Programación Matemática y Software (2014) 6 (1): 39-47. ISSN: 2007-3283

Modelado de prototipos robóticos para robótica en aplicaciones web utilizando X3D/WebGL

Robotic prototype modeling for robotic in web applications using X3D/WebGL

Emmanuel Mendoza-Escobar,[1]* Ignacio Lopez-Martínez,[1] Ignacio Herrera-Aguilar,[1] Ulises Jesús Lopez-Maldonado,[2] Alejandro Domingo Velázquez-Cruz,[2] Sergio David Ixmatlahua-Díaz[1]

[1] Departamento de Posgrado e Investigación, Instituto Tecnológico de Orizaba.
Av. Instituto Tecnológico 852, col. Emiliano Zapata. Orizaba, Veracruz, México
[2] Sensa Control Digital.
Av. Bravo Oriente 93, col. Centro. Torreón, Coahuila, México. CP 27000
* Correo-e: lap.alejandro@acm.org

PALABRAS CLAVE:

robótica, prototipos, web, X3DOOM

RESUMEN

El desarrollo de prototipos robóticos engloba una enorme complejidad, dado que dichos prototipos robóticos deben ser simulados en ambientes virtuales para verificar su correcto funcionamiento antes de implementarlos en un ambiente real. Este trabajo muestra la arquitectura de la aplicación web para la generación de prototipos robóticos. Fue desarrollado mediante el estilo arquitectónico MVC, que incorpora X3D/WebGL en su incorporación. La arquitectura de la aplicación web es un componente de un proyecto de mayor envergadura denominado "TEXCALLI", un proyecto de tesis de la Maestría en Ingeniería en Sistemas Computacionales del Instituto Tecnológico de Orizaba.

KEYWORDS:

robotic, X3DOOM, X3D, prototypes

ABSTRACT

Development of robotic prototypes involves a great complexity because they must be simulated in virtual environments to verify their correct functionality before its implementation in a real environment. This paper shows web application architecture to generate robotic prototypes. It is develop using MVC architectonic style, which incorporates X3D/WebGL on its implementation. Web application architecture is a component of a bigger complexity project called "TEXCALLI", a Masters on Computer Systems thesis' project at the Technological Institute of Orizaba.

Recibido: 20 de septiembre de 2012 • **Aceptado:** 15 de agosto de 2013 • **Publicado en línea:** febrero de 2014

1 INTRODUCCIÓN

La aplicación de nuevas tecnologías en la enseñanza es cada vez más habitual. Es común ver que un profesor publique en una página web el temario de sus asignaturas, los apuntes e incluso los exámenes ya realizados. Ya existen en internet las llamadas universidades virtuales que permiten al alumno realizar cualquier tipo de estudios en un ambiente virtual, sin una sede física donde se impartan las sesiones. La mayoría sólo permite interactuar con la institución a través de páginas web en dos dimensiones, sin considerar recursos tridimensionales que puedan favorecer el aprendizaje en las distintas asignaturas.

La aparición de internet como medio de comunicación permite el acceso a la información de manera sencilla y rápida. La mayor parte de esta información se encuentra contenida en las páginas web, que suelen presentar texto e imágenes en dos dimensiones. El mundo real es tridimensional, por lo que al reducir el "mundo" web a sólo dos dimensiones se está perdiendo información, de ahí la conveniencia de la integración de una tercera dimensión que permita, por ejemplo, recorrer las instalaciones de un museo o de una universidad hasta llegar a la información que interese al visitante.

A. Rowell indica: "La realidad virtual es una simulación interactiva por computador desde el punto de vista del participante, en la cual se sustituye o se aumenta la información sensorial que recibe". La simulación que hace la realidad virtual se puede referir a escenas virtuales, crear un mundo virtual (que sólo existe en el ordenador) de lugares u objetos que existen en la realidad.

