Construcción "sin esfuerzo" de animaciones de programas basadas en la web

Jaime Urquiza Fuentes

Departamento de Lenguajes y sistemas informáticos, Universidad Rey Juan Carlos,
C/ Tulipán s/n, 28933 Móstoles, Madrid, España
jaime.urquiza@urjc.es

Abstract. En este artículo se describe la extensión de un entrono de programación para generar animaciones de programas basadas en la web. Dicha generación se realiza sin apenas esfuerzo por parte del usuario, ya que se minimiza la información extra que debe proporcionar para generar dichas animaciones. Esa información extra se limita a la personalización de las animaciones. Se describen los diferentes tipos de información que componen las animaciones de programas, su proceso de construcción y los diseños alternativos para su publicación en la web. Se ha evaluado su efectividad pedagógica así como el grado de satisfacción de los usuarios. Los resultados obtenidos son prometedores, ya que los usuarios opinan que el proceso de construcción de las animaciones es fácil de aprender y usar, también opinan que las animaciones basadas en la web son útiles como herramientas educativas. Además, existe evidencia empírica de que la construcción de estas animaciones permite obtener un mayor grado de comprensión de los algoritmos para los que se construyen las animaciones.

1 Introducción

Las animaciones de software se usan en la enseñanza de la informática desde principios de los 80 [1]. Estas animaciones se pueden usar en múltiples escenarios [2]: clases, laboratorios, estudio fuera de clase, etc. Sin embargo, a pesar del potencial educativo que poseen, su uso no se ha incorporado al día a día de la enseñanza de la informática. En el informe del grupo de trabajo sobre visualización de algoritmos organizado en la conferencia ITiCSE 2002, se pueden encontrar datos de tres encuestas sobre el uso educativo de las visualizaciones [3]. La encuesta previa a la conferencia aporta información detallada sobre las razones por las que los encuestados se muestran reacios o creen que no son capaces de utilizar animaciones. Dichas razones se pueden dividir en varios grupos: aquellas relacionadas con el tiempo necesario para preparar la infraestructura (por ejemplo la instalación del software), aquellas relacionadas con el tiempo necesario para construir las animaciones, y aquellas relacionadas con la calidad y adecuación de las herramientas a la asignatura (por ejemplo adaptar las animaciones al enfoque de la asignatura).

Otro grupo de trabajo [4] centró su atención en reducir el esfuerzo necesario para usar animaciones en el ámbito educativo, identificando puntos en los que el profesor

recibe más carga de trabajo, así como dando guías para facilitar la adopción de un sistema de animación. En concreto se tratan la independencia de la plataforma y la diseminación de estas herramientas dentro de la comunidad educativa. Ambos aspectos se suelen mejorar a través de la web.

WinHIPE es un entorno de desarrollo integrado (IDE), al que se le ha añadido la posibilidad de generar visualizaciones estáticas, que el usuario puede personalizar y utilizar posteriormente para la construcción de animaciones (visualizaciones dinámicas). Estas animaciones son discretas, en el sentido en que están compuestas de una secuencia de visualizaciones estáticas. De esta forma se ha implementado un enfoque para la construcción y mantenimiento de animaciones sin esfuerzo [5,6].

Aquí se presenta la extensión de nuestro marco de trabajo al ámbito de las animaciones basadas en la web (animaciones web, para el resto del texto). En primer lugar se trata el problema de la generación, uso y mantenimiento de las animaciones individuales. En segundo lugar se describe su evaluación como herramienta educativa en el ámbito de la animación de algoritmos. Finalmente se exponen las conclusiones y trabajos futuros.

2 Animaciones web

En esta sección se tratan los contenidos, el proceso de construcción y el diseño gráfico de las animaciones basadas en la web. La publicación de animaciones de algoritmos en la web es un tema ya tratado por otros autores [2]. Algunos de los sistemas de animación basados en la web son applets diseñados ad-hoc para visualizar determinados algoritmos [7-10]. Otros sistemas, más flexibles, permiten al usuario la generación de sus propias animaciones [11-18]. La extensión del entorno WinHIPE aquí presentada pertenece al último grupo.

La idea de la generación automática de animaciones web con WinHIPE se describió por primera vez en [15]. El resultado era una página web dinámica, cuya estructura era una lección sobre un determinado algoritmo. Dicha estructura estaba compuesta de cuatro secciones (ver figura 1). Las dos primeras secciones describían el problema y el algoritmo que lo resolvía; estas secciones se generaban con un simple comentario, siendo el usuario el responsable de su generación e inserción en la página web. Las secciones tercera y cuarta; contenían respectivamente el código fuente del programa que implementaba el algoritmo y que posteriormente sería animado, y la animación propiamente dicha.

