Animación interactiva de algoritmos para cursos de introducción a la programación

Lluís Ribas Xirgo
Departament de Microelectrònica i Sistemes Electrònics (MiSE)
Universitat Autònoma de Barcelona (UAB)
Edif. Q, Campus UAB
08193 Bellaterra (Cerdanyola del Vallès)
Lluis.Ribas@uab.cat

Resumen

La enseñanza de los algoritmos básicos y de los principios de la programación requiere, entre otras cosas, ilustrar cómo se ejecutan los programas. Para ello, pueden emplearse distintas técnicas de animación, desde realizar el seguimiento de la ejecución con esquemas en una pizarra hasta el empleo de material multimedia más o menos interactivo. En este trabajo se presenta una herramienta basada en Javascript que permite realizar un seguimiento visual de la ejecución de programas simples sobre un navegador cualquiera, interactuar con él y, en última instancia, que profesores y estudiantes puedan manipularlo libremente para observar los efectos de los cambios en los programas que realicen.

Summary

Teaching basic algorithms and programming principles needs, among other things, to illustrate how programs are executed. For it, different animation technologies can be used, ranging from following program executions with schemes in a blackboard to the more or less interactive usage of multimedia material. In this work, it is presented a Javascript-based tool that allows doing visual tracing of the execution of simple programs on any web-browser, interacting with it and, last but not least, freely manipulating programs by teachers and students so to show off the effects of the changes.

Palabras clave

Algoritmos animados, Javascript, presentación interactiva, programación, visualización de *software*.

1. Motivación

Para un estudiante que se inicia en la programación resulta fundamental comprender cómo se ejecutan los programas y el efecto que causan en la información que manejan.

En un aula tradicional, los profesores ilustran la ejecución de un programa escribiendo, por ejemplo, la secuencia de instrucciones en una parte de la pizarra y, con más o menos fortuna, la representación gráfica de los datos que se manipulan en la parte adyacente. Con el tiempo, la tiza y la pizarra han dado paso a otras herramientas, incluidas las pizarras electrónicas, pero el principio de funcionamiento sigue siendo el mismo.

En muchos casos, los seguimientos de ejecuciones se han transformado en animaciones específicas, más o menos "programables". Este último término se refiere, por ejemplo, a que para los algoritmos de ordenación se pueda variar el número de elementos o la distribución inicial de los mismos para mostrar sus ventajas e inconvenientes.

De hecho, se pueden encontrar videos en Youtube que ilustran el funcionamiento de una gran variedad de algoritmos, desde los que manipulan pilas y colas hasta otros que trabajan con grafos. También hay una gran variedad de aplicaciones especializadas ([2] y [7], por ejemplo) que permiten ciertos grados de interacción con la animación.

Las animaciones de los algoritmos han permitido, entre otras cosas, agilizar la presentación, introducir elementos gráficos más atractivos y hacerlo accesible a los estudiantes, a través de la web.

Sin embargo, se carece de aplicaciones que se ejecuten directamente en los navegadores, con lo

que los usuarios de las aplicaciones correspondientes deben de realizar una instalación previa de las mismas en plataformas concretas.

En cualquier caso, las animaciones resultan de gran ayuda para la comprensión de los algoritmos pero no permiten interactuar con ellos. Esto les resta interés tanto para los docentes como para los estudiantes. En el primer caso, el uso de la animación en clase resulta útil para ilustrar el concepto pero no para resolver dudas acerca del mismo, especialmente aquellas que deban aclararse con mecanismos que se alejen de las capacidades de programación de la animación. En el caso de los estudiantes, el hecho de que sean receptores pasivos de la información provoca, en muchos casos, el uso de la animación se limita a su función meramente ilustrativa y, a lo sumo, algunos variarán los parámetros de la misma.

A la vista de todo ello, es interesante poder disponer de algún recurso que permita, además de animar la ejecución de los algoritmos, interactuar con ellos. En este sentido, la idea sería parecida a la de disponer de un depurador con una visualización de los datos específica para cada programa.

