WEB3D. ANÁLISIS COMPARATIVO DE VRML, JAVA3D Y X3D

E. Jiménez Macías
Departamento de Ingeniería Eléctrica. Área de Ingeniería de Sistemas y Automática.
Universidad de La Rioja. C/Luis de Ulloa 20 CP-26004 Logroño (La Rioja)
e-mail: emilio.jimenezm@die.unirioja.es

F. Sanz Adán, J. Santamaría Peña, E. Martínez Cámara
Departamento de Ingeniería Mecánica. Área de Expresión Gráfica en la Ingeniería.
Universidad de La Rioja. C/Luis de Ulloa 20 CP-26004 Logroño (La Rioja)
e-mail: {felix.sanz;jacinto.Santamaría}@dim.unirioja.es; eduardo.martin@alum.unirioja.es

M. Pérez de la Parte
Departamento de Ingeniería de Sistemas y Automática. Área de Ingeniería de Sistemas y Automática.
Universidad de Sevilla. Avda. de los Descubrimientos SN, CP-41092 Sevilla
e-mail: mpparte@cartuja.us.es

Resumen

El término Web3D hace referencia a cualquier lenguaje de programación, protocolo, formato de archivo o tecnología que pueda ser usado para la creación y presentación de universos tridimensionales interactivos a través de Internet. Tres de esos lenguajes se incluyen como estándares abiertos: Virtual Reality Modeling Language (VRML), Java3D y X3D (Extensible 3D). En este artículo se hace una comparativa de estas tres distintas tecnologías no propietarias destinadas a la representación de escenarios tridimensionales en Internet. La comparativa se realiza sobre la experiencia de dos proyectos llevados a cabo por parte de los autores, pertenecientes al Grupo de Simulación de la Universidad de La Rioja. Esos proyectos consisten concretamente en un Navegador Virtual Integral por el Territorio de La Rioja, basado en ortofotos , y un entorno multiplataforma de simulación robótico en escenarios virtuales en internet.

Palabras Clave: 3D, Realidad Virtual, Web3D, VRML, Java 3D, X3D.

1 INTRODUCCIÓN

El término Web3D hace referencia a cualquier lenguaje de programación, protocolo, formato de archivo o tecnología que pueda ser usado para la creación y presentación de universos tridimensionales interactivos a través de Internet [4].

Dentro de estos lenguajes para programar universos virtuales se incluyen como estándares abiertos: VRML (Virtual Reality Modeling Language), Java3D y X3D (Extensible 3D). También existen y se están desarrollando un gran número de soluciones a nivel propietario para satisfacer las necesidades con-

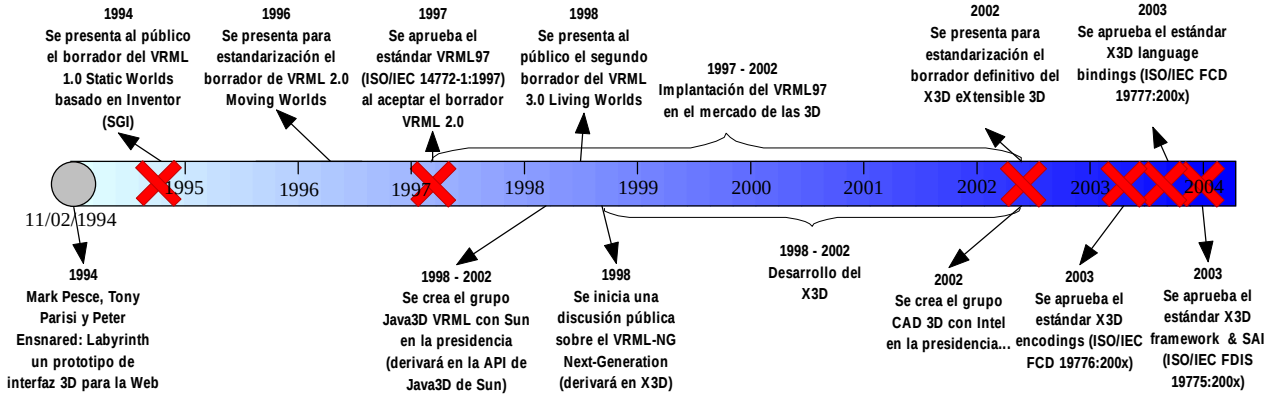

Figura 1: Cronograma del Web3D

cretas de los clientes, generalmente, encaminadas al comercio y entretenimiento electrónico: Cult 3D, Pulse 3D, ViewPoint, etc.

