lo que hace muy difícil de entender, utilizar y transformar. No

presenta una interfaz gráfica adecuada que permita una mejor interacción entre el usuario y el sistema; tampoco tiene funcionalidad para la representación de estructuras más complejas como árboles, listas, grafos, lo que hace que la forma de representar estos datos no sea la más correcta. Actualmente es una ineficiencia que el sistema de visualización del paquete de animación PolkaW no incluya los eventos del mouse lo que dificulta que exista una mejor interacción entre el usuario y la aplicación.

Se puede expresar que existe ineficiencia en el uso de recursos gráficos para la representación de los elementos que intervienen en la ejecución-visualización de programas en SVP-SubC. Por lo que se pretende incrementar la funcionalidad relacionada con el diseño gráfico y la manipulación de eventos en el proceso de ejecución-visualización de SVP-SubC. Para lo cual se hace necesario realizar Estudio del Sistema de Visualización PolkaW para el tratamiento y manipulación de eventos que permita a los usuarios finales tener una mejor interacción y lograr utilizar el mouse en dicha aplicación.

DESARROLLO

Es importante el valor educativo de las técnicas de visualización de software. La animación de algoritmos y la visualización de programas ayudan a los estudiantes a comprender los conceptos de programación y también a los docentes en su tarea de enseñar dichos conceptos. Distintas experiencias con respecto a la aplicación de la Visualización de Software en la enseñanza de la programación se han realizado en varias universidades del mundo. Actualmente se emplean como recurso didáctico tanto en los cursos elementales como en los avanzados (Basik et al. 2006).

Stasko y Lawrence (2001), consideran que aunque todas las animaciones de algoritmos tienen el mismo propósito, el uso de las mismas puede variar considerablemente. Algunas de las aplicaciones parecen ser las más comunes:

- Para acompañar una lectura y ayudar a comprender los conceptos claves que explica el profesor durante la clase.
- En un laboratorio formal donde los estudiantes interactúan con las computadoras.
- Para uso informal por los estudiantes fuera de la clase, en su tiempo libre, para ayudar a comprender más acerca de un algoritmo.
- Para realizar un aprendizaje personalizado e individualizado.

La visualización de programas presenta vistas del código fuente del programa y de las estructuras de datos. La visualización estática de programa realiza un análisis del texto del

mismo y provee la información que es válida para todas las ejecuciones independientemente de su entrada. (Conroy et al. 1972)

La visualización de algoritmos es el campo de la visualización de software que representa la semántica del programa de computadora. Constituye, por ese motivo, una representación más abstracta que la del programa en sí mismo y, al mismo tiempo, ofrece posibilidades mayores de interpretación de su conducta. La visualización de algoritmos comprende tanto la representación estática de los mismos como la dinámica .(Brown 1985).

La visualización estática de algoritmos está representada generalmente por la descripción de la estructura de su especificación (García et al. 1996). En esta se muestran las relaciones entre las distintas componentes del software, que son especialmente útiles en la fase de diseño de los programas. Por lo general, esta representación se realiza por medio de gráficos, grafos y organigramas que permanecen invariables a lo largo del tiempo. Son importantes en cuanto a la visualización de la especificación del algoritmo, pero difícilmente suministran una visión adecuada de su funcionamiento.

La naturaleza dinámica de los algoritmos hace que estos sean difíciles de explicar con medios didácticos clásicos de pizarra, por eso el exhibir su evolución en el tiempo resulta un procedimiento más próximo al modo de operar del algoritmo y por ende más adecuado. Se habla en este caso de animación de algoritmos. Desde la aparición de los primeros sistemas de visualización, este ha sido el principal uso de las visualizaciones de algoritmos. (Brown 1985).

Según Gomes (2003), la visualización de algoritmos se ha desarrollado comenzando desde los primeros diagramas y notas escritas a mano usadas por los programadores y científicos de la computación para describir la estructura del programa, hasta el advenimiento de las herramientas visuales y auditivas automatizadas actuales. En cualquiera de sus dos formas, dinámica o estática, puede ser muy difícil de relacionar con las construcciones del programa que la codifican y que constituyen las especificaciones de su comportamiento.