Por desgracia, el uso de los depuradores de los entornos de programación sólo es factible para los docentes y, además, la presentación de la información (código y datos) no tiene orientación educativa.

A medio camino entre las animaciones estáticas y los depuradores hay una variedad de herramientas de carácter docente [9] que permiten disponer de animaciones programables muy efectivas a cambio de perder generalidad.

El recurso que se presenta pretende resolver los inconvenientes de los depuradores de los entornos de programación en cuanto a su uso en la animación de algoritmos, manteniendo las ventajas que aportan los entornos de programación en cuanto a "interactividad", ya que se puede modificar el programa que se en cualquier momento. Por lo tanto, se sitúa en el mismo plano que las herramientas educativas antes referidas aunque, a diferencia de las mismas, se basa tecnología web para que sea accesible y utilizable desde cualquier dispositivo con navegadores de Internet y se organiza de manera que sea fácilmente modificable y ampliable por sus usuarios.

Para ello, la aplicación se ofrece como *software* de código abierto, con una interfaz de usuario básica que puede complementarse con funciones escritas por los mismos usuarios, de manera que cada profesor pueda adaptarlo a sus necesidades y que, por otra parte, puede servir de ejemplo para estudiantes avanzados.

El recurso ha sido programado Javascript, puesto que se trata de un lenguaje bien soportado por los navegadores de Internet, que es ampliamente conocido y cuyas aplicaciones no requieren de ninguna instalación más que la de algún navegador de Internet capaz de interpretarlo.

Así pues, es posible incluso un escenario en el que el docente emplea el recurso en el ordenador conectado al proyector y los estudiantes lo pueden recrear en sus portátiles o dispositivos móviles. Más aun, pueden realizar ejercicios directamente en ellos.

Antes de pasar a describir la herramienta que se ha desarrollado, se revisará el estado del arte en este tema. En la sección 3 se detalla la arquitectura de la aplicación. La siguiente se dedica a presentar diversos casos de uso de la misma. Finalmente, en la conclusión se repasan los aspectos más destacados tanto del recurso como de su aplicación y se comentan los resultados preliminares de una primera experiencia de su uso real.

2. Animación de algoritmos

La animación de algoritmos es una parte del área de visualización de software, que recibe mucha atención tanto por el impacto que tiene en la productividad del desarrollo de aplicaciones como en el ámbito de la enseñanza.

En [3] se clasifican los distintos sistemas de animación de algoritmos según el método en que se basan: Desde aquellos dirigidos por sucesos hasta los más automatizados. La idea es visualizar la ejecución de un programa cada cierto tiempo:

- Cuando hay cambios de estado significativos (*state-driven*), que afectan a la visualización del mismo.
- Cuando se producen determinados sucesos (*event-driven*) como, por ejemplo, un cambio de valor en una de las variables observadas o llegar a un punto concreto del programa.

- Por observación de su evolución si está construido de forma visual. En cualquier caso, la programación visual puede verse como un elemento complementario de la visualización del software.
- De forma automática, a partir de un análisis del grafo de flujo de control y datos del programa.

En todas las técnicas mencionadas es necesario identificar los datos a mostrar y presentarlos de forma gráfica en una manera bien adaptada al problema que resuelve el programa.

En el ámbito educativo, los problemas son relativamente simples y podrían emplearse formas automáticas de identificación de variables clave, pero sería difícil tener una visualización de las mismas bien ajustada a su uso en docencia.

La programación visual tiene sus ventajas, en cuanto a facilitar el aprendizaje de la algorítmica [4] pero no constituye, por sí misma, una herramienta para la visualización de la información que manipulan los programas. Por otra parte, el empleo de entornos de programación visual requiere de un tiempo de aprendizaje que se resta del dedicado a los entornos de programación en lenguajes textuales como, por ejemplo, el C.

La utilización de técnicas basadas en sucesos proporciona la ventaja de poder visualizar la información del programa sólo en los momentos más interesantes, pero exige determinarlos.

Las actualizaciones de las visualizaciones por cambio de estado son las más comunes en sistemas que se focalizan en la visualización. De hecho, muchos de los sistemas "dirigidos por estados" son sistemas que permiten especificar cómo mostrar la información que manipulan los programas e incorporan mecanismos simples de seguimiento de su ejecución.