En los institutos tecnológicos de nuestro país es cada vez más difícil que las instituciones cuenten con la infraestructura material y los recursos humanos suficientes para soportar de manera eficiente la formación de nuevos ingenieros e investigadores [1], particularmente en áreas como la robótica, que tiene un componente tecnológico fuerte, y esto se debe a los altos costos de inversión y operación y al constante deterioro de los equipos, que demandan un permanente mantenimiento [2]. Por ello, una de las estrategias complementarias para la formación y para el aprovechamiento de instalaciones es implementar laboratorios virtuales [3, 4].

Dado que un sistema de realidad virtual permite explorar la escena de forma interactiva y ver el mundo virtual desde cualquier punto de vista, es necesario disponer de una representación geométrica 3D de este mundo.

Según Jacyntho y colaboradores y Ziemer [5, 6], la tendencia en el uso masivo de las tecnologías multimedia en nuestra vida cotidiana tiene también un impacto sobre la generación de prototipos robóticos industriales y su cadena de valor a partir de la digitalización, procesamiento y presentación dentro de entornos web.

La experiencia directa de la manipulación de los artefactos tecnológicos enseña al estudiante mucho más sobre las tecnologías que hacerlo solamente mediante las descripciones o modelos de los sistemas de información. Este proyecto se centra en el procedimiento para la elaboración de prototipos robóticos por internet usando X3D/WebGL.

2 TRABAJOS RELACIONADOS

En la actualidad existen en el mercado diversas tecnologías para desarrollar prototipos robóticos, ninguna engloba los conceptos fundamentales de una arquitectura web para el desarrollo de prototipos robóticos [13]. A continuación se presentan cuatro herramientas que contemplan los principios básicos de una arquitectura para el desarrollo de prototipos robóticos.

Microsoft Robotics Studio (aplicación desarrollada por Microsoft) es para el desarrollo de robots nuevos incluso antes de construirlos físicamente. La característica más sobresaliente de esta tecnología es que puede simular exactamente la misma aplicación que se ejecutará en el robot real. Pero con ciertos límites, ya que el mundo real es complejo y está lleno de imprevistos y de ruido.

ROBO Pro Software de Fischertechnik permite programar al robot ROBO Mobile Set, de la misma compañía. Su programación se basa en diagramas de flujo, descartando el sistema de movimientos por modelos matemáticos como son geométrico y cinemático. El código de salida que se genera en la aplicación se envía a la unidad de procesamiento por medio del USB o vía serial (RS-232).

Mindstorms de Lego, definida por Seung y colaboradores [12], permite programar al robot Mindstorms NTX de Lego. Está basado en el programa LabView, de la empresa National Instruments. LabView es un entorno de programación gráfica usado por miles

de ingenieros e investigadores para desarrollar sistemas sofisticados de medida, pruebas y control usando iconos gráficos e intuitivos y cables que parecen un diagrama de flujo. LabVIEW ofrece una integración incomparable con miles de dispositivos de *hardware* y brinda cientos de bibliotecas integradas para análisis avanzados y visualización de datos. Esta aplicación año con año se tiene que actualizar para mejorar su desempeño con base en los nuevos complementos que se van integrando. Los sistemas operativos soportados son Windows en todas sus versiones, GNU/Linux, Mac y UNIX.

De acuerdo con Thomaz y colaboradores [7], RoboEduc es el *software* pedagógico para soportar robots educativos y se enfoca en alumnos de educación primaria. Trabaja con robots Lego Mindstorms RCX de manera remota, utilizando un lenguaje de programación textual; para hacer funcionar la aplicación, ésta debe estar instalada en nuestra computadora y esperar a que sean generadas todas las sentencias basadas en el proyecto que se esté realizando. Contiene un pequeño módulo de visualización que muestra cómo funciona el prototipo en un ambiente virtual simulado.

De todas las aplicaciones anteriores ninguna hace mención de los modelos matemáticos utilizados en robótica para llevar a cabo las tareas encomendadas en la industria del desarrollo de prototipos robóticos. En el caso de LabVIEW es una aplicación propietaria y necesita actualización cada año para tener la última versión estable del producto. Es por eso que se propone en esta investigación una arquitectura basada en la tecnología web que contiene lo necesario para llevar a cabo el diseño de prototipos robóticos mediante el uso del navegador web.

