20

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

SISTEMA DOCUMENTAL EN REALIDAD VIRTUAL SOBRE LA CULTURA MAYA

T E S I S

QUE PARA OBTENER EL TITULO DE:

INGENIERO EN COMPUTACION

P R E S E N T A N:

URBANO CESAR CARRILLO SILVA
GUILLERMO DOMINGUEZ OSORNO

DIRECTOR DE TESIS:

ING. ORLANDO ZALDIVAR ZAMORATEGUI

MEXICO, D. F. 2000

286884

Universidad Nacional Autónoma de México
UNAM
Biblioteca Central
Dirección General de Bibliotecas de la UNAM

UNAM – Dirección General de Bibliotecas
Tesis Digitales
Restricciones de uso

DERECHOS RESERVADOS ©
PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis esta protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

Agradecimientos

Al Ingeniero Orlando Zaldívar Zamorategui

Al Doctor Jesús Savage Carmona

A Claudia Canche por toda su ayuda.

A todos los compañeros del Laboratorio de Interfaces Inteligentes

César Carrillo Silva

A mi familia, especialmente a mis padres por el apoyo que me brindaron.

A David, Guillermo, Pedro, Luis y demás amigos, por los buenos ratos.

A los padres de Guillermo por su hospitalidad.

En general, a todos los que hicieron posible la realización de ésta Tesis.

Guillermo Domínguez Osorno

A mis padres por inscribirme en la UNAM y no en una privada.

A mis amigos Cesar, David y Alma por su amistad en esta etapa de mi vida.

A Pedro, Ericka, Elizabeth y los demás por darme tantos viernes de alegría.

A los del laboratorio de multimedia

ÍNDICE

Introducción

En el presente trabajo se propone una metodología para el desarrollo de sistemas en realidad virtual.

El objetivo de esta tesis es proveer una metodología clara, y probada para la creación de este tipo de sistemas. Se pretende que los alumnos de la Facultad de Ingeniería de la UNAM tengan una herramienta que les ahorre tiempo de planeación a fin de que utilicen el mismo para perfeccionar y desarrollar mejores ambientes y, probablemente, utilizar mejores herramientas en un futuro.

Otro aspecto que pretende la realización de esta tesis, es proporcionar una guía para la realización de sistemas en realidad virtual, desde la metodología propuesta hasta la descripción de las herramientas, tanto de hardware como de software, que podrían ser utilizadas. La idea es que un alumno sin conocimiento alguno sobre el tema pueda, con la ayuda del presente trabajo, desarrollar un sistema completo que explote las capacidades de la realidad virtual.

En el primer capítulo presentamos una visión general de lo que ha sido la realidad virtual a través de su historia, de lo que es actualmente y de lo que puede llegar a ser. También se describe la forma en que las herramientas visuales han sido incorporadas en la antropología. Esto último debido a que la aplicación desarrollada en este trabajo es precisamente enfocada a esta disciplina.

En el segundo capítulo se describen algunas metodologías para la elaboración de sistemas, como lo son: la *metodología en cascada, modelo en espiral, el modelo de prototipos y el modelo de versiones sucesivas.* Esto como preámbulo al desarrollo de una metodología acorde con las características particulares de los sistemas de realidad virtual. El desarrollo de esta metodología es el objetivo fundamental del presente trabajo.

En el tercer capítulo se describen las herramientas tanto de software como de hardware que podrían ayudar a la conformación de un sistema en realidad virtual. Destaca en este capítulo una guía básica para la utilización del programa dVISE, programa en el cual fue desarrollada nuestra aplicación. Este capítulo resulta importante debido a la reducida bibliografía que existe respecto a este programa, aún así es pertinente aclarar que en el Laboratorio de Interfaces se puede disponer de la documentación completa, distribuida por DIVISION.

En el cuarto capítulo se desarrolla una aplicación, tomando como guía la metodología propuesta, esto con el fin de darle un sustento práctico y presentar un resultado perceptible de la misma.

1 Antecedentes de la realidad virtual

1.1 La realidad virtual

1.1.1 Conceptos básicos

1.1.1.1 Realidad virtual

La realidad virtual es un concepto sugerido por Jaron Lanier a principios de los 80's como una tecnología viable, aunque el término ya existía desde los 60's.

Frecuentemente se confunde con el término de "ciberespacio". Este último fue introducido al lenguaje informático por William Gibson en 1984 por medio de su libro Neuromancer; sin embargo, éste es únicamente un libro de ciencia ficción.

En cualquier caso, la realidad virtual pretende emplear una computadora para sumergir los sentidos en un ambiente lo suficientemente convincente como para que la mente lo acepte como una realidad alternativa. Idealmente, el usuario no debería distinguir entre la realidad y la realidad virtual como se advierte en el siguiente fragmento de una historia sobre realidad virtual.

"No sé tampoco si esto es un sueño. No puedo distinguir si es un sueño real, o uno creado por la computadora de realidad virtual.
Y así como en los sueños no se asombra uno de la irrealidad, ni le extraña a uno que sucedan cosas inexplicables, en estas fantasías de realidad virtual la realidad y el surrealismo onírico se mezclan y combinan.
La computadora se alimenta de los datos de mi mente y recrea mundos y objetos que están en ella. Y entonces me pregunto ¿yo soy yo, o una proyección de mi mismo? ¿Una proyección de mis pensamientos, de mis emociones, de mis deseos?".[1]

[1] Ugalde Corral Héctor. "En realidad ... ", http://amadeus.spin.com.mx/~hugalde/en-real.html

"El concepto agrupa dos ideas aparentemente opuestas: "realidad" y "virtualidad". Después de muchos recorridos filosóficos y discusiones, el término "realidad" parece estabilizado en su sentido positivo: es "real" aquello que "tiene existencia verdadera y efectiva", nos dice de forma un tanto circular el Diccionario de la Academia. El término "virtual", la segunda parte, viene definido como aquello "que tiene virtud para producir un efecto, aunque no lo produce de presente". También se señala que se utiliza "frecuentemente en oposición a efectivo o real". Nótese el carácter paradójico que adquiere el concepto "realidad virtual" al quedar su significado como "realidad no-real". En Física, nos dice también el Diccionario, se refiere a aquello "que tiene existencia aparente y no real", es decir, un espejismo, por ejemplo.

Philippe Quéau lo define así: "Un mundo virtual es una base de datos gráficos interactivos, explorable y visualizable en tiempo real en forma de imágenes tridimensionales de síntesis capaces de provocar una sensación de inmersión en la imagen. En sus formas más complejas, el entorno virtual es un verdadero "espacio de síntesis", en el que uno tiene la sensación de moverse "fisicamente". Esta sensación de "movimiento físico" puede conseguirse de diferentes formas; la más frecuente consiste en la combinación de dos estímulos sensoriales, uno basado en una visión estereoscópica total y el otro en una sensación de correlación muscular, llamada "propioceptiva", entre los movimientos reales del cuerpo y las modificaciones aparentes del espacio artificial en que está "inmerso""[2].

"La "realidad" es una construcción a partir de la información sensorial, un conjunto de impresiones que sitúan a los sujetos en el aquí y el ahora, en el espacio y en el tiempo. La "realidad virtual" sería entonces un conjunto de informaciones destinadas a los sentidos cuya función es sustituir la percepción espacio-temporal real del sujeto. Dentro de una realidad o entorno "virtual" el sujeto cree estar donde no está y concede el ser a lo que no es.

[2] Quéau, Philippe. Lo Virtual, Virtudes y Vértigos, Ed. Paidos. Madrid, 1995, pp. 15-16.

Los sueños y las alucinaciones son fenómenos de realidad virtual, pero tienen la característica esencial de no ser voluntarios.

La naturaleza nos ha diseñado —dentro de su plan de protección general— para ser conscientes del aquí y del ahora. Una especie con capacidad voluntaria de desconectarse de su entorno no habría durado mucho sobre la faz de la tierra.

Pero los seres humanos tenemos una insatisfacción permanente que nos ha llevado a elaborar diversas formas voluntarias de desconexión. La primera de ellas —en la que no entraremos— es el consumo de sustancias, naturales o de diseño, destinadas a evadirse de la realidad. El sujeto percibe otra realidad o de otra forma la realidad. En cualquier caso, se produce una alteración de los estados perceptivos que construyen una realidad distinta para el sujeto. Es un sistema en el que, una vez tomada la decisión, se pierde la posibilidad de un control consciente.

Otro gran sistema lo encontramos en las formas artísticas. ¿Qué es una representación teatral sino una realidad virtual? Lo que sucede en el escenario, a la vez, es y no es. Lo percibimos a través de los sentidos, pero se desarrolla en un espacio ficticio, en un paréntesis de la realidad. El escenario, en cuanto escenario, es real, pero es virtual en cuanto espacio escénico. Lo mismo puede decirse de otras formas de creación artística. La literatura crea "universos de la ficción" en donde los personajes se mueven desarrollando sus acciones. Tan ficticio —tan virtual—, en este sentido, es el universo creado en una novela realista decimonónica, como el que se establece en una novela utópica, de ciencia-ficción o cualquier género fantástico. Las formas narrativas apelan a la imaginación del lector para que se recreen esos mundos virtuales en los que se desarrollan las acciones. Pero, sin dejar de contar con la capacidad creativa o re-creativa del lector, el mundo de la ficción narrativa no ofrece la posibilidad de ser alterado, como decía Jacques el fatalista en la obra de Diderot, "todo está escrito en el Gran Rollo". Por así decirlo, es un turismo programado y con guía. O si se quieren utilizar los términos hoy en uso, no se da "interacción".