El recurso que se presenta se inspira en este último tipo de soluciones, puesto que los algoritmos a animar son relativamente simples y pueden ser seguidos paso a paso.

La visualización se hace con tecnología DHTML, de manera que la especificación de cómo mostrar las variables de un determinado programa se hace mediante una función en Javascript. Los algoritmos se describen en forma de programa con una sintaxis de Javascript restringida, especialmente en lo que atañe a instrucciones de control de flujo.

La ejecución de los programas se lleva a cabo de forma interpretada, igual que en [8], por lo que pueden alterarse en cualquier momento, incluso durante su ejecución (o animación).

3. Organización de la aplicación

El recurso consiste en una pequeña aplicación en Javascript, llamada **jsGorithm**, y un conjunto de casos de uso con ejemplos. En esta sección de comentan los aspectos que atañen la especificación del programa y a su arquitectura.

El programa es de código abierto para poder ser modificado según la conveniencia de cada uso. En cualquier caso, está dividido en dos partes: la de ejecución de programas y la de monitorización de variables.

En general, la primera parte no habría de requerir ningún cambio y la segunda sería la que habría de adaptarse a cada uso que se le diera.

3.1. Edición y ejecución de programas

Los programas pueden editarse con cualquier editor de texto o bien emplear la ventana de edición de la misma aplicación. Los programas tienen deben estar construidos como secuencias de instrucciones en una única línea.

Las instrucciones que se pueden emplear se resumen en la tabla 1: son las básicas para realizar cálculos y movimientos de datos y para construir estructuras alternativas e iterativas.

Instrucción
`variable = expresión;`
`if( ) {`
`} else {`
`while( expresión ) {`
`} /* fin de bloque */`
`do {`
`} while( expresión );`

Tabla 1. Repertorio de instrucciones admitidas.

Con estas instrucciones se puede construir cualquier programa. La instrucción `for` no se ha incluido por ser, de hecho, una estructura compleja que puede simularse con un bucle de tipo "mientras".

En cualquier caso, podría incorporarse en el conjunto ampliando la función de compilación de programas fuente y modificando ligeramente la que se ocupa de ejecutar los programas ya

compilados. (La compilación consiste en generar el grafo de flujo de control y datos del programa fuente para facilitar el seguimiento de la ejecución.)

Las líneas de comentario iniciadas con "//" también se admiten, de la misma forma que los comentarios de bloque, siempre que el bloque esté comprendido en una misma línea.

Las variables que emplea el programa pueden ser definidas en cualquier momento con una asignación. Todas ellas son de carácter global y ninguna puede empezar por "_gr", puesto que es la denominación de todos los objetos de la aplicación.

Para ejecutar un programa hay que cargarlo en la ventana de ejecución haciendo clic en el botón de "Load". Al hacer la carga, el código se verifica y, en caso de haber errores, se comunican al usuario. Los errores más comunes atañen al uso de instrucciones no previstas y a construcciones de control no estructuradas como, por ejemplo, un *else* fuera de lugar.

De hecho, con la carga del código se determina su grafo del flujo de control y de datos para que pueda seguirse su ejecución. En la ventana del código se numera cada instrucción, se indica la posición actual del contador de programa con ">>" y también se muestran los valores a sumar al contador en caso de que el resultado de la ejecución sea una condición cierta o falsa. Las instrucciones que no evalúan ninguna condición tienen ambos valores a 1.

Las instrucciones del programa se muestran con un sangrado a la derecha que refleja a qué estructura pertenecen.

Después de cargar el programa en la ventana de ejecución, puede iniciarse la misma apretando el botón "Play" o el de "Step".