Los sistemas para visualización de programas se clasifican básicamente en dos categorías: declarativo que consiste en el manejado por datos e imperativo manejada por eventos (Rivera 2006).

Existen herramientas que utilizan ambos estilos, el imperativo y el declarativo una de ellas es el sistema de visualización PolkaW. Stasko (1997), implementa este paquete: un conjunto de herramientas de diseño de gráficos y animación estructurado en C++. El mismo consta de tres niveles de abstracción. El primero y más alto nivel es el *Animator*. Se necesita crear un objeto *Animator* por cada programa a animar. *Animator* maneja principalmente la recepción de los

eventos del programa en cuestión. En el segundo nivel se encuentran las vistas, que es una ventana en el programa. Varias vistas de animación pueden ser abiertas por un programa. Un objeto *Animator* manipulará todas las vistas con las que fue asociado. Finalmente, en el último nivel existen tres clases principales de objetos: *AnimObject*, *Location*, y *Action*.

En el intérprete de SVP-SubC se instancia una subclase de *Animator* que consta de su propia función controladora, la misma tiene que verificar qué evento ocurrió e invoca a la apropiada escena de animación. Los eventos están relacionados con el acceso, reservación y liberación de memoria; invocación de funciones; entre otros.

Una vista de animación es una perspectiva gráfica particular de un programa. Cada una reside en su propia ventana. Una serie de marcos de animación podrán ser generados en la vista. El número de marcos se podrá asignar en tiempo de animación. Este valor es inicializado en cero en la clase miembro. Polka provee una clase base abstracta llamada *BaseView* –de la cual no puede crearse nunca una instancia. Las dos subclases de *BaseView* que se pueden instanciar son llamadas *View* y *StaticView*. *View* puede tener una variedad de objetos gráficos dentro de ella. Un *StaticView* es como un mapa de bits, en esta solo se puede dibujar. (Stasko 1997).

En la mayoría de las ocasiones se utiliza una subclase de *View* para generar animaciones. Además de las funciones ya suministradas por una vista, la misma puede tener escenas de animaciones individuales. Las vistas contienen un sistema de coordenadas de valores flotantes. Típicamente, los valores de (x, y) toman un rango de 0.0 a 1.0, con el origen en la esquina inferior izquierda y aumentando de izquierda a derecha y de la parte inferior a la parte superior. Si la ventana no es cuadrada inicialmente, una de las dimensiones será distorsionada, por ejemplo, los círculos parecerán elipses.

Un *StaticView* difiere de una vista ya que no mantiene actualizada una lista de visualización de objetos gráficos. Así que, tiene mucho menos funcionalidad en relación con la de una vista, por otro lado, es más rápida la visualización, y no usa el espacio de memoria que regularmente se necesita.

Polka provee tres tipos básicos de objetos que son creados y manipulados en una vista para crear animaciones. *Loc* es una posición lógica (x, y) o una localización dentro de una vista de una animación. *AnimObject* es un objeto gráfico como una línea, un círculo o un rectángulo que se interactúan para simular la animación. *Action* es un cambio o una modificación, como un movimiento a lo largo de un camino o un cambio de color. (Stasko 1997).

Location es una localización (x, y) dentro de una ventana de visualización (*View*). Son apropiados para ubicar objetos *AnimObjects* o moverlos dentro de una vista. La habilidad de

salvar puntos de coordenadas lógicas de una ventana *View* es una herramienta poderosa, ya que las localizaciones son identificadas dentro de un sistema de coordenadas de valores flotantes y la misma animación puede ser llevada a cabo en ventanas del tamaño diferente sin llevar a cabo ningún cambio en el control de la animación.

Un *AnimObject* es un tipo de objeto gráfico visible en una ventana de animación. Manipulando estos objetos, cambiando su posición, tamaño, color, visibilidad. En realidad, *AnimObject* es una superclase de todos los tipos de objeto gráficos, por lo que un objeto nunca es instanciado, solo se instancian las subclases individuales. *AnimObject* provee algunos métodos importantes que son heredados por los objetos gráficos, además suministra varias funciones que prácticamente deben ser redefinidas para cada subclase en específico.