3 DISEÑO Y ARQUITECTURA

El modelo ha sido diseñado tomando un enfoque global de orientación en servicios. Teniendo en cuenta el alcance esperado de la propuesta y el contexto al que se dirige, se concreta el enfoque de orientación en servicios en un conjunto de capas funcionales que satisfacen los requisitos de diseño de la solución.

Como se observa en la figura 1, la pila está formada por tres capas, caracterizadas de acuerdo con un sentido de desarrollo abajo-arriba, de manera que las capas inferiores aportan recursos genéricos que la capa superior concentra de acuerdo con sus necesidades particulares. Estas capas se podrían agrupar según su naturaleza: las dos inferiores (control y modelo) están diseñadas para satisfacer las necesidades de una aplicación web, de forma que la capa superior (vista) se adapte a los recursos que éstas suministran.

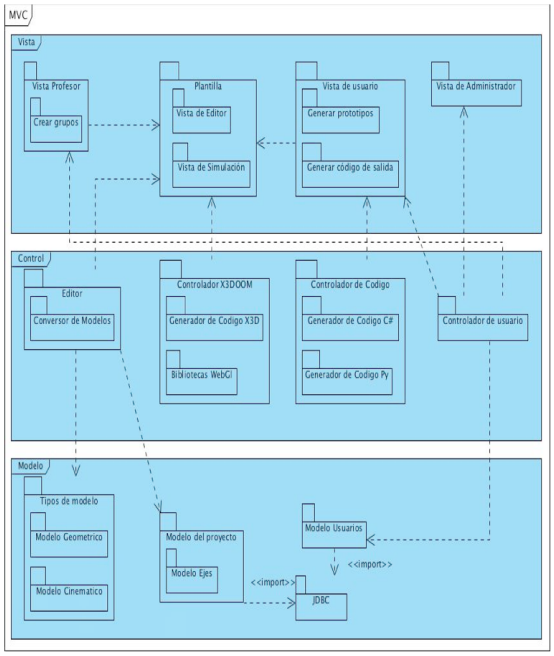

Figura 1. Vista completa de la arquitectura

3.1 Capa de vista

Dentro de esta capa, que es la única visible para el usuario, este se responsabiliza de la presentación de contenidos. De acuerdo con su caracterización, permite que las herramientas desarrolladas sobre esta solución sean utilizadas de forma homogénea en un entorno de interacción formado por un conjunto heterogéneo de dispositivos de consulta. El módulo "vista usuario" hace uso de la "vista plantilla", la cual se encarga de mostrar un editor de prototipos por desarrollar y un renderizado donde se muestra el elemento con el cual se está trabajando en un entorno en 3D dentro de la aplicación web.

La "vista plantilla" es la más utilizada durante la creación de los prototipos robóticos. En ella se pueden visualizar las manipulaciones como son color, forma, dimensiones y tipo de movimiento que realiza con base en el modelo matemático que se empleará.

En la "vista administrador" se encuentra el manejo de las contraseñas para cada usuario; el encargado de este apartado puede activar o

desactivar a un usuario dentro del sistema para trabajar dentro de su lista de proyectos.

El profesor cuenta con una "vista profesor". Ésta da la facilidad de administrar grupos en los cuales puede agregar o modificar usuarios mediante grupos compartidos, y éstos a su vez ayudan a actualizar los diseños de los prototipos robóticos o simplemente observar los avances que se han realizado en éstos.

3.2 Capa de control

La gestión del proceso de interacción entre los usuarios y el sistema se delega sobre el controlador. Esta capa es responsable de gestionar las peticiones que intercambian usuarios con la aplicación. Actúa como mediador entre la vista y el modelo, interpretando las peticiones por medio de clases que definen las peticiones de usuario. En la figura 1 se observan cuatro módulos que son editor, controlador X3DOOM, controlador de código y controlador de usuarios.

El paquete "controlador editor" contiene una serie de paquetes con base en las peticiones del usuario (tipo de modelo y número de ejes del prototipo), y se encarga de realizar las peticiones necesarias para mostrar la información correcta con base en el trabajo que se realice dentro del editor.