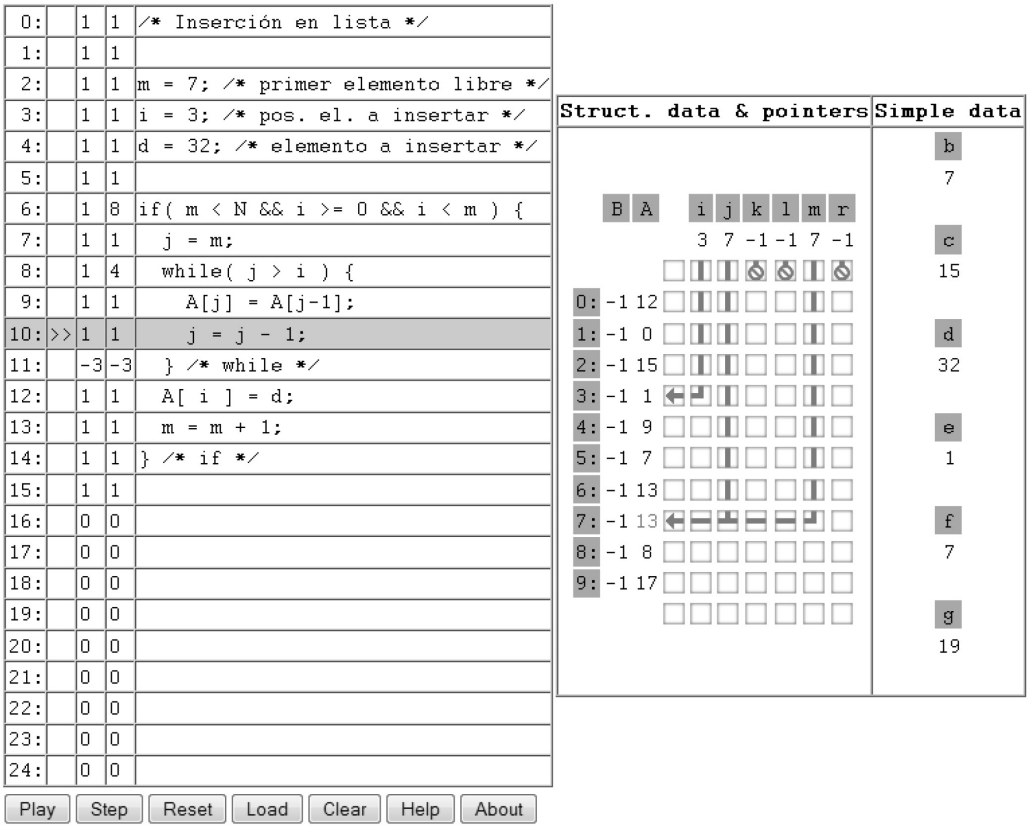

Figura 1. Ventanas de seguimiento de código y visualización de variables de jsGorithm.

El contador de programa apunta a la siguiente instrucción a ejecutar. Después de que se haga clic en "Step", el contador se modifica convenientemente: o se incrementa en una unidad o se le suma un desplazamiento, en caso de que la condición de salto se cumpla.

La ejecución de cada paso implica una llamada a la función `eval()` de Javascript para que evalúe la instrucción. Por este motivo, los programas que pueden construirse son, de hecho, secuencias de instrucciones en Javascript que pueden ser vistas, igualmente, como si fueran en C, C++ o C#, por ejemplo.

De hecho, algunos cursos de introducción a la programación ya emplean directamente Javascript [5] en lugar de lenguajes compilados. Otros siguen prefiriendo estos últimos, aunque emplean intérpretes como el Ch [1].

La ejecución en modo continuo no es más que la realización de una secuencia de pasos que se inicia al apretar el botón de "Play". La ejecución se detiene al llegar a la última instrucción del programa o cuando se aprieta "Pause".

Haciendo clic en "Reset", la ejecución del programa se para, el contador de programa se pone a 0 y se reinicializa la ventana de variables. Tiene, de hecho, el mismo efecto que si se cargara la página de nuevo.

3.2. Monitorización de variables

La organización de la aplicación tiene su reflejo en la estructura de la página web que la contiene: se divide en varias áreas. El área de visualización de las variables es una sección `<DIV>` del documento HTML correspondiente que se denomina "varArea".