Por otro lado, *Action* implica: una acción que encapsula una modificación a un *AnimObject* como cambiar su posición, tamaño, color, relleno, etcétera. Múltiples acciones pueden ser programadas dentro de un *AnimObject* al mismo tiempo, o una acción en particular puede ser usada por varios objetos *AnimObject*. Está compuesta por:

- un tipo de acción, que puede ser un movimiento, cambio de tamaño o cambio de color.
- una secuencia de (dx, dy) denominado un camino o secuencia de pasos en el espacio (x, y).

En un movimiento, los puntos de control o desplazamiento asociados a un camino definen dónde se ubicará próximamente el objeto *AnimObject*. Un punto de control en un camino es como una imagen en una película. En marcos subsecuentes, las imágenes cambian la ubicación en un incremento pequeño, al iterar los marcos en esta serie, el objeto parece moverse a lo largo de este camino. La longitud de una acción es el número de desplazamientos que contiene su camino. Los caminos no solo son usados para el movimiento, cada acción utiliza un componente camino para definir los cambios exactos de un objeto. (Stasko 1997).

PolKaW es una librería estática que trae un grupo de ejemplos sobre su utilización para un mejor entendimiento.

Entre ellos se puede mencionar la representación del algoritmo de las Torres de Hanoi donde el usuario introduce el número de discos y puede ver cómo se crean y luego se realizan los movimientos hasta quedar organizados. Existen muchos ejemplos que también se manejan de forma estática como el Balls que dibuja una determinada cantidad de círculos y cuadrados que al llegar al borde de la ventana cambian de color y trasforma los cuadrados en círculos y viceversa. El sistema te da la opción de animar un cuadrado en las coordenadas dadas por el usuario. Tutorías como su nombre lo indica es una muestra de las múltiples acciones que realiza el PolKaW, mostrando en consola una breve descripción de lo que se visualiza.

En ninguno de los ejemplos antes mencionados se puede ver qué parte del código crea los objetos, hace los movimientos, qué tipos de variables se utilizan, ni cómo se maneja la recursividad en algunas casos, todo esto se realiza de forma interna y no está a la vista del usuario, además de presentar una gran complejidad al tratar de realizar su propia animación si no tiene conocimientos sólidos de programación. Tampoco presenta un ambiente gráfico adecuado para una mejor interacción con el usuario, tema en el que se centró la presente investigación.

Las animaciones para SubC se realizan en la ventana de visualización del PolkaW, lo que trae como consecuencia que muchas de estas dificultades sean ya heredadas en SVP-SubC.

Se hace muy engorroso el trabajo con una aplicación que cuente con poca bibliografía lo que dificulta un mejor entendimiento y utilización. Es trabajoso transformar el PolkaW debido a que no tiene documentación del código fuente y la que se encuentra sobre su modo de uso es muy escasa y no explica la suficientemente claro su uso. Frecuentemente en las aplicaciones de tipo gráfico es deseable poder hacer algo más que simplemente desplegar menús en la pantalla. Es necesario ser más que un mero espectador durante la ejecución y poder por ejemplo: cambiar la perspectiva de su escena, mover objetos dentro de esta o simplemente marcar un punto de interés. Actualmente esto es una necesidad en el sistema de visualización ya que el paquete de animación PolkaW no incluye los eventos del mouse lo que dificulta que exista una mejor interacción entre el usuario y la aplicación. (Holland et al. 2000)

Para la transformación del sistema es necesario tener en cuenta las técnicas de Ingeniería de Software, que garantizan la eficiencia y calidad desde el punto de vista computacional.

Siempre se hace necesario el desarrollo de modelos y diagramas que ayudan en la tarea de realización de software para cumplir con los requisitos del usuario y el sistema se ejecute sin problemas, se detallan los distintos pasos para el desarrollo de la aplicación. Esto incluye la elaboración del modelo del dominio, así como la exposición de los casos de uso en el diagrama de caso de uso del sistema y una breve descripción de cada uno. Se presentan también elementos del diseño e implementación de la propuesta, apoyados en la metodología de desarrollo de software RUP (Proceso Unificado de Racional).