La realidad virtual crea un espacio, pero no crea unas acciones; tan sólo su posibilidad. Si toda trama narrativa es el resultado de un proceso de selección, en los mundos virtuales no existe trama porque no existe selección. Un mundo virtual es un espacio de posibilidades, es decir, tiene las condiciones de otra realidad. La participación no se da mediante mecanismos de identificación, como sucede con las formas narrativas o teatrales. De hecho, la idea de "identificación" desaparece porque existe una auténtica participación. No nos identificamos con los protagonistas; somos los protagonistas.

La "realidad virtual" se encuentra en su estado inicial. Las posibilidades, en todos los campos, producen vértigo. También lo producen sus posibles consecuencias, según los usos a los que pueda destinarse. Por primera vez, vamos a tener la posibilidad de actuar simultáneamente sobre los sentidos creando un mundo en el que, a diferencia del real, nadie nos exigirá responsabilidades porque es asocial. El mundo virtual es un mundo privado, una fantasía en la que sus participantes poseen una libertad absoluta, y cuyas acciones están, por decirlo así, fuera del derecho."[3]

1.1.1.2 Sistemas de realidad virtual

Un sistema de realidad virtual es aquel que intenta crear una realidad alterna a través de un sistema de cómputo. Para esto se ha desarrollado tecnología a fin de poder interactuar con los sentidos del ser humano; estos dispositivos son muy diversos y van desde un casco para presentar el ambiente ante el sentido de la vista hasta un guante que permite transmitir pulsos eléctricos para simular el tacto.

Algunas características importantes que los dispositivos deben tener son las siguientes:

[3] Aguirre Romero, Joaquín Mª. "Artes de la memoria y realidad virtual". http://www.ucm.es/info/especulo/numero2/memoria.htm.

El ambiente deberá ser tridimensional (*tridimensionalidad*), a fin de asemejarse a la realidad. Deberá también tener una respuesta inmediata a los movimientos que realice el usuario (*capacidad sintáctica o respuesta en tiempo real*), es importante la capacidad que tenga el sistema para reaccionar a las acciones del usuario (*interactividad*).

Evidentemente, es necesario que el usuario se pueda desplazar libremente en el ambiente (*navegación*). Resulta indispensable, para ello, que el sistema reconozca la posición y orientación del usuario; esta ubicación servirá para definir el llamado *punto de vista*. Finalmente, definiremos dos conceptos igualmente importantes en un ambiente virtual.

Un ambiente con una buena *inmersión* deberá:

- Ser percibido como auténtico
- Permitir la interacción intuitiva y responder a ella en menor tiempo posible.
- Facilitar la recopilación, percepción y análisis de datos

Los *niveles de inmersión* de un ambiente virtual, son determinados de acuerdo con la semejanza que el ambiente logre de la realidad.

Para que un usuario se sienta inmerso en un ambiente tridimensional es necesario que perciba la distancia a la que se encuentran los objetos. Para lograr esto los sistemas en realidad virtual se valen de dos métodos; por un lado se utiliza la estereoscopia que funciona a partir del método utilizado por el cerebro humano. Este método toma dos perspectivas y presenta las imágenes a los ojos del usuario.

El otro método para lograr que el usuario capte la profundidad a la que se encuentran los objetos es mediante las llamadas claves de profundidad psicológicas, que se refieren a aspectos visuales como la interposición de objetos, y el sombreado.

Los modos de presentación se refieren a la forma en que se presenta la imagen ante el usuario. El modo de presentación cilindrico muestra al ambiente en una pantalla plana verticalmente y circular en el plano horizontal, de tal forma que el usuario ve perfectamente hacia cualquier dirección a excepción de arriba y abajo.

El modo de presentación en hemisferio permite al usuario ver el ambiente en una cúpula que lo rodea; un ejemplo de esto son los planetarios[4].

1.1.2 Aplicaciones

En la actualidad la realidad virtual ha encontrado múltiples aplicaciones en áreas tan diversas como la medicina y la milicia. A continuación citaremos algunas de ellas.

La telepresencia es una aplicación de la realidad virtual mediante la cual se controla remotamente algún dispositivo. Una de sus aplicaciones es un robot teledirigido utilizado por grupos de bomberos para situaciones de extremo peligro.

Otras aplicaciones de gran importancia se dan en Internet. Para esto se creó un lenguaje que permite la descripción de mundos virtuales en la WWW. Este lenguaje es llamado VRML (Virtual Reality Modelling Language). Evidentemente, la problemática que actualmente enfrenta el lenguaje es la velocidad de la red; sin embargo, esto se encuentra en vías de resolverse gracias a avances como la red Internet II y las nuevas frecuencias utilizadas en las comunicaciones satelitales.

Otra aplicación importante se da en el campo de la Internet pero independientemente de VRML. Se trata de DIVE (Distributed Interactive Virtual Enviroment), un sistema multiusuario de realidad virtual que se basa en la Internet donde los participantes navegan en una espacio tridimensional y ven, conocen e interactúan con los otros usuarios y aplicaciones.

[4] Casey Larijani, L. Realidad virtual.McGraw Hill. pp. 17-29.

1.1.3 Cronología

A continuación se presenta una cronología[5] del desarrollo de la realidad virtual que podrá ubicarnos en el tiempo conforme los avances en el área se fueron dando.

1965	Surge el concepto de realidad virtual, cuando Ivan Sutherland (hoy miembro de Sun Microsystems Laboratories) publicó un artículo titulado "The Ultimate Display", en el cual describia el concepto básico de la realidad virtual. El trabajo inicial del doctor Sutherland fue básico para investigaciones subsecuentes en este terreno.
1966	Sutherland creó el primer casco visor de realidad virtual al montar tubos de rayos catódicos en un armazón de alambre. Este instrumento fue llamado "Espada de Damocles", debido a que el estorboso aparato requería de un sistema de apoyo que pendía del techo. Sutherland también inventó casi toda la tecnología.
1968	Ivan Sutherland y David Evans crean el primer generador de escenarios con imágenes tridimensionales, datos almacenados y aceleradores. En este año se funda también la sociedad Evans & Sutherland.
1971	Redifon Ldt en el Reino Unido comienza a fabricar simuladores de vuelo con displays gráficos. Henri Gouraud presenta su tesis de doctorado "Despliegue por computadora de Superficies Curvas".
1972	General Electric, bajo comisión de la Armada norteamericana, desarrolla el primer simulador computarizado de vuelo. Los simuladores de vuelo serán un importante renglón de desarrollo para la realidad virtual.
1973	Bui-Tuong Phong presenta su tesis de doctorado "Iluminación de imágenes generadas por computadora".
1976	P. J. Kilpatrick publica su tesis de doctorado "El uso de la Cinemática en un Sistema Interactivo Gráfico".
1977	Dan Sandin y Richard Sayre inventan un guante sensitivo a la flexión.
1979	Eric Howlett (LEEP Systems, Inc.) diseñan la Perspectiva Optica Mejorada de Extensión Larga (Large Expanse Enhanced Perspective Optics, LEEP).
80´s	A principios de los 80´s la realidad virtual es reconocida como una tecnología viable. Jaron Lanier es uno de los primeros generadores de aparatos de interfaz sensorial, acuñó la expresión "Realidad Artificial", también colabora en el desarrollo de aparatos de interface VR, como guantes y visores.
1980	Andy Lippman desarrolla un videodisco interactivo para conducir en las afueras de Aspen.
1981	Tom Furness desarrolló la "Cabina Virtual". G. J. Grimes, asignado a Bell Telephone Laboratories, patentó un guante para introducir datos.
1982	Ocurre uno de los acontecimientos históricos en el desarrollo de los simuladores de vuelo, cuando Thomas Furness presentó el simulador más avanzado que existe, contenido en su totalidad en un casco parecido al del personaje Darth Vader y creado para la U.S. Army AirForce. Thomas Zimmerman patentó un guante para introducir datos basado en sensores ópticos, de modo que la refracción interna puede ser correlacionada con la flexión y extensión de un dedo.