Dicha área debe ser inicializada y actualizada con las funciones `_grVarWinInit()` y `_grVarWinUpdate()`, respectivamente. Esta última función se llama después de ejecutar una instrucción (con "Step" o "Play") o de hacer un "Reset".

Aunque la idea original es que cada usuario se cree las funciones que visualizan los datos que maneja un programa, resulta más práctico contar unas funciones en la aplicación que ya estén listas para ser usadas.

Por ello, jsGorithm lleva incorporadas dos funciones de inicialización y actualización del área de variables para un conjunto de datos suficientemente amplio como para cubrir la mayoría de casos de uso habituales de la herramienta.

A continuación, se detalla la configuración básica de visualización del área de variables que se ofrece con las funciones `_grVarWinInit()` y `_grVarWinUpdate()` que se ofrecen con jsGorithm. Dichas funciones trabajan con un conjunto dado de variables:

- `A` es un vector de *N* (10) elementos que se inicializa a valores enteros aleatorios, para que pueda ser empleado en algoritmos de ordenación, por ejemplo.
- `B` es otro vector de *N* elementos que puede actuar como vector auxiliar o bien como vector de direcciones en casos de listas con apuntadores.
- `i` es el índice principal del primer vector
- `j` es un índice secundario
- `k` es un índice del elemento clave (*key*)
- `l` es un índice de elemento en la izquierda
- `m` significa índice de elemento en el medio
- `r` es un índice de elemento en la derecha
- `b` es una variable para datos y de *buffer* para resultados intermedios
- `c` es un contador
- `d` es para datos, en general
- `e` es una variable extra para datos
- `f` es para contener resultados finales
- `g` es una variable de propósito general

Todas las variables se pueden emplear libremente en el programa. Los significados que se les han dado en el listado anterior son a efectos mnemónicos.

Las funciones que inicializan y actualizan el espacio de estas variables muestran los dos vectores en dos columnas en el lado izquierdo y las variables de índice en las columnas siguientes, de manera que puedan trazarse flechas que indiquen a qué elementos de los vectores apuntan. Las variables de datos de muestran en la parte derecha.

Se trata de una visualización relativamente simple, pero efectiva. Es conveniente, en cualquier caso, limitar las variables visualizadas a aquéllas que se empleen en los programas. Para ello, las funciones que se proveen emplean una lista que puede ser modificada libremente. Así, por ejemplo, puede mostrarse sólo un vector o un conjunto más reducido de índices, cosa que

facilita la comprensión de la animación de los algoritmos.

De manera similar, es posible, además, emplear nombres de variables mucho más significativos que los provistos.

4. Casos de uso

En este apartado se comentan los casos de uso probados con el programa jsGorithm a modo de ilustración de sus aplicaciones.

4.1. Algoritmos básicos de programación estructurada

Se trata de ejemplos de cálculos sencillos que requieran de concatenación de estructuras fundamentales de programación, como la alternativa o la iterativa.

En todos los casos basta con unas pocas variables: las que definen el problema, las que representan la solución y alguna de intermedia.

Como se ha comentado, la función de visualización que se incluye en jsGorithm emplea una lista de las variables que mostrará en la ventana correspondiente. Así pues, bastará con adaptar esa lista a las variables que se requieran para cada algoritmo.

El profesor habrá de hacer los programas correspondientes con cualquier editor de texto y, en el momento de utilizar jsGorithm, será necesario que copie el texto de uno de los programas en la ventana de edición de la aplicación.

Suponiendo que esté realizando una presentación en clase, bastará con cargar la página de *jsGorithm.html* desde cualquier navegador de Internet. La ventaja añadida de este sistema de hacer presentaciones es que los navegadores permiten variar el tamaño de lo visualizado, con lo que se puede adaptar a las necesidades de cada auditorio.

Para poner estas animaciones al alcance de los estudiantes basta con que tanto el texto de los programas como la versión correspondiente de jsGorithm sea accesible a ellos a través de Internet.

Esto abre varias posibilidades, incluida la del uso simultáneo por profesores y alumnos en clase, con lo que pueden realizarse clases de teoría mucho más dinámicas o de problemas en los que los estudiantes puedan resolver de una manera más intuitiva los ejercicios que les propone el profesor.