Tanto el texto como el diseño gráfico de la página web se podían cambiar con un editor de HTML. Sin embargo, si el usuario quería cambiar la animación, debía construirla entera desde el principio; aunque esta tarea no es muy costosa, la aplicación no proporcionaba ninguna ayuda para realizarla. Dicha ayuda constaría de tareas simples; como la búsqueda de los ficheros de programa y expresión; y la utilización de la información de personalización de la animación: selección de fotogramas y configuración gráfica de las visualizaciones estáticas.

Estamos especialmente interesados en ofrecer al profesor una herramienta que le permita construir animaciones web con poco esfuerzo. En consecuencia, hemos rediseñado y reimplementado la herramienta. Hemos cambiado los contenidos de las

páginas web y su proceso de construcción, también hemos enriquecido su diseño gráfico mejorando su usabilidad.

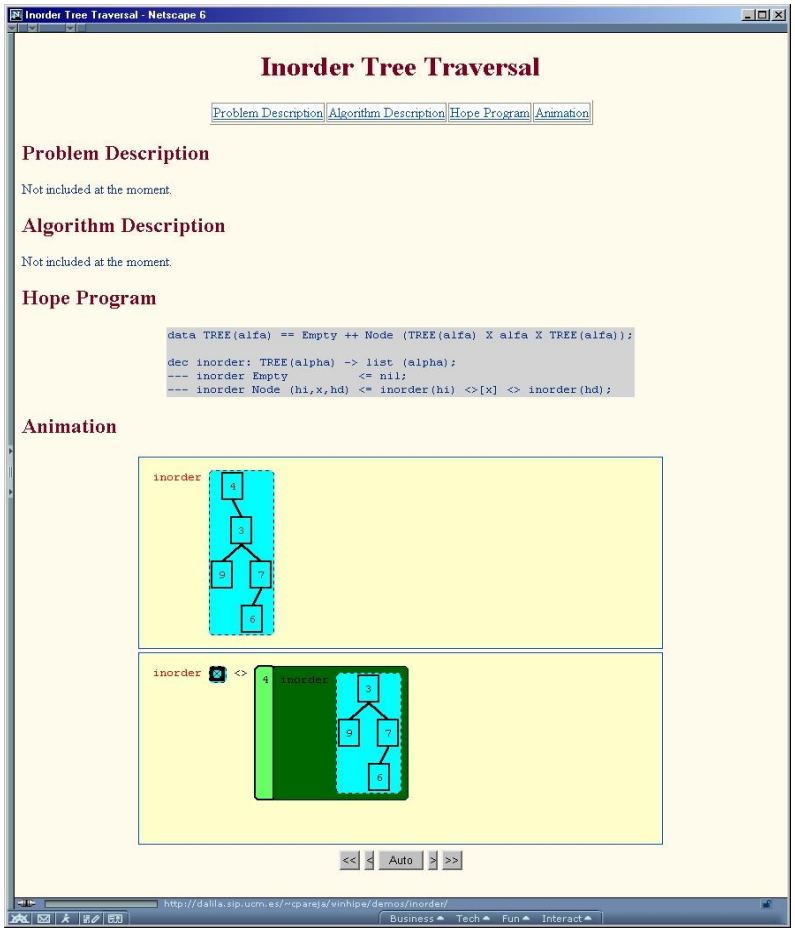

Fig. 1. Un ejemplo de las primeras animaciones web generadas con WinHIPE.

2.1 Contenidos de las animaciones web

Hemos cambiado los contenidos de las animaciones web para que su proceso de modificación se más fácil de cara al usuario. Ahora, una animación web contiene toda la información necesaria para realizar cambios sobre ella. Estos cambios pueden influir en su apariencia, secciones de contenido textual, e incluso la propia animación. Las animaciones web se componen de información de dos clases diferentes. En primer lugar tenemos la información visible, que es aquella que será procesada por el navegador web para visualizar la animación web; la información

visible estará compuesta de código HTML, CSS e imágenes. En segundo lugar tenemos la información de mantenimiento, que es aquella información necesaria para la gestión y modificación de las animaciones web; está información se almacenará en XML, su DTD se puede ver en el anexo A, y un ejemplo de dicha información se encuentra en el anexo B.

Los contenidos de una animación web se dividen en tres partes: texto, animación y, apariencia. Los contenidos de texto tienen la información referente a las descripciones del problema y del algoritmo. Los contenidos de animación tienen los datos necesarios para construir y reproducir la animación: código fuente, expresión evaluada, lista de fotogramas que forman la animación, y los datos de configuración que describen la representación gráfica de las visualizaciones. Los contenidos de apariencia describen el aspecto de la página web que contiene la animación web. En la tabla 1 se puede ver un esquema de los contenidos y la información de una animación web.