[5] Gonzalez Guizar, Alberto. "Historia de la realidad virtual". http://exodus.dcaa.unam.mx/virtual/history1.html

1983	Mark Callahan construyo un HMD en el Instituto Tecnológico de Massachusetts (MIT).
1984	William Gibson publica su novela de ciencia ficción, Neuromancer en el que se utiliza por primera vez el término "Ciberespacio" refiriéndose a un mundo alternativo al de las computadoras; por lo que algunos aficionados empiezan a utilizarlo para referirse a la realidad virtual. Mike Mc Greevy y Jim Humphries desarrollaron el sistema VIVED (Representación de un Ambiente Virtual, Virtual Visual Environment Display) para los futuros astronautas en la NASA.
1985	Jaron Lanier funda la institución VPL Research. Los investigadores del laboratorio Ames de la NASA construyen el primer sistema práctico de visores estereoscopios. Mike Mc Greevy y Jim Humphries construyen un HMD con un LCD monocromo del tamaño de una televisión de bolsillo.
1986	En el centro de investigaciones de Schlumberger, en Palo Alto, California, Michael Deering (científico en computación) y Howard Davidson (fisico) trabajaron en estrecha relación con Sun Microsystems para desarrollar el primer visor de color basado en una estación de trabajo, utilizando la tecnología de Sun. Existen ya laboratorios como el de la NASA, Universidad de Tokio, Boeing, Sun Microsystems, Intel, IBM y Fujitsu dedicados al desarrollo de la tecnología VR.
1987	La NASA utilizando algunos productos comerciales, perfecciona la primera realidad sintetizada por computadora mediante la combinación de imágenes estéreo, sonido 3-D, guantes, etc. Jonathan Waldern forma las Industrias W (W Industries). Tom Zimmerman et al. desarrolla un guante interactivo.
1988	Michael Deering y Howard Davidson se incorporan a la planta de científicos de Sun. Una vez allí, el Dr. Deering diseñó características VR dentro del sistema de gráficos GT de la empresa, mientras que el Dr. Davidson trabajaba en la producción de visores de bajo costo.
1989	VPL, y después Autodesk, hacen demostraciones de sus completos sistemas VR. El de VPL es muy caro (225,000 dólares), mientras que el de Autodesk no lo es tanto (25,000 dólares). Jaron Lanier, CEO of VPL, creó el término "realidad virtual". Robert Stone forma el Grupo de Factores Humanos y realidad virtual. Eric Howlett construyo el Sistema I de HMD de video LEEP. VPL Research, Inc. comenzó a vender los lentes con audífonos que usaban despliegues ópticos LCD y LEEP. Autodesk, Inc. Hizo una demostración de su PC basada en un sistema CAD de realidad virtual, Ciberespacio, en SIGGRAPH'89. Robert Stone y Jim Hennequin coinventaron el guante Teletact I. Las Tecnologías de Reflexión producen el visor personal.
1990	Surge la primera compañía comercial de software VR, Sense8, fundada por Pat Gelband. Ofrece las primeras herramientas de software para VR, portables a los sistemas SUN. ARRL ordena el primer sistema de realidad virtual de Division. J. R. Hennequin y R. Stone, asignados por ARRL, patentaron un guante de retroalimentación tangible.
1991	Industrias W venden su primer sistema virtual. Richard Holmes, asignado por Industrias W, patentó un guante de retroalimentación tangible.
1992	SUN hace la primera demostración de su Portal Visual, el ambiente VR de mayor resolución hasta el momento. Al Gore, vicepresidente de Estados Unidos y promotor de la realidad virtual, dictó seminarios sobre la importancia de esta

	tecnología para la competitividad norteamericana. T. G. Zimmerman, asignado por VPL Research, patentó un guante usando sensores ópticos. Division hace una demostración de un sistema de realidad virtual multiusuario. Thomas De Fanti et al. hizo una demostración del sistema CAVE en SIGGRAPH.
1993	SGI anunció un motor de realidad virtual.
1994	La Sociedad de realidad virtual fue fundada. IBM y Virtuality anunciaron el sistema V-Space. Virtuality anunció su sistema serie 2000. Division hizo una demostración de un sistema integrado de realidad virtual multiplataformas en IITSEC, Orlando.

1.2 Antropología visual

La aplicación que se realiza en el presente trabajo responde a la necesidad de utilizar los recursos tecnológicos que se desarrollan en la facultad de ingeniería a campos de la educación distintos a la misma ingeniería. En este caso abordamos la antropología debido a la necesidad de difundir esta disciplina. Por otro lado, consideramos de un gran interés el estudio de las culturas prehispánicas, en este caso de la cultura Maya.

"El registro visual ha transitado de las pinturas rupestres a la imagen virtual. De salto en salto cualitativo y de revolución en revolución tecnológica, estos desarrollos extraordinarios han permitido que en este último siglo se haya podido transitar del daguerrotipo y la imagen fija, invento de los hermanos Lumier y, de ahí, a la inclusión del sonido en la imagen, al movimiento y al color, diversificando y ampliando el ámbito del registro visual. La historia de la antropología moderna ha incorporado el registro de la imagen etnográfica, el uso de la fotografía, el cine, el vídeo, la digitalización, el CD-ROM y todas las posibilidades de la multimedia hasta llegar a la imagen virtual."[6]

"La etnografía moderna no sería lo mismo sin el registro de la imagen y el sonido. Este recurso para la investigación científica, amplió las posibilidades de la lectura e interpretación de las culturas; las bondades de su aplicación en la etnología han sido replicadas en la antropología clásica, en la arqueología y en la lingüística. Más aún, han sido también ámbito de discusión teórico y metodológico, en el que se debate su estatuto científico como una subdisciplina antropológica"[7].

En el primer encuentro de antropología visual organizado por el Congreso Internacional de Ciencias Etnológicas y Antropológicas, Francisco Fernández Repetto señaló que "la antropología visual, además del registro de los hechos de la realidad, también pretende imprimir las presentaciones visuales en una realidad

[6] Salazar Peralta, Ana M. Antropología Visual. UNAM, IIA. México, 1997. pp. 9
[7] ibídem pp.10

ordenada teóricamente". Por otro lado, Octavio Hernández menciona que "el elemento primario de la antropología visual es la imagen y, por ende, cualquier paradigma de la antropología visual se construye a partir de la imagen misma. En este sentido podemos estudiar algunas imágenes producidas por el hombre que vaya de lo rupestre a lo iconográfico, representaciones en dos dimensiones, y desde el tótem hasta los santos, representaciones en tercera dimensión: para construir de esta manera una antropología de la imagen."[8]

[8] ibidem pp. 11

2 Metodología para el desarrollo del sistema

Existen numerosas metodologías para el desarrollo de sistemas de cómputo. Para el desarrollo de sistemas en realidad virtual, sin embargo, se hace necesario el estudio de estas metodologías a fin de definir cuál es la más adecuada debido a las condiciones tan particulares que estos sistemas tienen.

Entre las metodologías existentes, se encuentran dos principales enfoques posibles hasta el momento, el tradicional y el de cuarta generación. El enfoque tradicional utiliza programas procedurales en el desarrollo de los sistemas, las técnicas basadas en este enfoque acostumbran describir los sistemas realizados mediante una explicación del cómo trabajan, esto es, en su documentación se especifica toda la lógica con la que el código del sistema funciona y esto permite, a quien de mantenimiento al mismo, entenderlo y por ende poderlo modificar.

Las técnicas basadas en los lenguajes de cuarta generación prefieren especificar las funciones que el programa tiene en vez de explicar cómo lo hacen. Los programas de cuarta generación son, generalmente, orientados a objetos y/o eventos. Por otro lado, estos lenguajes son generadores de código por lo que la mayor parte del mismo no es escrito por el desarrollador directamente y sería no sólo difícil sino improductivo el explicar un código que se genera automáticamente.

En el caso de los sistemas en realidad virtual, se maneja un programa de cuarta generación, sin embargo, las técnicas que estudiaremos están diseñadas para cualquier tipo de lenguaje; esto es debido a que no determinan el modo en que se programará directamente sino más bien el orden y los objetivos generales de cada etapa del proyecto. Como podremos observar en las diversas metodologías así como en la propuesta que haremos, la etapa de programación está considerada. Será dentro de ésta, y sobre todo en la etapa de documentación, donde se tomará en cuenta la diferencia entre los tipos de lenguajes utilizados.[9]

[9] Pressman, Roger. Software engeeniering, A beginner's guide. Mc Graw Hill, E.U.A., 1988. pp.18-21.

En este capítulo estudiaremos algunas de las metodologías más representativas en el desarrollo de sistemas de cómputo, analizaremos la posibilidad de aplicar alguna de ellas a nuestro proyecto. En caso de no existir posibilidad de aplicar alguna metodología existente a nuestro proyecto, tomaremos conceptos de cada una de ellas a fin de conformar una nueva metodología. En el estudio de estas metodologías, evaluaremos las ventajas y desventajas de éstas con el fin de tomar sus cualidades y desechar los elementos que no se acoplen al problema actual.

El estudio y aprovechamiento de las metodologías existentes nos permitirá darle una base conceptual más sólida a esta nueva propuesta, si la hubiese. Posteriormente, se presentará una aplicación que le dará una base práctica a la metodología elegida.

2.1 Metodologías para el desarrollo de sistemas de cómputo

Como se mencionó anteriormente, el estudio de las metodologías existentes resulta necesario para la creación de la nueva propuesta; debido a esto, a continuación describiremos cuatro de las principales metodologías para el desarrollo de sistemas. En cada una de estas metodologías comentaremos los puntos que las hacen o no óptimas para el desarrollo de sistemas en realidad virtual como el que nosotros proponemos.

2.1.1 Metodología en cascada.

La metodología en cascada presenta una estructura rígida que, como veremos, podrá ser modificada para ser adaptada al problema especifico que se nos presenta. El modelo en cascada se popularizó en los años setenta y es el más frecuentemente citado en la literatura especializada.