4.2. Esquemas algorítmicos para el tratamiento de secuencias

La importancia del procesamiento secuencial hace que merezca la pena que los estudiantes dediquen un tiempo a comprender y saber poner en práctica los esquemas algorítmicos correspondientes.

Se trata de esquemas de recorrido y de búsqueda para los que pueden encontrarse ejemplos tanto de secuencias temporales como espaciales.

Para las segundas, es necesario incluir vectores que almacenen la secuencia a procesar e índices de los mismos. Con las funciones de visualización de datos que hay, se pueden mostrar tanto vectores como índices. Los índices, además de mostrarse como valor, se muestran con una flecha que señala qué posición del vector están apuntando, tal como se puede apreciar en la fig. 1.

Así pues, se ha desarrollado una versión de jsGorithm para esquemas algorítmicos de recorrido y búsqueda, con varios programas de ejemplo.

De manera similar al caso anterior, los profesores pueden emplearlo en clase y también ofrecerlo a los estudiantes para que resuelvan ejercicios en el tema.

4.3. Algoritmos de ordenación

Se trata de una clase de algoritmos de la cual existe una gran variedad de recursos educativos, por lo que debe fijarse cuál es el objetivo de la animación.

En el caso que el objetivo sea mostrar un algoritmo que requiera de un anidamiento de esquemas de procesado de secuencias, basta con emplear la misma versión de jsGorithm que para el caso anterior con los programas de ordenación que se quieran mostrar.

Hay que tener presente que jsGorithm construye el grafo del flujo de ejecución de los programas que carga, que no pueden incluir definiciones de funciones. Por lo tanto, no puede animar la ejecución de programas recursivos. Así pues, es válido para aquellos algoritmos de ordenación que se resuelvan de esta manera.

Esta restricción no es muy significativa para cursos de introducción a la programación o de fundamentos de informática.

Si el objetivo es comparar la ejecución de varios algoritmos, hay que emplear el mismo número de ventanas del navegador con el jsGorithm. Por cuestiones prácticas, sólo se deben de comparar dos algoritmos que deben estar en sendas ventanas, una al lado de la otra. Dado que no es posible iniciar los dos programas al mismo tiempo, se recomienda introducir alguna instrucción de relleno (basta una línea de comentario, por ejemplo) al principio del programa cuya ejecución se empiece más tarde. Hay que tener presente también que los programas a comparar tengan una inicialización de los vectores a ordenar con los mismos valores.

Finalmente, especialmente para ilustrar los temas de la complejidad algorítmica, es interesante disponer de una serie de casos con vectores inicializados a valores concretos y que los mismos programas realicen un conteo del número de comparaciones e intercambios.

4.4. Estructuras de datos dinámicas

Para ilustrar las aplicaciones de las estructuras de datos dinámicas se pueden emplear listas en vectores. (No se tratan aquí ni árboles ni grafos.)

En el caso de las pilas y de las colas, puede bastar con un único vector, pero hay que tener un programa para cada operación: uno de inserción y otro de eliminación, como mínimo. De cara a realizar las presentaciones, es conveniente disponer de un programa de inicialización de la estructura de datos.

El guión de la presentación puede ser, para el caso de una pila, como sigue. Se explica la representación de la pila en jsGorithm y se inicializa con algunos valores, cargando el programa correspondiente. Luego se carga el *push* y se repiten varias ejecuciones hasta llenar la pila. Después se vacía completamente con ejecuciones repetidas del programa de *pop*. (La repetición se consigue haciendo un "Reset" seguido de un "Play" después de que el programa cargado haya llegado a su fin.) Finalmente, puede cargarse de nuevo el programa de *push* para poner algún elemento nuevo en la pila.

Hay que hacer notar que el "Reset" no modifica el contenido de las variables del programa y que, en la actualidad, el cambio de esos valores sólo puede hacerse por programa.

4.5. Variables dinámicas

En la actualidad, jsGorithm no ofrece soporte para trabajar con variables dinámicas porque se tratan de forma diferente en Javascript que en C y porque su representación visual es más compleja.