Dentro de esta capa se encuentra el módulo "controlador de código" y "controlador de usuario". El primero se encarga de generar el código de salida de cada prototipo realizado dentro de la aplicación. Por el momento sólo se manejan dos lenguajes de salida, el primero es C# y el segundo es Python. Por último, el módulo controlador de usuario se encarga de las peticiones que el usuario hace con referencia a sus trabajos, como lo es el último movimiento realizado a cada prototipo y las aportaciones llevadas a cabo en cada uno de ellos, así como sus datos generales.

En el interior del módulo controlador X3DOOM se encuentra una arquitectura propia del *framework* que ayuda a realizar la salida de escenarios en 3D dentro de la aplicación web.

3.3 Arquitectura X3D

En la primera sección se menciona que hay posibilidades de llevar contenidos 3D a la web. Sin embargo las tecnologías como Java3D, JOGL, JavaFX, Silverlight, Flash, PaperVision, O3D hacen uso de frameworks para su funcionamiento, sobrecargando la memoria para poder visualizar el contenido realizado utilizando su propia arquitectura para poder llevarlo a cabo.

La mayoría de los actuales sistemas basados en complementos y estándares ofrecen interfaces pequeñas para su ejecución y su modelo de aplicación. Estos complementos incluyen tiempo de ejecución de sus propios sistemas que controlan toda la visualización, la interacción, la comunicación y los problemas de distribución interna. Un nuevo modelo de integración debe evitar nuevos complementos o interfaces de sistema, y ser accesible a través del modelo estándar de objetos de documentos (DOM, por sus siglas en ingles). Éste fue el mismo modelo para la primera generación de visores SVG en años atrás. Actualmente, SVG es una parte integrada de un navegador web moderno y los elementos SVG pueden ser manipulados directamente utilizando la misma tecnología DOM usada por el contenido HTML.

Flash y Silverlight están basados en complementos y todavía tienen poco apoyo para el 3D, lo que limita el desarrollo de aplicaciones de efectos visuales.

Por su parte, X3D tiene una completa escena gráfica de los modelos, ya que cuenta con algunos mecanismos de importación DOM. Las aplicaciones X3D también necesitan complementos para realizar *scripting*, animación y renderizado cuando es necesario. El desarrollo reciente integra JavaScript/ OpenGL, propuesto por Mozilla y Khronos en Web 3D Consortium [9], es una abstracción de Niza que se ejecuta sin complementos; pero en realidad necesita una extensión interna de *scene -graph* (escena gráfica), como X3D, para un perfecto desempeño.

Por lo tanto, se propone una *scene-graph* más abstracta de la capa que debe ser directamente asignada a los elementos DOM. El modelo propuesto es una estructura de visualización de datos bien establecida y capaz de manejar grandes conjuntos de datos de manera eficiente. Y aun más importante, permite que el desarrollo de los prototipos robóticos dentro de las escenas de una manera declarativa similar a otras tecnologías web como SVG o XHTML y no sólo una interfaz de programación como OPENGL. Las bibliotecas como Web 3D Consortium [10] tratan de integrar la declaración DOM con un renderizado basado en CANVAS, que es representado para separar tanto los objetos que aumenta como la complejidad de manipulación.

Por lo tanto, se presenta una solución más integrada y sencilla: la solución basada en X3D y XHTML (utilizada para la arquitectura web que aquí se propone). Se utiliza como modelo de escenario gráfico X3D por tres razones. En primer lugar se trata de una norma ISO madura y establecida que ya define una codificación XML. En segundo lugar, como se mencionó en la sección anterior, es la primera arquitectura para generación de prototipos robóticos en la web. Por último, pero no menos importante, existe una interfaz de árbol para mecanismos de contenido en movimiento y cambiantes de DOM.

El método elegido de XHTML se utiliza en conjunto con X3D para declarar escenas 3D. Pero una vez más, la arquitectura descrita anteriormente hace uso de la integración del DOM muy similar para acceder al contenido de las etiquetas en XHTML. La actual situación lo hace también, al igual que W3C [11], al desacoplar el plugin-DOM-object y el DOM-data-elements de datos que conducen a una confusión todavía mayor, ya que ambos están basados en XML.