Tabla 1. Clases de contenidos e información de una animación web.

Info. \ Cont.	**Texto**	**Animación**	**Apariencia**
Información de mantenimiento	Elementos XML que describen el título de la animación, y las descripciones del problema y la solución.	Elementos XML que describen el código fuente, la expresión evaluada, la lista de fotogramas y la configuración de la representación gráfica de las visualizaciones.	Elementos XML que describen el aspecto de la página web.
Información visible	Elementos HTML para el título de la animación, y las descripciones del problema y la solución.	Elementos HTML para el código fuente, la expresión evaluada, la lista de fotogramas y la configuración de la representación gráfica de las visualizaciones.	Hoja de estilos CSS.

Nótese que las animaciones web son reusables desde dos puntos de vista diferentes. Por un lado, son reusables desde el punto de vista del usuario, ya que contienen toda la información necesaria para modificarlas. Por otro lado, son reutilizables desde el punto de vista de la aplicación y la plataforma, ya que su implementación se hace con XML. Por lo tanto, no sólo se pueden usar en el entorno WinHIPE, sino en cualquier aplicación que trabaje con XML.

En resumen, los contenidos de las animaciones web nuevas reducen el esfuerzo que un usuario debe dedicar a su modificación y mantenimiento, ya que no tiene que recordar ninguna información sobre contenidos o configuración. Además, estas animaciones web son reutilizables a nivel de usuario y aplicación.

2.2 El Proceso de Construcción

El proceso de construcción de las animaciones web se divide en dos fases: generación de la animación del programa y generación de la animación web. La figura 2 ilustra el proceso completo.

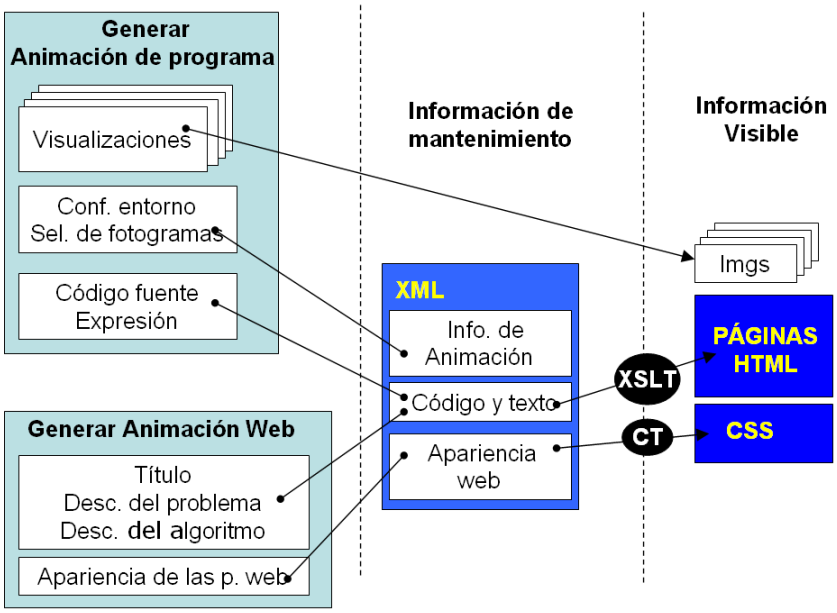

Fig. 2. El proceso de construcción de las animaciones web

En términos generales, la fase de generación de la animación del programa se componen de los siguientes pasos: edición y compilación del programa, edición y evaluación de la expresión, y selección de los fotogramas que compondrán la animación. El lector interesado puede encontrar más información sobre esta fase en [5,6].

La animación web se genera con ayuda del interfaz de usuario de IDE WinHIPE. El usuario debe introducir la información de mantenimiento relativa a los contenidos de texto y apariencia. La información de texto se compone del título de la animación web, la descripción del problema y del algoritmo. La información de apariencia es el estilo con el que se generará la animación web. Todos los datos generados durante la construcción de la animación del programa, junto con la información proporcionada por el usuario durante la generación de la animación web; forman parte de un documento XML donde se encuentra la información de mantenimiento y la secuencia de fotogramas que forman la animación en sí.

Es importante destacar que hasta ahora, el usuario ha seguido los mismos pasos que daría para construir cualquier animación. Además, deberá añadir explicaciones del problema y el algoritmo, y deberá especificar la apariencia de la animación web.

A partir de este momento el usuario a acabado su trabajo, el resto del proceso es automático.