Aun así, se puede emular con dos vectores, uno de ellos representando la dirección y otro el contenido de una determinada variable.

En cualquier caso, se trata de una opción de relativa utilidad, puesto que lo interesante sería visualizar las variables y sus interrelaciones de forma más gráfica.

5. Conclusiones

La animación de los algoritmos y programas es muy útil para la enseñanza y el aprendizaje en cursos de introducción a la informática o a la programación. En este artículo se ha presentado un recurso destinado a la animación de programas que puede ser empleado tanto en el aula como de forma individual.

El programa jsGorithm incorpora tanto una parte de seguimiento de programas como otra para la visualización de las estructuras de datos que manipulan. Esta segunda parte se puede adaptar a cada uso en particular para hacer más comprensible la visualización del conjunto.

Este recurso está disponible en [6] para su descarga y para su prueba directa vía web. En este sentido, hay que destacar que las ventajas del recurso propuesto respecto a otras herramientas similares residen, precisamente, en que es posible emplearlo directamente en cualquier navegador tanto de forma remota como local y, sobre todo, en que se puede modificar libremente. En concreto, es posible cambiar la parte de visualización de variables para adaptarla a casos de uso específicos.

En este trabajo también se han presentado diversos casos de uso para ilustrar su funcionamiento. Todos ellos han sido probados durante el primer semestre del curso 2010/11 en la asignatura de Fundamentos de Informática, buena parte de la cual es una introducción a la programación.

Los resultados en el aula de este primer curso de aplicación son positivos en cuanto a la mejora de la agilidad en la presentación de los algoritmos y de su posterior seguimiento en clase. La percepción de los estudiantes es buena pero las

encuestas informales realizadas han reflejado también un cierto descontento con la rapidez de las explicaciones. En cuanto al uso de este recurso por los estudiantes, cabe decir que ha sido bastante residual.

Así pues, se prevé continuar con el uso de este recurso en clase e introducir mecanismos para facilitar su uso por parte de los estudiantes.

En próximas ediciones del curso, se realizarán encuestas específicas que permitan mejorar el entorno de jsGorithm de acuerdo con las necesidades que se detecten.

Referencias

[1] H. H. Cheng. *C For Engineers & Scientists, An Interpretive Approach*, McGraw-Hill March 2009.

[2] Concepción, A.I.; Cummins, L.E.; Moran, E.J.; Do, M.M. "Algorithma 98: An Algorithm Animation Project" In *Proc. of SIGCSE'99*. 301–305. ACM, New York, NY, USA. 1999.

[3] Kerren, A.; Stasko, J.T. "Algorithm Animation" *Software Visualization State of the Art Survey*. Chapter 1, 1–15. Springer Lecture Notes in Computer Science LNCS 2269, (Editor Stephan Diehl), 2002.

[4] Powers, K.; Gross, P. (moderators) "Tools for Teaching Introductory Programming: What Works?" In *Proc. of SIGCSE'06*. 560–561. Houston, Texas, USA. March 2006.

[5] D. Reed. *A balanced introduction to computer science*, Prentice-Hall, 2008.

[6] Ribas Xirgo, Ll. "jsGorithm: simple algorithm animation," Sourceforge. Since 2011-04-18. [jsgorithm.sourceforge.net/]

[7] Rößling, G. *ANIMAL-FARM: An Extensible Framework for Algorithm Visualization*. VDM Verlag. Saarbrücken, 2008. [Disponible en www.algoanim.info]

[8] Sankupellay, M.; Subramanian, P. "Teaching C Programming with the Aid of an Interpreter – Online Interpreter for Novice C Programmer" Jurnal Teknologi, 43(D). 33–44. Universiti Teknologi Malaysia. Dec. 2005.

[9] J. Urquiza-Fuentes, J.A. Velázquez-Iturbide. "A Survey of Successful Evaluations of Program Visualization and Algorithm Animation Systems," in *ACM Trans. on Computing Education*, Vol. 9, No. 2. June 2009.