En este trabajo se propone ampliar esta arquitectura para integrar más modelos o prototipos robóticos de diferente índole en su integración para producir las imágenes no en un plugin adicional, sino más bien en algo similar a como funciona en la actualidad SVG. Para facilitar la integración y el desarrollo del modelo sólo admite un subconjunto bien definido de la completa especificación y no excluyendo a X3D de toda la secuencia de comandos y distribución de la ejecución de X3D que comparte los datos con el árbol-DOM. Los nodos X3D se integran directamente en los elementos DOM.

El subsistema de X3D se utiliza principalmente para hacer o generar la escena. Toda la manipulación de los prototipos robóticos en el contexto de ejecución DOM se hará utilizando el estándar de las interfaces DOM basadas tanto en secuencias de comandos del navegador como en documentos convencionales de DHTML.

De este modo, se puede mejorar el estado actual de la integración de 3D en la web al no aplicar una tecnología más para simular los prototipos robóticos por desarrollar. La clave consiste en reducir el sistema 3D para el componente de visualización y para tomar prestada la tecnología web, por ejemplo: dinámica, distribución, seguridad y secuencia de comandos.

Un ambiente "delgado" para un único punto de fusión de los entornos tradicionales de acceso a los mundos en 3D que adoptan nuevas estrategias basadas en un modelo adecuado y prolongado para evitar el paradigma "un nuevo plugin".

La arquitectura propuesta hace uso de la arquitectura de X3DOM, que como se observa en la figura 2, actúa como un conector para el contenido XHTML y en el contenido y mundos generados en X3D. El objetivo es apoyar el contenido XHTML, que incluye datos de X3D en un espacio de nombres separados en el interior del árbol DOM del navegador.

La base de este diseño es la especificación de XHTML [10], ya que utiliza X3D para este propósito. La arquitectura de X3D intenta explorar este modelo proporcionando el enlace a los desaparecidos entre ambos mundos.

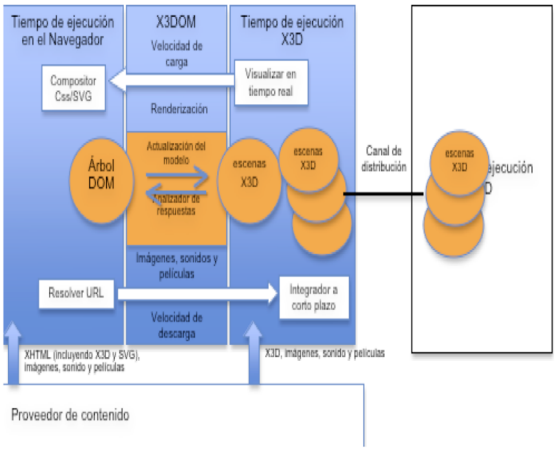

Figura 2. X3DOM, descripción que muestra los componentes de la propuesta de integración del *framework* [12]

3.4 Actualización de prototipos

Todas la actualizaciones en tiempo de ejecución se verán reflejadas en el árbol DOM. Al igual que crea, dispone y cambia los elementos DOM, debe reflejarse en el árbol X3D utilizando el adaptador final. Esto incluye que los prototipos robóticos (elementos DOM) que representan directamente a los nodos o ejes del mismo, son estructuras adicionales X3D.

3.5 Capa de modelo

Se define esta capa en torno a los provistos dentro de cada módulo que lo compone. En la figura 1 se observan tipos de modelo, modelo del proyecto y

modelo usuario. El primero contiene las ecuaciones necesarias para llevar a cabo cada método matemático. En este caso para la arquitectura se consideran dos, los cuales son modelo geométrico y modelo cinemático.

El módulo denominado "modelo del proyecto" contiene partes cruciales para la generación de los prototipos, como son los números de ejes, las características intrínsecas y extrínsecas del prototipo, así como su descripción y motivo por el cual se desarrolla dicho proyecto. El último módulo, "modelo usuario", contiene una serie de actividades que van desde los datos generales, como son: nombre, apellidos, ciudad, entre otros, para utilizarlos dentro de cada prototipo del proyecto que se va generando.