Para terminar, el proceso automático genera la información visible, excepto los fotogramas, que se han generado previamente. La información visible de los contenidos textuales y la animación, se generan en forma de código HTML (distribuido en una o varias páginas web, como se verá en la siguiente subsección), utilizando trasformaciones XSL (XSLT) La información visible correspondiente a la apariencia será construida mediante una transformación *ad-hoc* (CT) cuyo resultado es una hoja de estilos CSS.

2.3 Diseño gráfico de las páginas web

En la primera versión de las animaciones web [15], éstas eran una página web donde se podía navegar a través de enlaces locales y movimientos verticales de scroll. Y contenían la información visible correspondiente a los contendidos de texto y animación (ver figura 1)

Hemos enriquecido este formato para permitir diferentes presentaciones de las animaciones web. Dos de estos formatos se inspiran en principios generales de usabilidad de páginas web [19], les llamamos *Framed1* y *Framed2*. El tercero se diseñó pensando en ofrecer la máxima flexibilidad al usuario a la hora de ver la animación web, le llamamos *Star*.

El usuario (que está viendo la animación web) puede escoger, y trasladarse, entre los cuatro formatos, los tres anteriores junto al antiguo, gracias a la inclusión en todos ellos de una barra de navegación (ver figura 3) situada en la parte superior izquierda de la animación web. Además, con esta barra de navegación también se permite el acceso a los contenidos de la animación web codificados en XML.

Fig. 3. Barra de navegación

Los principales requisitos de usabilidad que se han considerado son: minimizar la cantidad de información oculta en la página (scroll vertical), ajustar el espacio ocupado por las facilidades de navegación a un máximo del 20% del espacio disponible en la pantalla, tiempos de respuesta predecibles (sabiendo que siempre dependen del número y tamaño de los fotogramas de la animación), utilización de hojas de estilo CSS enlazadas, y utilización de marcos embebidos (in-line frames)

El formato *Framed1* (ver figura 4) usa un marco embebido, permitiendo al usuario acceder a cada sección de la animación web con un solo clic de ratón. La barra de scroll aparecerá sólo cuando el contenido de la sección sea mayor que el espacio disponible dentro del marco embebido. Tanto el título como los enlaces directos a las secciones están permanentemente visibles en la pantalla.

El formato *Framed2* (ver figura 5) es similar a *Framed1*, pero con dos marcos embebidos, en vez de un solo. Permite acceder de forma simultánea a dos secciones de la animación web