En SVP-SubC, el Sistema de Visualización PolkaW trabaja como una librería estática que se utiliza creando un objeto animador encargado de ejecutar los eventos del sistema llamando una función controladora, los eventos pueden ser varios, desde crear la ventana de animación hasta realizar una animación completa.

En la versión original de PolkaW, el programador es el encargado de implementar las animaciones, desde crear un objeto hasta realizar movimientos, y localizar objetos. Está basado en la programación orientada a objetos donde se implementan un grupo de clases que van a servir para la realización de las Animaciones.

La clase principal es la clase *Animator* que tiene un conjunto de funciones en las que se almacenan el nombre y tipos de parámetros de las rutinas de la función controladora, además, brinda las opciones de eliminar o agregar varias vistas de animación.

View es la encargada de crear la ventana donde se va a mostrar la animación, en un programa pueden abrirse varias vistas de animación.

AminObject es la clase que tiene la función de la creación de los objetos que van a ser animados, pueden crearse círculos, elipses, rectángulos, líneas, y hasta texto.

La clase *Loc* es la que se dedica la localizacion de los objetos que han sido creados, tiene opciones, además, de convertir las coordenadas dadas, a las coordenadas del Polka.

Action, como su nombre lo indica, maneja un conjunto de acciones que realizan con los objetos, dichas operaciones pueden ser: mover, cambiar el tamaño de la figura, cambiar el color, cambiar la posición donde se encuentra el objeto entre otras.

Existen otras clases contenidas dentro de las que se mencionaron que no son menos importantes y que tienen funciones más específicas en el sistema, pero las antes mencionadas recogen la funcionalidad del sistema, así se presentan cuáles son los roles que se desempeñan en el proceso de animación.

Las mejoras que se desarrollaron tienen como objetivo los siguientes requisitos funcionales:

- El sistema PolkaW debe de ser capaz de manipular eventos del mouse.
- Incorporación de objetos relacionados con la visualización de estructuras de datos más complejas (listas, pilas, colas, etc.)
- Mejora de la eficiencia de los algoritmos de búsqueda en el proceso de interpretación de SVP-SubC.

En SVP-SubC, la animación se realiza en el módulo de ejecución, utilizando fundamentalmente la clase INTERPRETECLASS, encargada de la interpretación de programas en SubC. Para la visualización se apoya en el sistema PolkaW. Esto trae consigo que la interpretabilidad y la reutilización del código correspondiente al proceso de visualización no sea adecuada.

En la solución propuesta como mejora a SVP-SubC, el proceso de animación se separa en clases afines, teniendo así un mayor índice de reutilización.

Se adiciona la clase TABLASIMBOLOSVISUAL para la visualización de los tipos de datos en el

proceso de ejecución–visualización. Esta tiene como función principal el mismo que el de una tabla de símbolos convencional, adicionándole estructuras de datos para el control de los objetos de animación que intervienen en un momento determinado de la ejecución de un programa en SVP-SubC, todo esto para lograr separabilidad de funciones, requerida en la Programación Orientada a Objetos.(Pressman 1996)

Por otro lado, se incorporan estructuras de datos más eficientes en la tabla de símbolos buscando mayor eficacia en la búsqueda en el proceso de ejecución–visualización. Entre estas estructuras se encuentra la incorporación de tablas hash y métodos de búsqueda binaria; para las listas de variables, tipos de datos, declaración de funciones y administración de memoria,

Como se mencionó anteriormente, se adicionaron facilidades relacionadas con la manipulación de eventos, entre los que se encuentran los expuestos en la Tabla 1.

	Pulsado	**Soltado**	**Doble clic (pulsado)**
Izquierdo	WM_LBUTTONDOWN	WM_LBUTTONUP	WM_LBUTTONDLCLK
Central	WM_MBUTTONDOWN	WM_MBUTTONUP	WM_MBUTTONDLCLK
Derecho	WM_RBUTTONDOWN	WM_RBUTTONUP	WM_RBUTTONDLCLK

Tabla 1. Eventos del mouse.