Estos dos últimos módulos hacen uso de un cuarto módulo más que es el conector JDBC encargado de la conexión entre la aplicación web y el motor de la base de datos.

4 IMPLEMENTACIÓN DEL SISTEMA Y LA ARQUITECTURA

Ahora que se ha descrito el contenido de la arquitectura, se crea una aplicación en la cual se crearon los tres paquetes propuestos en la arquitectura de referencia. A continuación se analiza el código utilizado para la construcción del robot antropomórfico en un archivo XHTML:

```
<X3D>
 <Scene>
  <Viewpoint/>
   <Background/>
    <Transform DEF='Principal'>
         <Transform translation="2 0 0"
rota-tion="1 0 0 1.75">
      <Shape>
       <Appearance>
        <Material diffuseColor='red'/>
       </Appearance>
       <Box size='8 1 8' solid='true'/>
      </Shape>
       <Transform id='Eje01' DEF="Eje01"
rotation="1 0 0 1.75" translation="0 0
1.5" >
      <Transform >
       <Transform translation="0 3 0" >
        <Shape>
         <Appearance>
          <Material diffuseColor='blue'/>
         </Appearance>
         <Box size='2 9 2' solid='true'/>
        </Shape>
        <Transform translation="0 3 2"  >
        <Transform id='Eje02' DEF="Eje02"
rotation="0 0 1 0" >
         <Transform translation="0 3 0" >
          <Shape>
           <Appearance>
         <Material diffuseColor='black'/>
           </Appearance>
          <Box size='2 9 2' solid='true'/>
           </Shape>
           <Transform translation="0 1 0">
                    <Transform id='Eje03'
DEF='Eje03' translation="0 3 -1.8">
              <Shape>
               <Appearance>
              <Material diffuseColor='0.1
0.5 1'/>
               </Appearance>
                   <Sphere DEF='Esfera01'
radi-us='1' solid='true'/>
              </Shape>
              <Transform translation="0 2
-1.8" >
               <Shape>
                <Appearance>
                     <Material diffuseCo-
lor='yellow'/>
                </Appearance>
                     <Cylinder radius='1'
height='6'   bottom='true'   side='true'
top='true' sol-id='true'/>
               </Shape>
              </Transform>
             </Transform>
            </Transform>
           </Transform>
          </Transform>
         </Transform>
        </Transform>
       </Transform>
      </Transform>
     </Transform>
    </Transform>
 </Scene>
</X3D>
```

En el código anterior se aprecia la escena completa de dicho prototipo, dentro de las etiquetas `<Body></Body>` de XHTML se puede apreciar su despliegue en el navegador que se utilice. Para que el prototipo pueda tener movimiento propio se deben agregar las siguientes etiquetas después de la última etiqueta `</Transform>` para quedar de la siguiente manera:

```
<TimeSensor DEF='TIMER' cycleInter-val='10'
loop='true'/>
<PositionInterpolator DEF='ROTOR_A' key='0
```

```
0.25 0.45 1' keyValue='0 0 0 0 1 0 0 -5 0 0
0 0'/>
<ROUTE fromNode='TIMER' from-Fiel-
d='fraction_changed' toNode='ROTOR_A'
toField='set_fraction'/>
<ROUTE fromNode='ROTOR_A' from-
Field='value_changed' toNode='Eje01'
toField='translation'/>
<PositionInterpolator DEF='ROTOR_B' key='0
0.30 1' keyValue='0 -1 0 0 -5 0 0 -1 0'/>
<ROUTE fromNode='TIMER' from-Fiel-
d='fraction_changed' toNode='ROTOR_B'
toField='set_fraction'/>
<ROUTE fromNode='ROTOR_B' from-
Field='value_changed' toNode='Eje02'
toField='translation'/>
<PositionInterpolator DEF='ROTOR_C' key='0
0.30 1' keyValue='0 2 0 0 -4 0 0 2 0'/>
<ROUTE fromNode='TIMER' from-Fiel-
d='fraction_changed' toNode='ROTOR_C'
toField='set_fraction'/>
<ROUTE fromNode='ROTOR_C' from-
Field='value_changed' toNode='Eje03'
toField='translation'/>
```

Al unir los dos códigos se obtiene un objeto con características homogéneas, que se puede visualizar y entender su comportamiento, para posteriormente ver cómo realiza sus movimientos fluidos sin la necesidad de algún *plugin* adicional al navegador que se utilice.