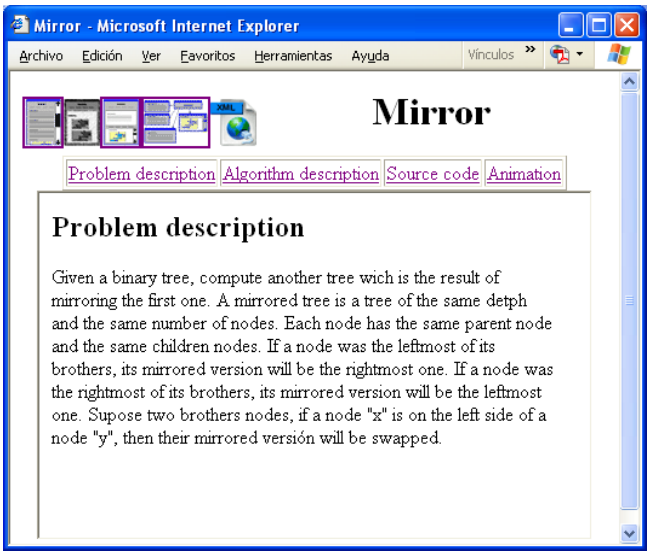

Fig. 4. Animación web con formato *Framed1*

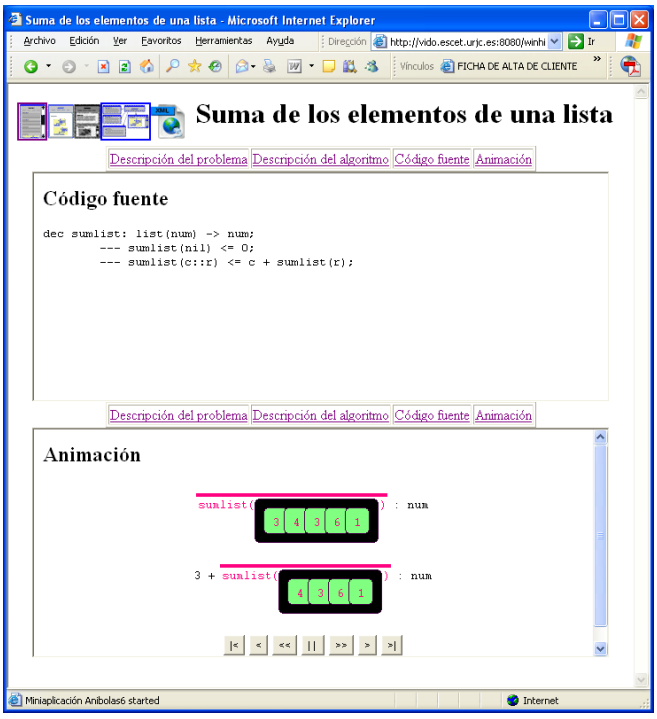

Fig. 5. Animación web con formato *Framed2*

Finalmente, el formato *Star* (ver figura 6) permite al usuario acceder de forma simultánea a tantas secciones como desee. Consiste en una pequeña página con el título y los enlaces a las secciones de la animación. Cada sección se muestra en una página web diferente, siendo el usuario quien adapta las dimensiones de cada página a sus necesidades.

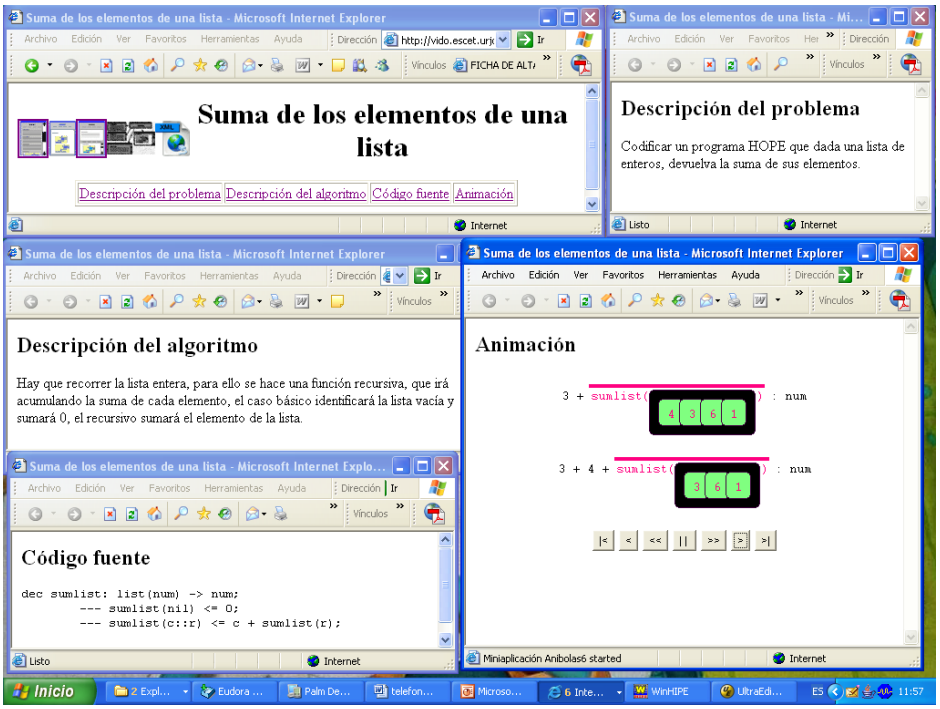

Fig. 6. Animación web con formato *Star*

2.4 Componentes de una animación web

Una vez conocidos los contenidos, el proceso de construcción y el diseño gráfico de las animaciones web, podemos identificar cuáles son sus componentes. Una animación web consiste en toda la información de mantenimiento codificada en un documento XML, los fotogramas que forman la animación, y las páginas web correspondientes a los cuatro formatos de presentación (antiguo, *framed1*, *framed2*, y *star*) Todos estos componentes se guardan en un mismo directorio (ver figura 7)

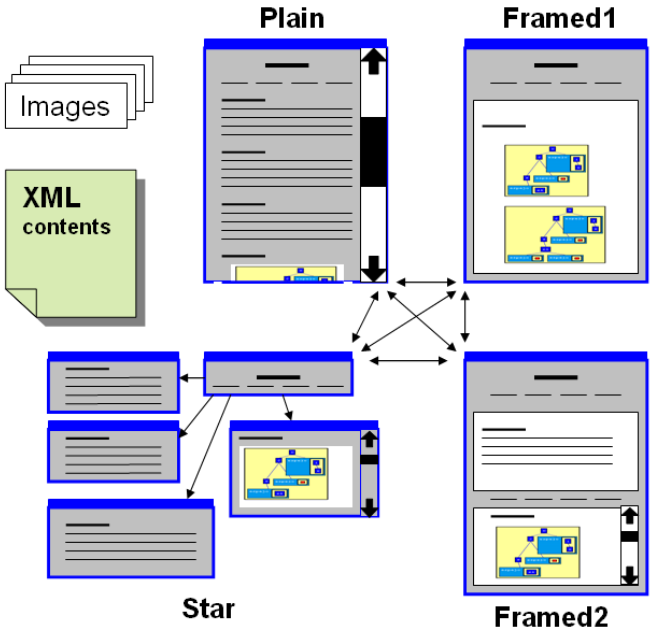

Fig. 7. Componentes de la animación web, con representaciones esquemáticas de los cuatro formatos posibles para la animación web.