A lo largo de este artículo se va intentar estudiar y comparar las características fundamentales de las distintas tecnologías no propietarias destinadas a la representación de escenarios tridimensionales en Internet. Las tecnologías que se van a incluir en este estudio son:

- VRML (Virtual Reality Modeling Language).
- X3D (Extensible 3D).
- Java3D.

La Figura 1 muestra un cronograma de la evolución de Wed3D desde 1994 hasta nuestros días.

2 CARACTERÍSTICAS GENERALES

2.1 VRML

VRML es un formato de archivo que permite la creación de objetos y mundos tridimensionales interactivos. El estándar VRML fue creado y desarrollado por el VRML Consortium. En principio el VRML Consortium era una organización sin animo de lucro centrada exclusivamente en el desarrollo y promoción de VRML como estándar 3D en Internet. VRML apareció en escena en 1994 y llegó a ser la primera tecnología reconocida oficialmente por la ISO (International Organization for Standarization) como estándar para la creación, distribución y representación de elementos 3D a través de Internet.

VRML fue diseñado para cumplir con los siguientes requerimientos básicos [1]:

- Habilitar la posibilidad del desarrollo de programas para crear, editar y mantener archivos VRML, además de programas para la importación y exportación del formato VRML a otros formatos gráficos tridimensionales (Authorability).
- Aportar la capacidad de utilizar, combinar y reutilizar objetos dinámicos tridimensionales dentro de un mismo mundo VRML (Composability).
- Incorporar la capacidad de crear nuevos tipos de objetos no definidos específicamente como parte de VRML (Extensibility).
- Abrir la posibilidad de que sea implementado en una amplia variedad de sistemas presentes en el mercado (Implementable).
- Resaltar la importancia del funcionamiento interactivo en una amplia variedad de plataformas existentes (Performance).
- Permitir la creación de mundos tridimensionales de cualquier tamaño (Scalability).

El VRML es un lenguaje jerárquico de marcas que usa Nodos, Eventos y Campos para modelar realidades virtuales tanto estáticas, como dinámicas.
Los Nodos, que se usan para instanciar alguna de las 54 primitivas del lenguaje, no son más que una colección de Campos que contienen los atributos básicos de la primitiva.

Los Campos (Fields) son los atributos que definen el comportamiento de la primitiva, con la excepción los campos especiales (eventIn y eventOut) que permiten enviar y recibir eventos a otros Campos.

Mediante estos Campos especiales y el comando ROUTE, se puede controlar el flujo de Eventos, encaminando el efecto de una acción entre múltiples objetos para animar una escena o simplemente pasar información a esos objetos.

Cabe indicar que VRML es “case sensitive” y nunca aceptará un código que no respete el uso preciso de mayúsculas y minúsculas.

La Figura 2 muestra una captura de un simulador de Vuelo Virtual Integral por La Rioja, desarrollado por algunos de los autores de este artículo, que está basado en VRML y en ortofotos sobre el terreno.

Figura 2: Ejemplo de escena VRML

Las primitivas del VRML se agrupan según función, en nueve colecciones distintas:

- Agrupación de nodos
- Grupos especiales

- Sensores
- Geometría
- Propiedades de geometría
- Apariencia de la geometría
- Interpoladores
- Nodos excluyentes
- Nodos comunes

2.2 JAVA3D

El API de Java 3D™ es un conjunto de clases para crear aplicaciones y applets con elementos 3D. Ofrece a los desarrolladores la posibilidad de manipular geometrías complejas en tres dimensiones. La principal ventaja que presenta este API 3D frente a otros entornos de programación 3D es que permite crear aplicaciones gráficas 3D independientes del tipo de sistema [2].

Es parte de la API JavaMedia y por tanto puede hacer uso de la versatilidad del lenguaje Java, así como soportar un gran número de formatos como VRML, CAD, etc.

Java 3D es un conjunto de clases, interfaces y librerías de alto nivel que permiten aprovechar la aceleración gráfica por hardware que incorporan muchas tarjetas gráficas, ya que las llamadas a los métodos de Java 3D son transformadas en llamadas a funciones de OpenGL o Direct3D. Aunque tanto conceptualmente como oficialmente Java 3D forma parte del API JMF, se trata de unas librerías que se instalan independientemente del JMF.