Para lograr los cambios al sistema de visualización polka se incluyó en algunos ejemplos los archivos cpp y se eliminó el icono que utilizaba PolkaW como una librería.

En el parámetro lparam de la función WinProc se almacenan las coordenadas del Mouse y de esta forma se toman las coordenadas (x, y), este proceso se realiza creando una variable global en el WinMain declarada como externa en la función DrawinAreaProc donde se toma el evento que ocurrió, y se transforman estas coordenadas a las coordenadas del PolkaW con las función PickCoord(x,y) de la clase View .

Se logra además, que el sistema reconozca cuando ha capturado un objeto con el mouse mostrando en la consola un mensaje.

En la vista de animación se muestra la animación, después que es presionado el botón de inicio de la ventana de control. Incluye opciones de refrescar la vista, mover a la derecha, izquierda, hacia arriba, hacia abajo, acercar y alejar la imagen.

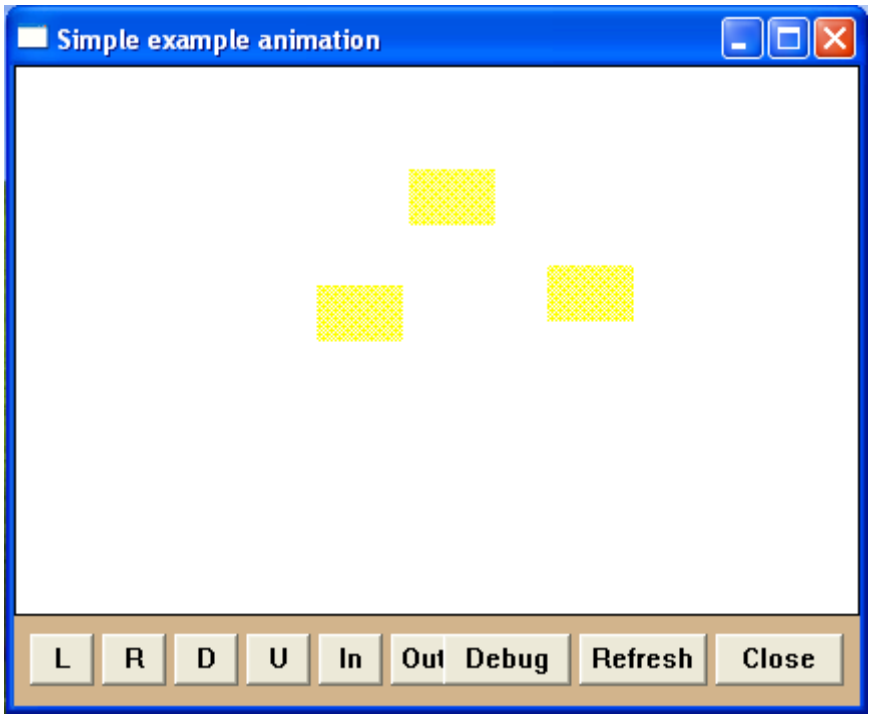

Figura 1. Vista de Animación

Como interfaz de entrada-salida tenemos la consola de la figura 2, que cumple la función de permitir el usuario introducir los datos así como constituir la salida principal del programa.