4 Evaluación de las animaciones web

Las animaciones de programas funcionales generadas con WinHIPE se han evaluado tanto desde el punto de vista de la satisfacción de los estudiantes como de la eficacia pedagógica. Los detalles de la evaluación de la satisfacción de los estudiantes se pueden encontrar en [6]. Dicha evaluación se realizó en un curso de programación funcional [20]. El resultado de esta evaluación mostró que los estudiantes consideraban la construcción, el uso y el mantenimiento de estas animaciones fácil de aprender y usar sin apenas esfuerzo.

La evaluación de la eficacia pedagógica se ha llevado a cabo en una asignatura dedicada a la enseñanza de algoritmos. En primer lugar los estudiantes recibieron una sesión de dos horas para el aprendizaje del entorno, luego respondieron a un cuestionario sobre la usabilidad del entorno y las animaciones web. Dos semanas después se realizó el experimento. Dicho experimento consistía en responder a un conjunto de preguntas sobre el algoritmo de recorrido en anchura de un árbol binario de izquierda a derecha. Se dividió a los alumnos en dos grupos; excepto un alumno, nadie más conocía previamente el algoritmo. A ambos grupos se les proporcionó una descripción textual del algoritmo. Además, el grupo de control (7 alumnos) tenía disponibles un conjunto de animaciones web sobre el algoritmo. El grupo de

experimentación (6 alumnos) tuvo que desarrollar animaciones sobre el algoritmo. Tras completar cada grupo las tareas asignadas, realizaron el test.

Los resultados muestran que los alumnos del grupo de control adquirieron un conocimiento superficial sobre el algoritmo, pudiendo predecir resultados. Sin embargo el grupo de experimentación adquirió además un conocimiento mucho más profundo, ya que fueron capaces de realizar de forma correcta la versión de recorrido en anchura de derecha a izquierda, cosa que no consiguieron los alumnos del grupo de control. Los alumnos en el grupo de experimentación respondieron de nuevo al cuestionario de opinión sobre el entorno. La mayoría de ellos mantuvo su opinión sobre el entorno: fácil de aprender, útil y fácil usar.

6 Conclusiones y trabajo futuro

Hemos descrito una extensión de un entorno de programación, para generar animaciones web. El punto más importante es que estas animaciones se crean sin apenas esfuerzo por parte del usuario (profesor o alumno). Así, la interacción necesaria con el usuario se minimiza tanto en su construcción como en el mantenimiento.

Hemos ha evaluado el uso de estas animaciones por los alumnos. Los resultados demuestran que los alumnos opinan que el proceso de construcción de estas animaciones es fácil de aprender y usar, aparte de opinar que son útiles como herramienta educativa. Además los alumnos que construyeron animaciones web consiguieron mejores resultados en el que aquellos que únicamente vieron animaciones web, lo que demuestra que el proceso de generación de animaciones web sin esfuerzo es pedagógicamente efectivo.

Como trabajo futuro, se pretende evaluar la usabilidad desde el punto de vista del profesor, haciendo un estudio comparativo con otros enfoques de construcción de estas animaciones web. También se pretende la eficacia pedagógica a largo plazo durante parte de un parcial dedicado exclusivamente a la programación funcional.

Agradecimientos

Este trabajo se ha realizado gracias los fondos proporcionados por el proyecto TIN2004-07568 del Ministerio de Educación y Ciencia del Reino de España. Además, agradezco la ayuda prestada durante las dos sesiones de evaluación a los profesores de la universidad Rey Juan Carlos, Carlos Lázaro Carrascosa y Ángel Velázquez Iturbide, así como la participación en la evaluación de los alumnos matriculados en la asignatura de Diseño y Análisis de Algoritmos del 3er curso de Ingeniería Informática de la Universidad Rey Juan Carlos.