Aunque Java3D no soporte directamente cada posible necesidad 3D, sí proporciona la capacidad de implementarlo a través del código java. En otros casos se provee de cargadores (de VRML, X3D...) que traducen ficheros de ese formato en objetos apropiados en Java3D.

Proporciona una interface de programación de alto nivel basado en el paradigma orientado a objetos, lo que permite obtener todas las ventajas de este: desarrollo simple y rápido de aplicaciones.

La Figura 3 muestra una captura de un entorno multiplataforma de simulación robótico en escenarios virtuales en internet, basado en Java 3D, si bien es compatible con VRML para la portabilidad de escenarios.

La programación de aplicaciones 3D está basada en el Modelo de grafos de escena. Conecta objetos separados en una estructura arbórea que incluye los datos geométricos, los atributos e información de visualización, que dan una completa descripción de la escena.

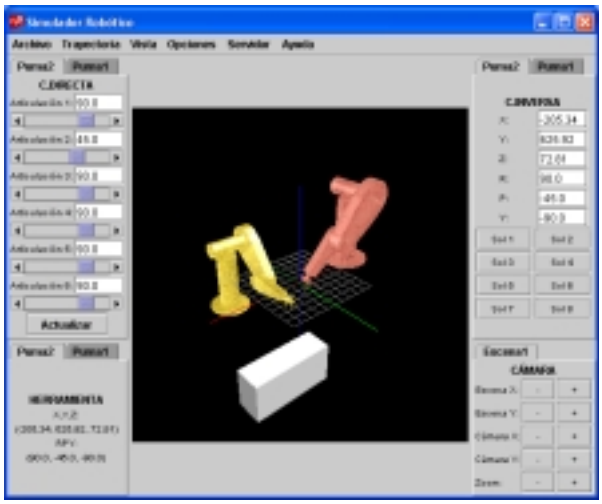

Figura 3: Ejemplo de ESCENA EN Java 3D

2.3 X3D

X3D es un estándar abierto XML, un formato de archivo 3D que permite la creación y transmisión de datos 3D entre distintas aplicaciones y, especialmente, aplicaciones en red. Sus principales características son:

- X3D esta integrado en XML: esto representa un paso fundamental a la hora de conseguir una correcta integración en:
  - Servicios Web.
  - Redes Distribuidas.
  - Sistemas multiplataforma y transferencia de archivos y datos entre aplicaciones.
- X3D es Modular (tiene componentes): lo cual permite la creación de un núcleo 3D más ligero ajustado a las necesidades de los desarrolladores.
- X3D es Extensible: permite añadir componentes para ampliar las funcionalidades según las necesidades del mercado.
- X3D es Perfilado: se pueden escoger distintos grupos de extensiones apropiadas según las necesidades específicas de la aplicación.
- X3D es Compatible con VRML: se mantiene el desarrollo, el contenido y la base de VRML97.

X3D, en lugar de mantener una única especificación amplia y estática, como VRML, que requiere la completa adopción de la misma para conseguir la compatibilidad con X3D, ha optado por una arquitectura basada en componentes que da soporte para la creación de diferentes perfiles, los cuales pueden ser individualmente soportados. Los componentes pueden ser individualmente extendidos o modificados agregando nuevos niveles, o añadiendo nuevos componentes con nuevas características. A través de este mecanismo, se pretende conseguir que los avances de la especificación sean rápidos y que el desarrollo en un

área no retrase la evolución de la especificación en su conjunto.

2.3.1 Perfiles X3D

Los perfiles definidos en la actualidad en X3D son los siguientes:

- *Intercambio:* es el perfil básico para la comunicación entre aplicaciones. Este perfil da soporte a las geometrías, las texturas, las iluminaciones básicas y las animaciones.
- *Interactivo:* es el perfil encargado de aportar las capacidades básicas de interacción con el escenario virtual mediante la incorporación de varios nodos sensoriales (PlanseSensor, TouchSensor, etc.). Además aporta más capacidades de iluminación (Spotlight, PointLight).
- *Inmersivo:* habilita todas las capacidades de gráficos 3D e interacción con el entorno, además incluye el soporte para audio, colisiones, niebla y scripting.
- *Completo:* incluye todos los nodos definidos incluyendo los componentes Nurbs, H-Anim y GeoSpatial.