Como lo muestra la ***Figura 2.1***, el modelo de cascada está estructurado como una cascada de fases donde la salida de una fase constituye la entrada para la siguiente. Cada fase está estructurada por un conjunto de actividades que pueden ser ejecutadas por diferentes personas al mismo tiempo.

Las fases mencionadas en la figura no son, sin embargo las únicas en la realización del sistema de acuerdo con este modelo, existen independientemente de éstas, otras tres actividades que no son contempladas en la imagen.

Esto es debido a que en realidad son paralelas a todo el proceso y no forman parte de la cascada directamente como se muestra en la ***Figura 2.2***.

La primera de las actividades que se introducen en el diagrama es la ***administración***. La administración es una actividad fundamental que agudiza y monitorea el desarrollo entero y el proceso de mantenimiento.

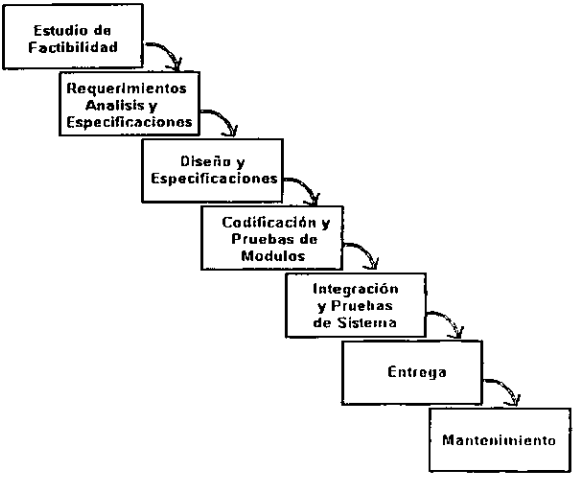


Figura 2.1 Metodología en cascada

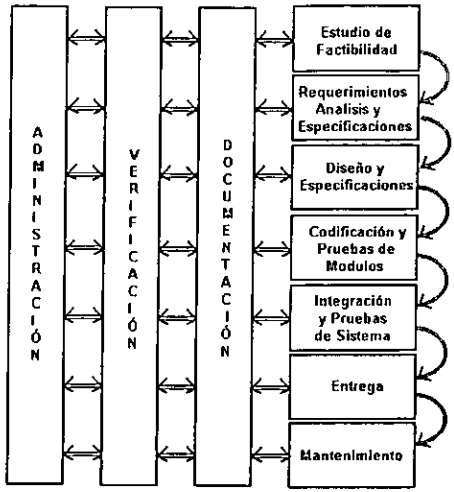


Figura 2.2 Actividades paralelas a la metodología en cascada"

Existen tres aspectos primordiales que caen dentro de la administración. El primero de ellos es la adecuación del proceso con el método que se pretende utilizar. Esta actividad da por sentada la ruptura con la rigidez original del modelo en cascada. *La definición de políticas* es el segundo de los aspectos fundamentales que cubre la administración. Finalmente, la administración tiene que lidiar con los recursos que se necesitarán para el desarrollo del proyecto, en particular con los *recursos humanos*. Nuevamente podemos observar que en este punto el modelo no se aplica a nuestro caso; en un desarrollo como el nuestro, los recursos tanto humanos como técnicos están dados y el único recurso a modificar es el tiempo.

La siguiente actividad paralela al modelo de cascada es la ***verificación***. Si bien la verificación se realiza en dos etapas específicas del modelo en cascada – pruebas modulares y pruebas de sistema.-, en realidad la verificación debe realizarse en cada una de las etapas del modelo. En la mayoría de los casos esta actividad se efectúa en forma de proceso de control de calidad, esto es, en forma de revisiones e inspecciones. El objetivo de esta actividad es monitorear la calidad de la aplicación durante el desarrollo de la misma y no únicamente a posteriori. Los

errores deben ser detectados lo antes posible con el fin de no tener que hacerle mejoras a un sistema defectuoso.

Finalmente, la ***documentación*** del sistema es una actividad de suma importancia para el futuro del proyecto, tan es así que de su calidad dependerá el posible crecimiento de la aplicación en un futuro; la documentación permitirá también una rapidez mucho mayor en la corrección de errores dentro de la etapa de mantenimiento. La documentación es una actividad intrínseca al modelo de cascada, en este modelo, no es vista como una actividad a realizarse al final del proyecto, mas bien se trata de una salida obligada en cada etapa del método, de esta manera, dependerá de la salida si se da o no acceso a la siguiente etapa del modelo. Por esto se dice que el modelo en cascada está guiado precisamente por la documentación.

El modelo de cascada[10] es elegido generalmente para aquellos sistemas sencillos y fáciles de entender debido a que los sistemas más complejos pueden requerir ser divididos en procesos más pequeños con el fin de controlar de mejor manera el proceso y asegurarse más rigurosamente del desarrollo paso a paso. Otro aspecto que favorece la elección de este modelo son los casos en que el usuario final es experto, ya que de no serlo se haría necesario incluir una fase de entrenamiento para el mismo dentro del modelo.

Un tercer factor hace prácticamente necesario modificar de cierta manera el modelo. Este factor se refiere al papel que desempeña el desarrollador del software, es decir, el tipo de software a producir. Se tienen tres casos principalmente:

1. El caso en que una empresa desarrolla un sistema respondiendo a requerimientos específicos dados por un cliente que pertenece a una empresa distinta.

[10] Sommerville,Ian. Ingeniería de Software. 2ª Edición Addison Wesley, México, 1988. pp. 41-42

2. El caso en que un grupo de desarrolladores realizan una aplicación para una empresa a la cual pertenecen.
3. Finalmente, tenemos la opción que se acerca más a nuestro caso; se trata de aquellas empresas que desarrollan un software bajo sus propios requerimientos con la idea de lanzar el producto final al mercado. La diferencia con nuestro caso, como se mencionó en la introducción, radica en la necesidad de la empresa de obtener dinero del proyecto lo cual obliga a la realización de un estudio de mercado.

Cualquiera de estos casos provocarán distintas variantes al modelo de cascada.

2.1.2 Modelo de prototipos

Un prototipo es una representación o modelo del producto de programación que, a diferencia de un modelo de simulación, incorpora componentes de un producto real. Por lo regular, un prototipo tiene un funcionamiento limitado en cuanto a capacidad, confiabilidad o eficiencia.

Existen varias razones para desarrollar un prototipo; una de ellas es ilustrar los formatos de datos de entrada, mensajes, informes y diálogos al cliente, éste es un mecanismo adecuado para explicar opciones de procesamiento y tener un mejor entendimiento de las necesidades de él.

La segunda razón es para explorar aspectos técnicos del producto propuesto. Con frecuencia una decisión importante del diseño dependerá, por ejemplo, del tiempo de respuesta del controlador de un dispositivo o de la eficiencia de un algoritmo de clasificación; en tales casos, un prototipo puede ser la mejor o única manera de resolver un problema.

La tercera razón para desarrollar un prototipo se da cuando el modelo en cascada clásico, análisis → diseño → instrumentación, es inapropiado. Este último modelo se aplica cuando se puede especificar el total de los requerimientos al inicio del

ciclo de vida. Algunas veces no es posible definir el producto sin un desarrollo exploratorio y, en ocasiones no es claro cómo proceder a la mejora del sistema hasta que no se instrumenta y evalúa una versión. El desarrollo exploratorio se utiliza para desarrollar algoritmos para jugar ajedrez, para resolver problemas confusos, y para llevar a cabo tareas que requieren la simulación del comportamiento humano; sin embargo, esta técnica no se limita a estas situaciones. [11]

En el caso de un sistema de realidad virtual como el que proponemos, el desarrollo del proyecto requiere, en su conjunto, de un método en fases, esto debido a la necesidad de finalizar una etapa del proyecto antes de continuar con la siguiente. Sin embargo, el modelo en prototipos es muy adecuado para ser implementado en el desarrollo de una etapa en particular, este es el caso de las etapas de análisis e implementación.

2.1.3 Modelo de versiones sucesivas

Se pueden desarrollar nuevas versiones de un producto ya existente con el modelo en cascada y sin ningún prototipo. El avance de un producto totalmente nuevo, tal vez requiera de prototipos durante las fases de planeación y análisis. El producto también se puede desarrollar como una serie de diseños e instrumentaciones.

A esto último se le denomina método de versiones sucesivas.

El desarrollo de productos mediante el método de versiones sucesivas es una extensión del método de prototipos, en el que se define un esqueleto inicial del producto, obteniendo así, cada vez más capacidades. En dicho método, cada versión es un sistema funcional y capaz de realizar un trabajo útil. La Figura 2.3 a) ilustra la fase de análisis seguida por el diseño de instrumentación de versiones sucesivas en un proceso iterativo. Las líneas punteadas indican que al conseguir la versión I pueden necesitarse más análisis antes de diseñar la versión I+1.