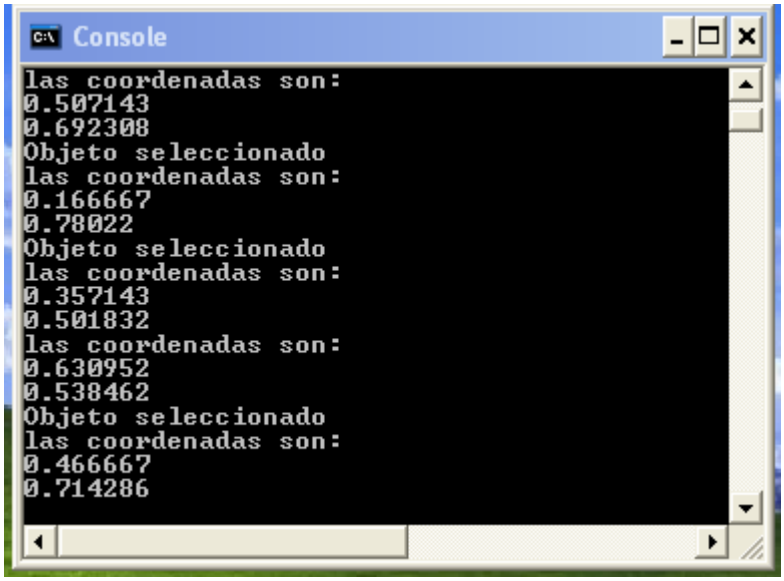

Figura 2. Interfaz de Entrada/ Salida

Para la realización de esta investigación el tiempo estimado de terminación fue de 5 meses. Teniendo en cuenta la escasa documentación que existía y el código fuente que apenas estaba comentado. Se fueron haciendo pruebas de corrimiento y Debugger o depuración para encontrar el flujo de sucesos de activación de los objetos y así agregarle los eventos del mouse, tarea que fue muy trabajosa debido, además, a la forma de uso del sistema de visualización

PolkaW. Para la implementación se utilizó el Microsoft Visual C++ y para el modelado del sistema se utilizó el Racional Rose.

CONCLUSIONES

1.- El trabajo tuvo como salida fundamental el incremento de la funcionalidad relacionada con elementos gráficos y manipulación de eventos del Sistema de Visualización PolkaW, para obtener una mayor interacción entre el usuario y el sistema en el proceso de ejecución-visualización de programas SVP-SubC.

REFERENCIAS BIBLIOGRÁFICAS

Almeida F., et al. (2003). EDApplets: Una Herramienta Web para la Enseñanza de Estructuras de Datos y Técnicas Algorítmitcas.

Axoft (2005). Axoft S.A., Argentina.

Basik, J., et al. (2006). SV in teaching at Brown University. Machine learning: 15.

Brown, J. S. (1985). Process versus product: A perspective on tools for communal and informal electronic learning. Journal of Educational Computing Research, 2.

Conroy, K. y Smith, R. (1972). NEATER2: A AP/I Source Statement Reformatter. Communitations of the ACM: 28.

Frías, I. (2007). Sistema de Enseñanza Asistida por Computadora para la visualización de operaciones sobre estructuras de datos y animación de algoritmos.

García, J., et al. (1996). In IV Ateneo de Profesores Universitarios de Computación.San Luis.

Gómez, H. (2003). Sistematización y técnicas de Visualización de Programas Concurrentes.

Holland, J., et al. (2000). Processes of Inference, Learning, and Discovery, Occasional and eclectic book reviews by Cosma Shalizi. The Best-Laid Schemes o' Mice an' Men.

Lawrence, A., et al. (2000). Empirically Evaluating the Use of Animations to Teach Algoritms. Doctor en Ciencias, Georgia Institute of Technology.: 14.

Lezcano, M. (1998). Vol. Doctor en Ciencias Universidad Central de Las Villas, Santa Clara, Las Villas, pp. 80.

Pressman, R. S. (1996). Ingeniería de Software: Un Enfoque Práctico, Standford.

Rivera, M. (2006). Entorno de programación visual para reglas ECA, Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional, México.

Ruiz, F. (1996). Nuevas herramientas tecnológicas para la realización de cursos por computador. *Revista de Enseñanza y Tecnología* 5: 21 - 31.

Soler, Y. (2007). In Centro de Informática Educativa, Vol. Maestría en Computación Aplicada Universidad Central de Las Villas, Santa Clara, Las Villas, pp. 75.

Stasko, J. (1997). Sistema de visualización de programas Polkaw.

Stasko, J., et al. (1998) Software Visualization: Programming as a Multimedia Experience. MIT Press: 28.

Stasko, J. y Lawrence, A. (2001) Empirically Assessing Algorithm Animations as Learning Aids. 174.