2.3.2 Estructura de la especificación X3D

Debido a los nuevos requisitos establecidos X3D esta actualmente comprendido por 3 especificaciones ISO distintas:

- *X3D framework & SAI — ISO/IEC FDIS 19775:200x (2003-12-16):* presenta una descripción abstracta de todas las partes funcionales del sistema sin ninguna referencia a cualquier representación concreta. Básicamente explica los modelos estructurales y runtime en términos abstractos. La funcionalidad de los lenguajes de programación externos también es expuesta en términos abstractos como un conjunto de servicios que pueden ser solicitados y servido.
- *X3D encodings — ISO/IEC FCD 19776:200x (2003-02-27):* presenta un conjunto de descripciones de formatos de archivo y cómo la estructura abstracta de la especificación es codificada dentro de los archivos. Se describen dos formatos de archivo de texto-plano: Classic VRML y XML.
- *X3D language bindings — ISO/IEC FCD 19777:200x (2003-05-15):* Define la estructura de distintos lenguajes de programación para los servicios abstractos incorporados en la especificación de X3D. Actualmente están definidas las estructuras para Java y ECMAScript.

3 COMPARATIVA

3.1 VRML y X3D

X3D toma el trabajo seguido por el VRML97 y clarifica las zonas grises que no han sido cubiertas por la especificación a través de los años. Entonces tomado como premisa las bases propuestas por VRML se brinda una mayor flexibilidad. Los grandes cambios incluyen la completa reescritura de la especificación en tres partes distintas que tratan con conceptos abstractos, formatos de archivo y accesos al lenguaje de programación. Otras modificaciones implican una mayor precisión con la iluminación y los modelos de eventos, y cambiar el nombre de algunos campos para una conseguir una mayor consistencia del estándar.

Los cambios importantes se pueden resumir en:

- Expansión de las capacidades gráficas.
- Un modelo de programación de aplicaciones revisado y unificado.
- Múltiple codificación de archivos para describir el mismo modelo abstracto, incluyendo XML
- Arquitectura modular que permite tener rangos de niveles de adopción y soporte para los distintos tipos de mercado existentes.
- Expansión de la estructura de la especificación.

La gráfica de la escena X3D, el corazón de una aplicación X3D, es idéntica a la gráfica de la escena VRML97. El diseño original de la estructura gráfica de VRML97 y sus tipos de nodos estaba basada en tecnología establecida ya existente para gráficos interactivos. Los cambios efectuados inicialmente en los gráficos X3D fueron para incorporar los avances del hardware comercial, a través de la introducción de nuevos nodos y nuevos tipos de campos para datos. Además se hicieron cambios menores para una mejor clarificación, como la mayor precisión en la iluminación y los eventos, y permitir dar valores alpha en los campos de coloración.

X3D tiene una única interfaz de programación de aplicaciones unificada (API). Esto difiere de VRML97 el cual tenía una API interna de scripting API además de un API externa. La API unificada de X3D's simplifica y resuelve muchos de los problemas que existían con VRML97 como resultado de una implementación más robusta.

X3D da soporte para múltiples codificaciones de archivos: VRML97, XML (Extensible Markup Language) y binario comprimido. El formato binario comprimido esta actualmente en desarrollo.

X3D emplea una arquitectura modular para dar una mayor extensibilidad y flexibilidad. La gran mayoría de las aplicaciones no necesitan de todo el poder de X3D, como tampoco el soporte para todas las plataformas y funcionalidades definidos en la especificación. Una ventaja de X3D es que se agrupa por componentes que pueden ser usados por las implementaciones para una plataforma definida o un mercado concreto.

X3D también incluye el concepto de perfiles, una colección predefinida de componentes empleados generalmente en ciertas aplicaciones, plataformas, o en escenarios, por ejemplo el intercambio geométrico entre herramientas de diseño. A diferencia de VRML97, el cual requiere tener un soporte total por parte de la implementación, X3D permite tener un soporte particular para cada necesidad. El mecanismo de componentes X3D también permite a las empresas implementar sus propias extensiones de acuerdo a un riguroso grupo de reglas.

Además la especificación X3D ha sido reestructurada, permitiendo una mayor flexibilidad del ciclo de vida del estándar ajustándose a la evolución del mismo. El estándar X3D esta dividido en tres especificaciones distintas. Esto posibilita cambiar los tiempos y el modo para la adopción por parte de ISO de partes concretas de la especificación.