[11] Fairley, Richard E. Ingeniería de Software. Mc Graw Hill, México, 1988. pp.52-53

En la Figura 2.3 b) las versiones 1 a la N del producto se diseñan antes de cualquier actividad de instrumentación. En este caso, las características de cada diseño sucesivo serán planeadas durante la fase de análisis. Las líneas punteadas de la Figura 2.3 b) indican que la instrumentación de la I-esima versión puede demostrar la necesidad de mayor análisis y diseño antes de la puesta en marcha de la versión I+2. En realidad el ciclo de desarrollo de un producto de programación es una combinación de los distintos modelos presentados. Las organizaciones y proyectos especiales pueden adoptar algunos de estos modelos en particular; sin embargo, ciertos elementos de ellos se encuentran en todo proyecto de programación. Por ejemplo, para proyectos de desarrollo de programación no es extraño adoptar el modelo en cascada como marco de referencia básica e incluir prototipos y versiones sucesivas en el desarrollo.

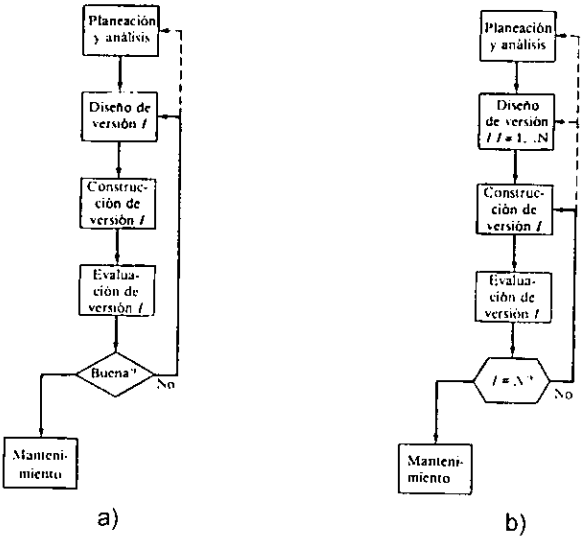


Figura 2.3 Modelo de versiones sucesivas

Este último punto ilustra de forma adecuada el razonamiento hecho en la conformación del método que presentaremos, como se verá en su momento, la metodología planteada maneja conceptos de los cuatro métodos expuestos.[12]

2.1.4 Modelo en espiral

La meta del modelo en espiral en el proceso de producción de software es proveer un marco de trabajo para el diseño de dicho proceso guiado por los niveles de riesgo en un proyecto dado.

Opuestamente al modelo anterior, el modelo en espiral puede ser considerado como un metamodelo, esto es debido a que cualquier otro modelo de producción de software puede ser adaptado a él. La elección del modelo que se adaptará dependerá del principio de nivel de riesgo, de acuerdo con esto, el modelo en espiral provee una visión del proceso de producción que soporta la administración de riesgos.

Con el fin de evitar confusiones, definiremos los conceptos de riesgo y análisis de riesgos.

Riesgo: Se entiende por riesgo a las circunstancias adversas potenciales que puedan deteriorar el proceso de desarrollo y la calidad de los productos.

Análisis de Riesgos: Se define como una disciplina cuyos objetivos son identificar, direccionar y eliminar puntos de riesgo en el software antes de que se conviertan en una amenaza para la correcta operación del mismo o la principal fuente de costos por reconstrucción en el software.

El modelo en espiral se enfoca en la identificación y eliminación de problemas de alto riesgo y en el cuidado riguroso del diseño del proceso.

[12] ibidem. pp. 53-55

La principal característica del modelo en espiral se encuentra en que es cíclico y no lineal como en el caso del modelo en cascada clásico.

Este modelo puede ser concebido de distintas formas de acuerdo con la bibliografía que se consulte. Sommerville[13]considera que cada ciclo de la espiral está constituido por cuatro estados y cada estado está representado por un cuadrante en el diagrama cartesiano. El radio de la espiral representa el costo acumulado hasta el momento en el proceso, el ángulo representa el progreso en el proceso.

El estado uno identifica los objetivos de la porción del producto bajo consideración, esto en términos de calidad. Inclusive, identifica alternativas en términos de si comprar, diseñar o utilizar partes del software que ya se tiene.

Las alternativas identificadas en el estado 1 son evaluadas en el estado 2. Además, se identifican las áreas de riesgo potencial. La evaluación del riesgo puede requerir planear distintos tipos de actividades, algunas de ellas son el prototipaje, entendido éste como una versión funcional del sistema a diferencia del concepto tradicional de prototipo, y la simulación.

La evaluación de riesgo se refiere a un estudio que implica conocer las posibilidades que se tienen de acuerdo con las condiciones existentes, esto es, se determina la factibilidad del proyecto. El Análisis y Control de riesgos es un tema de la teoría de administración, existen muchas técnicas para identificar los riesgos del proyecto, determinar su impacto, monitorearlos y controlarlos. El conocimiento de dichas técnicas permite al líder del proyecto aplicarlas cuando sea necesario para incrementar las oportunidades de éxito del proyecto.[14]

En el tercer estado se desarrolla y se verifica el siguiente nivel del producto, nuevamente la estrategia a seguir es dictada por el análisis de riesgo.

[13] Sommerville, Ian. Op. cit.
[14] Ghezzi, Carlos, et al. Fundamentals of Software Engineering. Ed. Prentice Hall, E.U.A. 1991. pp. 447

Finalmente el estado cuatro consiste en una revisión de resultados de las etapas anteriores y en la planeación de la siguiente iteración del ciclo, si es que esta iteración es necesaria.

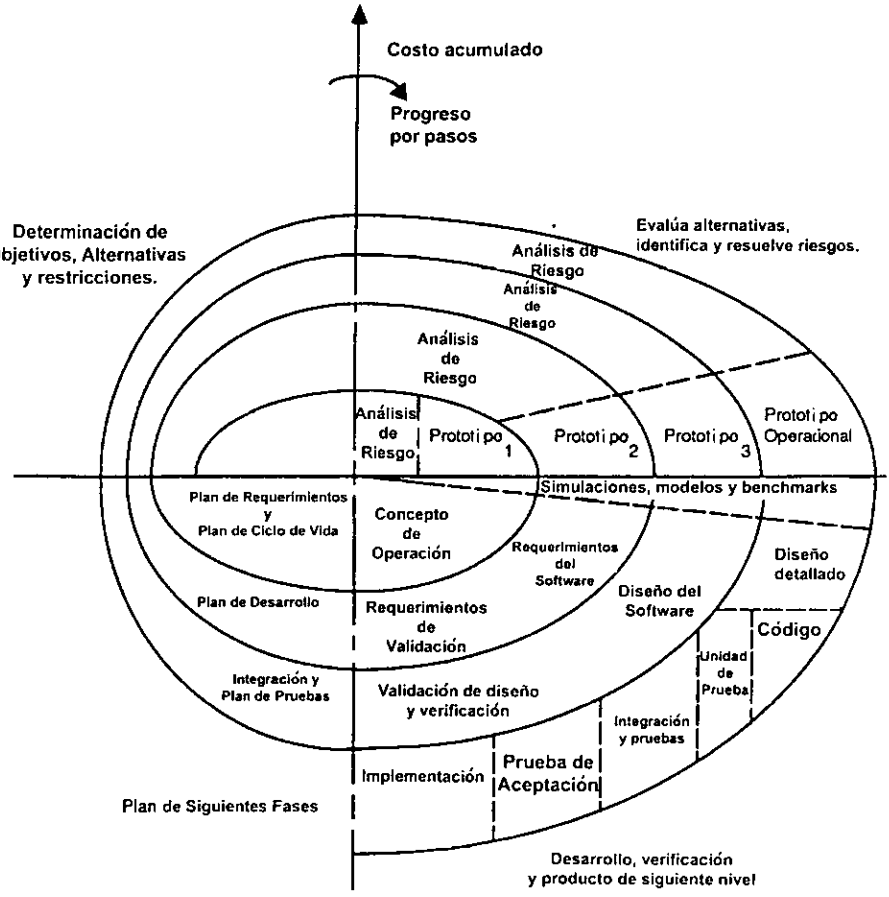


Figura 2.4(a) Metodología en espiral de acuerdo con Ian Sommerville

El modelo de espiral nos permite reiterar la expansión del crecimiento del sistema contra la corrección del mismo. Después de un ciclo, los requerimientos no especificados anteriormente se convierten en requerimientos para la siguiente iteración. A consecuencia de esto, el crecimiento del sistema se acerca cada vez

más al mejoramiento del mismo. El modelo de espiral, de acuerdo con Sommerville, es ilustrado en la ***Figura 2.4(a)***.

Por otro lado, Pressman[15], basado en el modelo original de Boehm, considera que el ciclo del modelo se divide en seis regiones (ver ***Figura 2.4(b)***). Éstas son:

- ✓ Comunicación con el cliente: las tareas requeridas para establecer comunicación entre el desarrollador y el cliente.
- ✓ Planificación: las tareas requeridas para definir recursos, el tiempo y otras informaciones relacionadas con el proyecto.
- ✓ Análisis de riesgos: las tareas requeridas para evaluar riesgos técnicos y de gestión.
- ✓ Ingeniería: las tareas requeridas para construir una o más representaciones de la aplicación.
- ✓ Construcción y adaptación: las tareas requeridas para construir, probar, instalar y proporcionar soporte al usuario.
- ✓ Evaluación del cliente: las tareas requeridas para obtener la reacción del cliente según la evaluación de las representaciones del software creadas durante la etapa de ingeniería e implementada durante la etapa de instalación.