Referencias

1. Bazik, J., Tamassia, R., Reiss, S. P., van Dam, A.: Software visualization in teaching at Brown University. In: Stasko, J.T., Domingue, J., Brown, M.H., Price, B.A. (eds.): Software Visualization. MIT Press, Cambridge MA (1998) 383-398
2. Pareja-Flores, C., Velázquez-Iturbide, J. Á.: Program execution and visualization on the Web. In: A. Aggarwal (ed.), Web-based Learning and Teaching Technologies: Opportunities and Challenges. Idea-Group Publishing, Hershey, PA (2002) 236-259
3. Naps, T., Rößling, G., Almstrum, V., Dann, W., Fleischer, R., Hundhausen, C., Korhonen, A., Malmi, L., McNally, M., Rodger, S. Velázquez-Iturbide, J.Á.: Exploring the role of visualization and engagement in computer science education. SIGCSE Bulletin 35(2) (2003) 131-152
4. Naps, T., Rößling, G., Anderson, J., Cooper, S., Dann, W., Fleischer, R., Koldeofe, B., Korhonen, A., Kuittinen, M., Leska, C., Malmi, L., McNally, M., Rantakokko, J., Ross, R. J.: Evaluating the educational impact of visualization. SIGCSE Bulletin 35(4) (2003) 124-136
5. Naharro-Berrocal, F., Pareja-Flores, C., Velázquez-Iturbide, J.Á.: Automatic generation of algorithm animations in a programming environment. Proceedings of the 30th ASEE/IEEE Frontiers in Education Conference. Stiples Publishing (2000) S2C 6-12
6. Velázquez-Iturbide, J.Á., Pareja-Flores, C., Urquiza-Fuentes, J.: An approach to effortless construction of program animations. Under review
7. Dershem, H.L., McFall, R.L., Uti, N.: Animation of Java linked lists. Proceedings of the 33rd SIGCSE Technical Symposium on Computer Science Education. ACM Press, New York (2002) 53–57
8. Hadlock, F., Williams, J., Wyatt, J., Balasubramanian, D., Bittinger, J., Davis, C., Kesiraju, S., Kolpack, J., Northcutt, J., Sudireddy, S., Tyler, M.: An Internet based algorithm visualization system. Journal of Computing Sciences in Colleges 20(2) (2004) 304 - 310
9. Najork, M.: Web-based algorithm animation. Proceedings of the 38th Conference on Design Automation (2001) 506-511
10. Stern, L., Søndergaard, H., Naish, L.: A strategy for managing content complexity in algorithm animation. Proceedings of the 4th Annual SIGCSE/SIGCUE Conference on Innovation and Technology in Computer Science Education. ACM Press (1999) 127-130
11. Akingbade, A., Finley, T., Jackson, D., Patel, P., Rodger, S.H.: JAWAA: Easy web-based animation from CS0 to advanced CS courses. Proceedings of the 34th SIGCSE Technical Symposium on Computer Science Education. ACM Press (2003) 162-166
12. Ross, R.J., Grinder, M.T.: Hypertextbooks Animated, active learning, comprehensive teaching and learning resources for the Web. In: Diehl, S. (ed.): Software Visualization. Lecture Notes in Computer Science, Vol. 2269. Springer-Verlag, Berlin Heidelberg New York (2001) 269-283
13. Esponda Argüero, M., Rojas, R.: Learning algorithms with an electronic chalkboard over the Web. Advances in Web-Based Learning – ICWL 2004. Lecture Notes in Computer Science, Vol. 3143. Springer-Verlag, Berlin Heidelberg New York (2004) 1-10
14. Korhonen, A., Malmi, L.: Algorithm simulation with automatic assessment. Proceedings of the 5th Annual SIGCSE/SIGCUE Conference on Innovation and Technology in Computer Science Education. ACM Press (2000) 160-163
15. Naharro-Berrocal, F., Pareja-Flores, C., Velázquez-Iturbide, J.Á., Martínez-Santamarta, M.: Automatic Web publishing of algorithm animations. Informatik/Informatique 2 (2001) 41-45
16. Naps, T., Eagan, J., Norton, L.: JHAVÉ: An environment to actively engage students in Web-based algorithm visualizations. 31st SIGCSE Technical Symposium on Computer Science Education. ACM Press (2000) 109–113

17. Rossling, G., Freisleben, B.: Program visualization using AnimalScript. First International Program Visualization Workshop. University of Joensuu Press (2001) 41-52
18. Sutinen, E., Tarhio, J., Teräsvirta, T.: Easy algorithm animation on the Web. Multimedia Tools and Applications, 19(2) (2003) 179-194
19. Nielsen, J.: Designing Web Usability. New Riders Publishing, Indianapolis Indiana USA, 1999
20. Velázquez-Iturbide, J.Á.: The programming languages course for freshmen: Choices and experience. Proceedings of the 10th Annual SIGCSE/SIGCUE Conference on Innovation and Technology in Computer Science Education. ACM Press (2005) in press