Desde un punto de vista más funcional los cambios más importantes introducidos en X3D son los siguientes:

- Los archivos están estructurados para definir las necesidades de capacidad como parte del encabezado o inicio.
- Los externprotos sólo definen información externa del archivo X3D. No pueden ser usados como mecanismos de extensión de los navegadores. La forma de hacerlo es a través de componentes personalizados.
- Acceso a los nombres de los campos atendiendo a los cambios realizados desde *eventIn*, *eventOut*, *field* y *exposedField*, a *inputOnly*, *outputOnly*, *initializeOnly* y *inputOutput*, respectivamente.
- Los Scripts pueden tener campos inputOutput (exposedFields).
- El modelo Runtime para interactuar entre el contenido de un script y la gráfica de la escena está rigurosamente definido y muy controlado. VRML97 permitía un script multi-threaded para cambiar arbitrariamente la escena gráfica en un momento dado, sin embargo X3D define sólo un punto específico donde se realizarán los cambios.
- El modelo runtime y de programación para scripting es consistente entre los lenguajes de programación, ya sea que este dentro o fuera el navegador - una API define las reglas de ambos.
- Un conjunto estrictamente definido de tipos abstractos para nodos.

3.2 VRML/X3D Y JAVA3D

Una de las principales diferencias entre VRML/X3D y Java3D, a nivel conceptual, es que Java3D se define como un lenguaje de programación de escenarios 3D a bajo nivel. Es decir, que la creación de objetos y elementos tridimensionales en Java3D requiere no sólo la formación de los elementos 3D, sino también la definición de todos los aspectos relacionados con la visualización y control de las capacidades del escenario.

Esto tiene sus ventajas e inconvenientes. Por ejemplo, para la creación del escenario más simple, el código en Java3D es notoriamente superior al necesario en VRML/X3D, pero por otro lado el control de los distintos elementos presentes en el sistema es superior y más natural en Java3D. Esto no quiere decir que no sea posible controlar el mundo virtual, en VRML, para crear una interacción con el usuario, pero sí que es más complejo. Para ello, en primer lugar, es necesario elegir entre el uso de programación interna dentro del propio código VRML/X3D o de programación externa. Además se está sujeto a la implementación de la especificación VRML/X3D que haya realizado el creador del visor VRML/X3D que se esté empleando. Por ejemplo a nivel de implementación del EAI, algunos visores de VRML como CosmoPlayer se basan en la Máquina Virtual Java de Sun's Microsystem y otros como BS Contact se apoyan en la versión de Microsoft.

Otro aspecto destacable es la pérdida de velocidad y prestaciones en el caso de Java3D frente a otros visores de VRML/X3D desarrollados en C/C++ y empleando directamente Direct 3D u OpengGL. La Tabla 1 [3] ofrece una comparativa exhaustiva en función de los elementos usados.

Elementos usados	Cálculo	Retardo Respecto a C++
C++ (sin 3D)	0.9 * 1 + 0.1 * 1	1.0 (0% más lento)
Java puro (sin 3D)	0.9 * 1.35 +0.1 * 3.25	1.54 (54% más lento)
Mixto Java/C++ (sin 3D)	0.9 * 1 + 0.1 * 3.25	1.225 (22.5% más lento)
C++ (con 3D)	0.4 * 1 + 0.54 * 1 +0.06 * 1	1.0 (0% más lento)
Java puro (con 3D)	0.4 * 2.5 + 0.54* 1.35 +0.006 * 3.25	1.924 (92.4% más lento)
Mixto: Java/C++ (con 3D)	0.4 * 1 + 0.54 * 1.35 + 0.06 * 3.25	1.32 (32% más lento)

Tabla 1: Comparativa de prestaciones entre Java y C++

También es posible emplear Java3D como visor de archivos VRML/X3D. Para ello sólo es necesario emplear alguno de los cargadores (loaders) de VRML/X3D desarrollados para Java3D. En estos momentos el Web3D Consortium esta desarrollando, bajo licencia GNU LGPL (Lesser General Public License), Xj3D como una herramienta totalmente escrita en Java para mostrar contenidos en VRML y X3D. Las ventajas principales de emplear Java3D como visor de VRML/X3D son la capacidad de ejecución en distintas plataformas y el liberar al usuario final de la necesidad de instalar un plug-in específico para el navegador.

Referencias

[1] Ames, A., Nadeau, D. & Moreland, J. (1997) VRML 2.0 sourcebook (2nd ed.), New York. John Wiley & Sons.

[2] D. Selman (2002) Java 3D Programming. Independent Publishers Group.

[3] Jacob Marner, (2002) Evaluating Java for Game Development.

[4] Walsh, A.E. and M. Bourges-Sevenier, (2001) Core Web3D, Prentice Hall PTR.