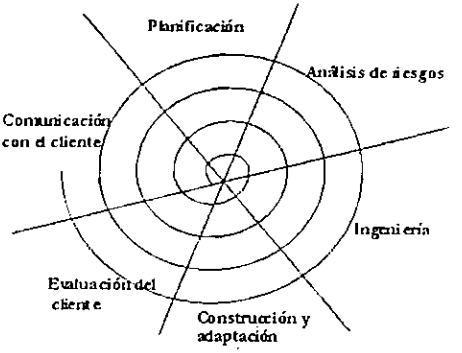


Figura 2.4(b) Modelo en espiral de acuerdo con Roger Pressman.

[15] Pressman, Roger, Op. Cit. pp. 28-29.

En este caso, se trata de una combinación del modelo de prototipos con el de versiones sucesivas; es decir, en los primeros ciclos se producen prototipos y posteriormente se logran versiones completas del proyecto que evolucionan en cada ciclo.

En el modelo que plantearemos más adelante, este concepto se implementará en la etapa de mantenimiento.[16]

[16] ibidem. pp.53

2.2 Metodología propuesta

Una vez estudiadas las metodologías existentes concluimos que ninguna de ellas se ajusta plenamente al desarrollo de sistemas en realidad virtual. Aún así encontramos en algunas de ellas elementos útiles para algunas secciones de una nueva metodología que a continuación presentamos.

Como se puede observar en la ***Figura 2.5***, la metodología que proponemos para la realización de sistemas en realidad virtual es más compleja que el resto. Esto se explica teniendo en consideración que para la concepción de esta nueva metodología se utilizaron los conceptos de varias otras. Estos conceptos fueron integrados de acuerdo con las necesidades que presentan las distintas etapas que constituyen a la metodología.

Una de las razones que hace imposible la adaptación de las metodologías existentes a proyectos de este tipo se encuentra en las diferencias del proyecto en sí, un ejemplo es el aspecto técnico del mismo; debemos entender que la tecnología de realidad virtual conlleva una redefinición de la estructura del sistema, por otro lado, la forma de programar también cambia de un lenguaje procedural a uno orientado a objetos y eventos, este último es nuestro caso.

En las siguientes páginas describiremos etapa por etapa las actividades a realizar de acuerdo con la metodología.

Debido a la naturaleza de este tipo de sistemas, es necesario que la estructura fundamental de la metodología que se utilice para su realización se constituya en fases. Esto último, debido a la ya mencionada necesidad de tener los resultados de una etapa para comenzar con la siguiente. Por otro lado, para la realización de cada etapa, es adecuado utilizar distintas metodologías, dependiendo de las características de la etapa en cuestión.

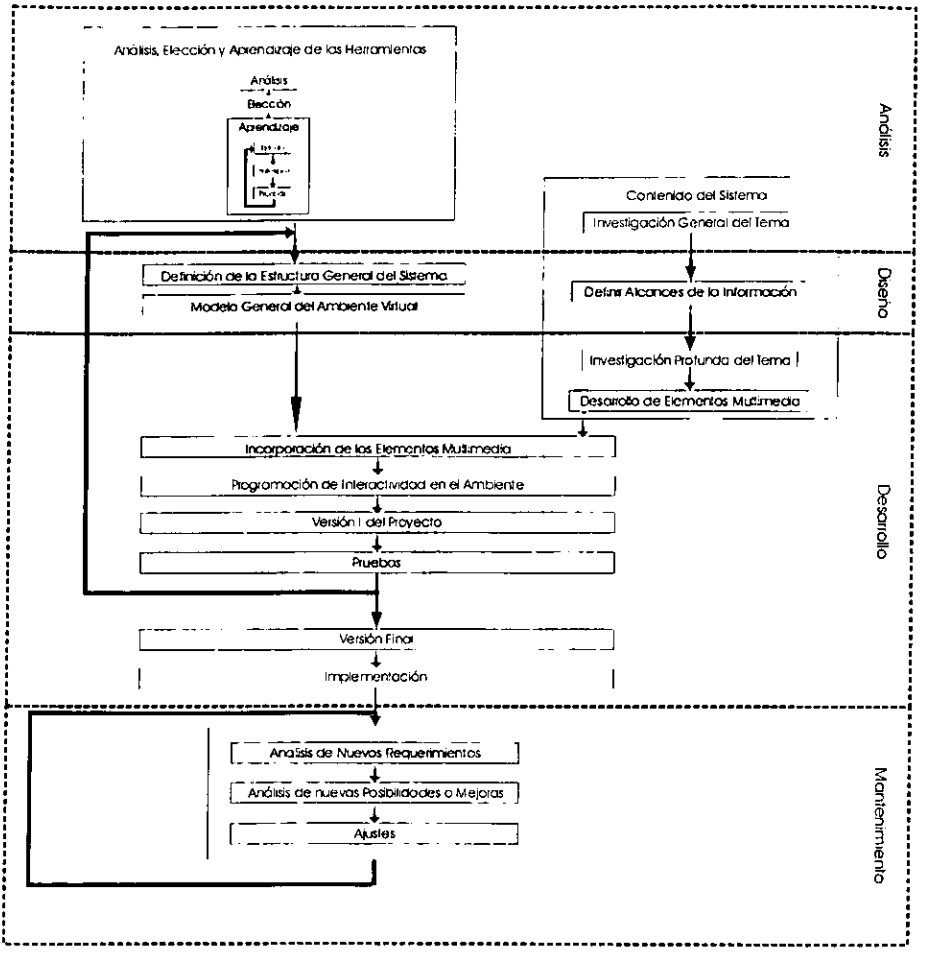


Figura 2.5 Metodología propuesta

El modelo en cascada le da a la propuesta una estructura base; sin embargo, la estructura de cascada es modificada en varios aspectos sobre todo en el desarrollo interno de cada etapa. La propuesta consta de cuatro etapas (ver ***Figura 2.5***), el análisis, el diseño, el desarrollo y el mantenimiento; esto es, la estructura de cascada clásica, sin embargo, en cada una de las etapas se aplican distintos modelos. En la etapa de análisis, se tiene una parte desarrollada con base en el modelo de prototipos. El modelo de versiones sucesivas está aplicado en la forma cíclica en que se realizan el diseño y el desarrollo. Por último, en la

etapa de mantenimiento se aplica el principio del modelo en espiral a fin de mejorar continuamente el sistema. La forma en que estos conceptos son aplicados se explicará con mayor detalle a continuación.

2.2.1 Análisis

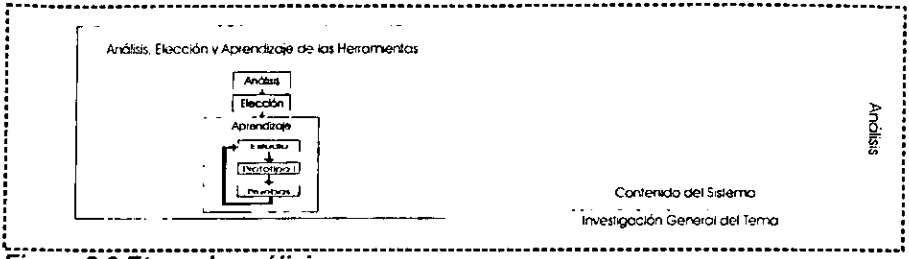


Figura 2.6 Etapa de análisis

En la etapa de análisis se realizan las tareas de aprendizaje de las herramientas así como el análisis del posible contenido del sistema. Estas actividades podrán ser realizadas de forma paralela.

2.2.1.1 Análisis, elección y aprendizaje de las herramientas

Una parte importante del análisis de un sistema de cualquier índole es la definición de los límites que éste tendrá en cuanto a sus capacidades, así como los requerimientos mínimos del mismo.

En proyectos como el que se está desarrollando, esta etapa requiere necesariamente, un proceso de análisis de las herramientas disponibles así como la elección de aquellas que se consideren adecuadas. También es necesario el aprendizaje de las herramientas que serán utilizadas.

Generalmente, los sistemas son desarrollados con herramientas de todos conocidas y por lo tanto la etapa que las analiza es obviada. En los sistemas en realidad virtual, por el contrario, las herramientas no son muy conocidas por lo que es importante estudiarlas bien para conocer sus capacidades y decidir así cuales son las más convenientes.

En lo que se refiere al análisis y la consiguiente elección de las herramientas a utilizar, se debe entender que depende totalmente de las herramientas disponibles en el preciso momento en que se desarrolla el sistema. Por esto, el análisis y las elecciones que se tendrán para el proyecto desarrollado en la presente tesis serán exclusivamente válidas en esta ocasión, ya que muy probablemente las condiciones cambien para cuando se realice el siguiente proyecto.