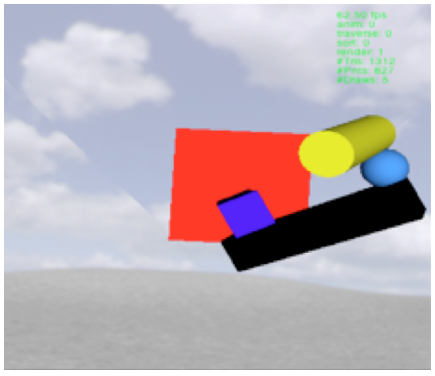

Figura 3. Robot antropomórfico con estructura simple utilizando X3D/WebGL en el navegador Firefox

En la figura 3 se muestra el prototipo de robot antropomórfico con el código descrito visualizado con Firefox 12.0.

Las desventajas encontradas en esta arquitectura es que sólo funciona en un ambiente web, por lo cual se necesita tener una conexión a internet para su funcionamiento. Los usuarios deben contar con ello para utilizarla y manejar los prototipos que provee la aplicación.

5 CONCLUSIÓN

La arquitectura presentada alcanza las características de calidad y los objetivos deseados mediante una estructura de capas que facilita el desarrollo de prototipos robóticos en un entorno web. Estos prototipos pueden ser construidos a partir de una perspectiva genérica que facilita su caracterización en específico, sin que ello implique pérdida en alguna funcionalidad deseada. Se están revisando las herramientas de Matlab en función de robótica. El apoyo del *framework* X3D/WebGL que utiliza muchos también etiquetas para su simplificación en el desarrollo de la simulación hace que sea más fácil de interpretar la producción de dichos robots, pues es también tecnología que funciona con DOM que facilita el uso de movimientos mediante JavaScript para implementarlos posteriormente en métodos más complejos o en prototipos que lo requieran. Los ejes cartesianos se encuentran implícitos dentro de la simulación dando una experiencia de navegación bastante aceptable. La parte de documentación de cada prototipo robótico realizado dentro de la aplicación corresponde a cada usuario que ha generado dicho artefacto para sugerir o tener un mayor control sobre lo que se está realizando.

6 TRABAJO FUTURO

Se pretende concretar en específico cada uno de los componentes del sistema de cara a la funcionalidad con la que han sido presentados. Una vez conseguido esto, se representará cada uno de estos conceptos sobre el modelo arquitectónico que define la capa de cada nivel. En la capa "vista" ya se tiene el controlador necesario para hacer funcionar la "vista editor", lo cual ya se está implementando. La explotación final de esos elementos sobre ejemplos de la vida real será el paso siguiente por considerar, con el fin de validar mediante experiencias reales la viabilidad de esta propuesta. Por último, cabe mencionar que cuando se tenga la aplicación los modelos que se generen tendrán un código de salida que será para los lenguajes C# y Phyton para su implementación en dispositivos físicos.