Anexo A. DTD de la información de mantenimiento

```
<!ELEMENT principal (content,style,animation)>
<!ELEMENT content (page)>
<!ELEMENT page
(title,MainHead,ProblemDescription,AlgorithmDescription,ProgramCode)>
            <!ELEMENT title (#PCDATA)>
            <!ELEMENT MainHead (#PCDATA)>
            <!ELEMENT ProblemDescription (PHead,PText)>
              <!ELEMENT PHead      (#PCDATA)>
              <!ELEMENT PText      (#PCDATA)>
            <!ELEMENT AlgorithmDescription (AHead,AText)>
              <!ELEMENT AHead      (#PCDATA)>
              <!ELEMENT AText      (#PCDATA)>
            <!ELEMENT ProgramCode (CHead,CText)>
              <!ELEMENT CHead      (#PCDATA)>
              <
!ELEMENT CText      (#PCDATA)>
            <!ENTITY content SYSTEM "contenido.xml">
<!ELEMENT style (root)>
<!ELEMENT root (MainH,ProblemD,AlgorithmD,ProgramC)>
            <!ELEMENT MainH (size,align,color)>
              <!ELEMENT size        (#PCDATA)>
              <!ELEMENT align       (#PCDATA)>
              <!ELEMENT color          (#PCDATA)>
            <!ELEMENT ProblemD (PH)>
              <!ELEMENT PH (size,align,color)>
            <!ELEMENT AlgorithmD (AH)>
              <!ELEMENT AH (size,align,color)>
            <!ELEMENT ProgramC (CH)>
              <!ELEMENT CH (size,align,color)>
            <!ENTITY style SYSTEM "estilo.xml">
<!ELEMENT animation (expression,photogramslist,surconfig)>
<!--expresion(expression),lista de
fotogramas(photogramslist),configuracion de entorno(surconfig)-->
            <!ELEMENT expression                (#PCDATA)>
            <!ELEMENT photogramslist            (#PCDATA)>
            <!ELEMENT surconfig                 (#PCDATA)>
```

Anexo B. Ejemplo de la información de mantenimiento

```
<?XML version="1.0"?>
<!DOCTYPE animcontent SYSTEM "anim.dtd">
<principal>  <content>
    <page>
```

```
      <title>Mirror</title>
      <MainHead>Mirror</MainHead>
      <ProblemDescription>
        <PHead>Problem description</PHead>
        <PText>
          Given a binary tree, compute another tree which is the
result of mirroring the first one.
          A mirrored tree is a tree of the same depth and the same
number of nodes.
          Each node has the same parent node and the same children
nodes.
          If a node is the leftmost of its brothers, its mirrored
version will be the rightmost one.
          If a node is the rightmost of its brothers, its mirrored
version will be the leftmost one.
          Given two brothers nodes, if a node "x" is on the left side
of a node "y", then their mirrored version will be swapped.
        </PText>
      </ProblemDescription>
      <AlgorithmDescription>
        <AHead>Algorithm description</AHead>
        <AText>
          The solution consists in swapping every children nodes in
the tree.
          We will use a recursive function that swaps children nodes,
until this nodes are empty nodes.
        </AText>
      </AlgorithmDescription>
      <ProgramCode>
        <CHead>Source code</CHead>
        <CText>
! Function mirror, compute the symmetric tree of a given one
! -------------------------------------------------------------

data tree == empty ++ node (tree X num X tree);

dec mirror: tree -> tree;
--- mirror (empty) <= empty;
--- mirror node (hi, n, hd) <= node (mirror (hd), n, mirror (hi));
        </CText>
      </ProgramCode>
    </page>
  </content>
  <style>
    <root>
      <MainH><size>6</size> <align>center</align>
<color>black</color></MainH>
      <ProblemD><PH><size>2</size> <align>left</align>
<color>black</color></PH></ProblemD>
      <AlgorithmD><AH><size>2</size> <align>left</align>
<color>black</color></AH></AlgorithmD>
      <ProgramC><CH><size>2</size> <align>left</align>
<color>black</color></CH></ProgramC>
      &style;
    </root>
  </style>
  <animation>
              <expression>
                mirror (node(node(empty,1,node(empty,8,empty)),0,
        node(empty,2,node(node(empty,65,empty),9,empty))));
              </expression>
               <photogramslist>1 2 3 4 5 6 8 10 11
14</photogramslist>
              <surconfig>
                <!--Contents of the .ini file. Ignored to be lots of
data-->
              </surconfig>
  </animation> </principal>
```