La etapa de aprendizaje de herramientas abarca dos aspectos importantes para el desarrollo de un sistema en realidad virtual. Por un lado, es imprescindible el conocer perfectamente el programa de realidad virtual que será utilizado, así como la plataforma y el sistema operativo en el que trabaja, para esto ha sido desarrollado el tercer capítulo de la presente tesis. El conocimiento de esto último se hace necesario toda vez que de otra forma no sería posible delimitar los tiempos que cada tarea se llevará durante el desarrollo del sistema. Por otro lado, es importante también conocer las herramientas adicionales que se utilizarán, por ejemplo, los programas para edición de los elementos multimedia.

Como se hace evidente en la ***Figura 2.6*** la etapa de análisis, elección y aprendizaje de herramientas (como bloque) tiene una sola entrada y una sola salida. Esto se debe a que es una etapa que se realiza una sola vez y al inicio del proyecto. Hemos planteado la etapa de esta forma debido a que es necesario conocer bien la herramienta antes de pasar a la etapa de diseño en la cual se definirán las capacidades del sistema. En realidad esto no significa que el programa se conozca a la perfección al momento de salir de esta etapa, pero si deberá conocerse lo suficiente para saber qué puede y qué no puede hacer, aun sin saber de qué manera se logra.

Examinando el contenido del bloque de aprendizaje de herramientas podemos observar que es un bloque hecho mediante la metodología de prototipos.

Se decidió utilizar esta técnica debido a la forma progresiva que tiene el aprendizaje de un programa. En la mayoría de los libros de cualquier tipo de

programas se trabaja, aunque no explícitamente, con este método. La idea consiste en ir conociendo la teoría de las capacidades de un programa y, al mismo tiempo, hacer pequeños proyectos que incorporen los nuevos conocimientos a fin de darles un uso práctico antes de conocer nuevos conceptos. Este método de aprendizaje es bastante viejo en realidad, pero resulta muy adecuado sobre todo cuando se es autodidacta, es decir, cuando el programa debe ser aprendido mediante bibliografía o cuando mucho tutoriales, pero sin la guía constante de un profesor.

La mecánica general de este método se puede resumir en un análisis donde se aprenden conceptos nuevos sobre el programa, una puesta en práctica de estos conceptos (el prototipo) y ciertas pruebas en las cuales se verifican los conocimientos adquiridos y surgen nuevas dudas por resolverse en la siguiente iteración del ciclo.

2.2.1.2 Contenido del sistema

Como puede notarse en la **Figura 2.6** el sistema se divide en sus etapas iniciales en dos aspectos fundamentales, el aspecto técnico y el contenido del sistema, esto es, por un lado tenemos las cuestiones concernientes a la programación y definición del ambiente virtual y por otro, lo concerniente a la información que se expondrá en el sistema, en este caso el estudio de la cultura Maya.

2.2.1.2.1 Investigación general del tema

Como se puede observar en la ***Figura 2.6*** esta etapa viene a ser la etapa de análisis dentro del aspecto de contenido del sistema, en ella se hará una investigación general del tema a desarrollar.

La investigación realizada en esta etapa deberá cubrir todo el tema que se desarrollará aunque no será necesario cubrir cada tema a profundidad. La intención de esta primera investigación es la de conocer todos los aspectos que

podrían ser contemplados en el sistema y tener así los elementos para pasar a la etapa de análisis donde se definirán los temas a profundizar.

Es importante, en esta etapa, recopilar y organizar las diversas fuentes de información; de ellas se obtendrá aquella que se necesite para el sistema. El objetivo de esto es que, al pasar a la investigación profunda, se tenga una idea clara de donde se encuentra la información requerida, y se sepa también qué información está disponible y cuál no.

2.2.2 Diseño

Como se puede apreciar en la ***Figura 2.7*** en esta etapa continuaremos con los dos aspectos que hemos manejado, el aspecto técnico y el del contenido del sistema.

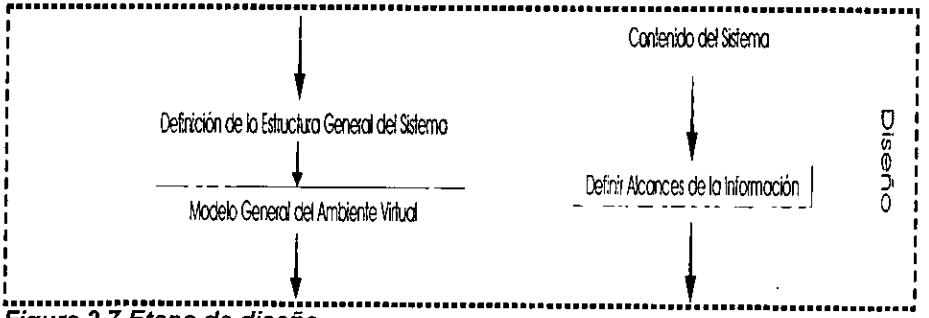


Figura 2.7 Etapa de diseño

2.2.2.1 Aspecto técnico

En lo que se refiere al aspecto técnico tenemos las dos tareas que se explican a continuación.

2.2.2.1.1 Definición de la estructura general del sistema

Debemos entender por esta definición a la especificación de la estructura en cuanto al funcionamiento que el sistema tendrá; es decir, en esta etapa se definirá

la forma en que el sistema funcionará, de que forma interactuará con el usuario y los elementos didácticos (multimedia, menús, textos, etc.) a utilizarse.
Es importante definir en esta etapa, las distintas formas que se utilizarán para exponer el tema, esto se refiere a las herramientas didácticas que se contemplarán, ya sean ventanas para la proyección de videos, tableros para su control, animaciones de objetos tridimensionales, animaciones en video, juegos, audio, etc.

Para la realización de esta etapa es completamente necesario el haber completado la primera etapa del sistema contemplada dentro del análisis y que se refiere al aprendizaje de las herramientas. Esta necesidad provoca una estructura en cascada de esta parte de la metodología, ya que de no conocerse las herramientas suficientemente, no se podrá definir el tipo de elementos didácticos que podrán utilizarse en el ambiente.

Esta etapa forma parte de lo que en otras metodologías se conoce como estudio de factibilidad, este estudio busca "evaluar el costo y el beneficio de la aplicación propuesta e identificar soluciones alternativas, sus costos y sus beneficios potenciales para el usuario"[17]. La parte restante del estudio de factibilidad se contempla dentro de la subetapa investigación general del tema.

Es también en esta etapa donde se comienza formalmente a aplicar una actividad paralela a la realización de las etapas que restan a la metodología. Esta actividad es la *documentación* del sistema y deberá reportar todas y cada una de las etapas subsecuentes a fin de facilitar toda modificación que se le realice a la aplicación una vez finalizada.

[17] Ghezzi, Carlo. Op. Cit. pp. 363

2.2.2.1.2 Modelo general del ambiente virtual

Nuevamente es de destacar la necesidad de cumplir con la etapa previa para pasar a ésta, lo cual hace patente nuevamente el uso del concepto del modelo en cascada en la conformación de esta metodología.

La etapa de definición de la estructura general del sistema nos provee de las herramientas didácticas que podrán ser utilizadas dentro del sistema, esto nos permite definir el ambiente virtual en el que se desarrollará. Estas herramientas nos sirven para presentar en ellas los elementos multimedia, algunos de estos serían las pantallas donde se desplieguen videos, objetos donde se asignen imágenes, animaciones dentro del ambiente virtual, etc. Es importante no confundir estas herramientas con los elementos multimedia que se desarrollarán en otra etapa, estos últimos utilizarán a las herramientas en cuestión para ser presentados dentro del ambiente.

En esta etapa del proyecto se deberá establecer la distribución arquitectónica que tendrá el ambiente virtual, es decir, si se tendrán varias habitaciones, si serán espacios abiertos, de que tamaño, el amueblado en caso de tenerlo, etc.

Se deberá definir también las luces, texturas, colores y todo lo que se refiere al ambiente virtual, el cual quedará completamente definido en esta etapa aunque aún no realizará ninguna función.

2.2.2.2 Aspecto del contenido del sistema

Dentro de la etapa de diseño, y en cuanto al aspecto del contenido del sistema, se cuenta únicamente con una actividad la cual describiremos a continuación.

2.2.2.2.1 Definir alcances de la información.

La información existente respecto a la mayoría de los temas es demasiada para ser abarcado en un sistema de cómputo de cualquier índole, por esto se requiere de una etapa en la que se limiten los temas a tratar. Con esto quedan definidos los requerimientos, en cuanto a información abarcada, que el sistema contemplará.

Cualquier tipo de sistema e inclusive cualquier libro tiene un publico al cual es dirigido – lo que en publicidad se llama su mercado o *target* -. Esto último es muy importante ya que, por un lado, no es el mismo lenguaje con el que se debe dirigir a un niño que a un adulto, y por otro, tampoco es la misma información la que espera un doctor en arqueología de un sistema que la que espera un estudiante de preparatoria. Por ello en esta etapa, además de definir los temas a tratar, se define también la profundidad con que esto se hará.

2.2.3 Desarrollo

Como se mencionó en la etapa de análisis, existen dos aspectos importantes en el sistema, el técnico y el que contempla el contenido del sistema. En esta etapa (ver **Figura 2.8**) contemplaremos subetapas en cada uno de estos aspectos y, adicionalmente, algunas subetapas en las que se unirán ambos aspectos a las que llamaremos fases de integración.