REFERENCIAS

1. Web 3D Consortium. (2008). X3D. http://www.web3d.org/x3d/specifications/
2. Garlan, D., Shaw, M. (1993). An introduction to software architecture. *Advances in software engineering and knowledge engineering,* 1, 1-40.
3. Jacyntho, M.D., Schwabe, D., Rossi, G. A. (2002). Software architecture for structuring complex web applications. *Journal Web Engineering*, 1(1), 37-60.
4. Taylor, R. N., Medvidovic, N., Dashofy, E. M. (2009). Software architecture: Foundations, theory, and practice. Wiley Publishing.
5. Jacyntho, M. D. Schwabe, D. Rossi, G. (2002). A software architecture for structuring complex web applications. *Journal Web Engineering*, 1(1), 37-60.
6. Ziemer, S. (2002). An architecture for web applications. essay in dif 8914 distributed information systems. Technical report, Norwegian University of Science & Technology.
7. Thomaz, S., Akynara Aglaé, Fernades, C., Pitta, R., Azevedo, S., Burlamaqui, A. Silva, A., Gonzalvez, L. M. G. RoboEduc: A pedagogical tool to support educational robotics.
8. Seung Han Kim, Jae Wook Jeon. Programming LEGO mindstorms NTX with visual programming. IEEE Conferences, ICCAS 07. 2468-2472. International Conference on DOI: 10.1109/ICCAS.2007.4406778.
9. Web 3D Consortium, K. G. Formal liaison with khronos group. http://www.web3d.org/press/detail/ web3d enters formal liaison with khronos group/
10. Web 3D Consortium. Scene access interface(sai), iso/iec cd 19775-2 ed. 2:200x. http://www.web3d.org/x3d/specifications/ISO-IEC-CD-19775- 2.2-X3D-SceneAccessInterface/
11. W3C. Namespaces in xml. W3C Consortium. http://www.w3.org/TR/REC-xml-names/
12. Arnaud, R., Parisi, T. (2010). Developing web applications with collada and x3d. http://www.khronos.org/collada/presentations/ Developing Web Applications with COLLADA and X3D.pdf

Acerca de los autores

Emmanuel Mendoza Escobar es Maestro en Sistemas Computacionales (2012) e Ingeniero en Sistemas Computacionales (2007), por el Instituto Tecnológico de Orizaba. Sus áreas de investigación son inteligencia artificial, robótica, mecatrónica, sistemas expertos, modelado y simulación en entornos hetereogenios, desarrollo ágil de proyectos de *software* y aplicaciones en web. Ha colaborado con diferentes instituciones de educación superior y ha trabajado en diferentes empresas del sector privado.

Ignacio López Martínez es Maestro en Redes y Telecomunicaciones (2007) por la Universidad Cristóbal Colón de Veracruz, y Licenciado en Informática por el Instituto Tecnológico Orizaba (1999). Profesor investigador de tiempo completo y perfil deseable Promep. Coordinador de servicios de red de la División de Estudios de Posgrado e Investigación del Instituto tecnológico de Orizaba.

Ignacio Herrera Aguilar es profesor del Instituto Tecnológico de Orizaba. Realizó su Doctorado en Sistemas Automáticos en la Université Paul Sabatier Toulouse III, en Toulouse, Francia. Sus áreas de investigación son la Robótica de Servicio, Visión Artificial, Rehabilitación y Mecatrónica.

Alejandro Domingo Velázquez Cruz es Licenciado en Administración Pública por la Universidad Abierta de San Luis Potosí. Sus intereses incluyen la administración y gestión de los procesos empresariales mediante la utilización de sistemas computacionales. Ha ocupado diversos cargos en la Secretaría de Desarrollo Social, el Instituto Mexicano del Seguro Social y la iniciativa privada.

Sergio David Ixmatlahua Díaz es Maestro en Sistemas Computacionales por el Instituto Tecnológico de Orizaba, Veracruz. Durante sus estudios de maestría realizó una estancia profesional en la Facultad de Informática de la Universidad Politécnica de Madrid, España. Ha participado en congresos internacionales y nacionales como ponente presentando artículos relacionados con la ingeniería de *software* y desarrollo web. Actualmente se desempeña como profesor en el Instituto Tecnológico de Iztapalapa en la ciudad de México, DF.

Ulises Jesús López Maldonado se graduó como Ingeniero en Sistemas Computacionales en el Instituto Tecnológico de Orizaba, Veracruz. Sus intereses incluyen el desarrollo de sistemas desde el enfoque del paradigma orientado a componentes, programación orientada a aspectos, ingeniería de *software* y *software* educativo. Actualmente se desempeña como docente en el Instituto Tecnológico de Iztapalapa.