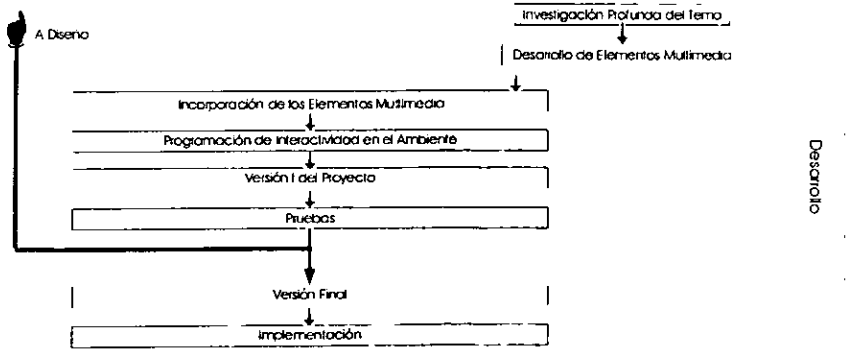


Figura 2.8 Etapa de desarrollo

2.2.3.1 Aspecto del contenido del sistema

2.2.3.1.1 Investigación profunda del tema

Como se recordará en la etapa de Diseño se definieron los alcances de la información que se presentaría en el sistema, en esta etapa se deberá ahondar en los temas particulares que se hubiesen determinado en la anterior etapa.

Se recurrirá a las fuentes de información obtenidas en la etapa de Investigación General del Tema que se realizó en la etapa de análisis.

Es importante también respetar las limitantes en cuanto a la profundidad de la investigación definidas también en la etapa de diseño. Recordemos que la profundidad de la investigación dependerá en gran medida del público al que se dirigirá el sistema.

2.2.3.1.2 Desarrollo de elementos multimedia

En esta etapa se desarrollarán los diversos componentes multimedia que necesitará el sistema. Aquí es evidente, nuevamente, la aplicación del concepto de cascada debido a que es indispensable haber investigado todo lo necesario respecto al tema para poderlo contemplar en los elementos multimedia.

Algunos de los elementos que se contemplan en esta etapa son: Imágenes, Videos, Animaciones y Audios. Esto implica, conocer las diversas herramientas para edición de los mismos. Nosotros recomendamos el uso de ***Adobe Photoshop*** y ***Premier*** para edición de Imágenes y Vídeo respectivamente, el ***Sound Forge*** para editar audio y el ***3D MAX*** para animaciones, estos programas se destacan por la gran calidad que han mostrado y a la gran adaptabilidad que presentan con otros programas y con el hardware en casi cualquier plataforma. En el caso particular del ***3D MAX*** también se recomienda por su compatibilidad con el programa ***dVISE*** en el cual se genera el ambiente virtual. Estos últimos son los

programas utilizados en la realización de este trabajo, sin embargo, quien desarrolle un ambiente puede escoger las herramientas que más le convengan para realizar esta labor.

En esta etapa se deberán realizar las siguientes actividades.

Para la edición de imágenes del sistema se recomienda el **Adobe Photoshop**. Algunas de las actividades que se realizan en él son las siguientes:

Edición de imágenes

Captura de imágenes

Es el punto primordial para la conformación del acervo gráfico del sistema. Se trata de la digitalización de las imágenes disponibles por medio del scanner. Esta tarea implica también decidir cuál es la resolución indicada con el fin de lograr imágenes con un tamaño y calidad adecuados.

Ajustes de tamaño

La gran mayoría de las imágenes que se requieren para el sistema necesitan una forma determinada de acuerdo a los objetos que se tengan en el ambiente tridimensional del sistema. Por otro lado, el origen de las imágenes a utilizar, a excepción de los iconos y etiquetas, es fundamentalmente por medio de la captura de imágenes de libros, folletos, revistas y fotografías. Es evidente que estas imágenes difícilmente se ajustarán a los tamaños requeridos por el ambiente, por esto se hace necesaria la transformación por medio del Adobe Photoshop.

Composición de imágenes

En múltiples ocasiones se requiere formar una sola imagen utilizando varias de ellas. Este es el caso de aquellas imágenes en libros o revistas que se encuentran divididas en varias páginas o segmentos. Para lograr este tipo de composiciones es necesario utilizar el programa en cuestión para crear una imagen nueva y en ella colocar las fracciones de la misma, enseguida se arma la imagen completa y se limita la zona deseada.

Ajustes de color

Otro de los problemas que pueden surgir al adquirir imágenes es que los tonos de color que presentan no sean compatibles entre sí, esto provoca un aspecto desagradable al realizar la composición de las mismas. Se hace necesario, por lo tanto, realizar ajustes de color e incluso de brillo y contraste con el objeto de homogeneizar a las imágenes utilizadas.

Retoque de imágenes

Ésta es probablemente la labor que más tiempo implica ya que se trata de la modificación minuciosa de las imágenes. Dentro de esta labor podemos encontrar la limpieza del fondo de algunas imágenes de tal forma que se puedan incrustar en el ambiente virtual sin que este fondo sea visible. Es decir, si se tiene un objeto rojo, el fondo de la imagen que le será asignada deberá ser rojo también, lográndose así la aproximación de una transparencia.

Creación de la interface de usuario

Independientemente de las imágenes logradas es necesario colocar una interface para el sistema. Para esto se requiere diseñar imágenes que servirán de botones, iconos, menús o etiquetas que se utilizarán en el ambiente en el cual se moverá el usuario del sistema.

Para esto se utiliza la enorme gama de filtros con que cuenta el programa, además de la creatividad necesaria de parte del programador.

Creación de ambiente tridimensional

Esta tarea se puede realizar mediante el programa **3D MAX** Versión 3.0. Es importante hacer notar que la utilidad del programa para el proyecto se limita a la creación del ambiente y en ningún momento se hace necesaria la creación de movimiento dentro del mismo. Como consecuencia de esto último no se atraviesa por la etapa de animación ni se utilizan la herramientas como las cámaras o los sistemas animados predefinidos por el programa.

Las labores que se realizan en esta herramienta son las siguientes:

Creación de los objetos requeridos.

En esta etapa se crean todas las paredes, cuadros, banquetas y de más objetos necesarios para poder armar el ambiente.

Para crear los objetos se utilizan las herramientas predefinidas por el programa y se les transforma mediante operadores booleanos y modificadores, a fin de lograr los objetos con la forma y dimensiones que se requieran.

Asignación de las texturas necesarias para los objetos del ambiente.

Antes de agrupar los objetos para la conformación del ambiente es necesario asignarles las texturas correspondientes, esto es debido a que una vez agrupados los elementos no es posible asignar texturas por separado, ya que se trata de cuerpos completos.

Construcción del ambiente en su conjunto.

Es la tarea que más trabajo requiere durante la construcción del ambiente. Esto es debido a que una vez teniendo los objetos requeridos es necesario agruparlos de tal forma que ajusten unos con otros. Esta labor resulta bastante minuciosa debido a que la mínima falla es muy notoria. La conformación del ambiente se realiza en un proceso de lo pequeño a lo grande, es decir, se van construyendo los objetos más pequeños y con ellos se forman unidades más complejas, esto continúa así hasta lograr el ambiente completo.

Edición del audio del sistema

Para la realización de esta tarea se puede utilizar el programa **Sound Forge**.

Algunas de las aplicaciones que se pueden realizar con esta herramienta son las siguientes:

Audios de narración

Se graban las narraciones de algunos temas importantes para el sistema a fin de ser reproducidas junto con la imágenes respectivas al momento de explicar un tema.

Audios de ambientación

Se refiere a la edición de música que servirá de fondo para las narraciones e incluso como parte del ambiente virtual.

En el programa se realiza la integración de la música con la narración.

Sonidos para eventos

Como parte de la interface de usuario es necesario lograr sonidos pequeños que se ejecuten al momento de suceder ciertos eventos. Esto será contemplado dentro del programa de realidad virtual.

2.2.3.2 Fases de integración

En las siguientes subetapas se realizarán las labores que integrarán a los aspectos técnicos con los del contenido del sistema de tal forma que se logre un sistema completo al finalizar la etapa de desarrollo.

2.2.3.2.1 Incorporación de los elementos multimedia

Evidentemente, será en esta etapa en donde se incorporarán los elementos multimedia, desarrollados anteriormente, con el ambiente virtual. Esta etapa es una de las más importantes debido a que en ella se empieza a dar forma final al sistema ya que al finalizar esta etapa el sistema llevará integrada ya, la información que pretende mostrar.

En esta subetapa se colocarán aquellos videos, imágenes, audios, etc que se pretenda utilizar en los respectivos objetos mediante las propiedades adecuadas que el programa de realidad virtual provea.

2.2.3.2.2 Programación de interactividad en el ambiente

El ambiente, hasta ahora, únicamente se presenta ante el usuario como un lugar para ser recorrido y observado, sin embargo, es necesario proveer de interactividad al mismo a fin de que el usuario tenga la capacidad de controlar el sistema.

Precisamente es en esta subetapa en la que se deberá programar todo lo concerniente a las posibilidades que tendrá el usuario de interactuar con el ambiente, así como las propiedades que tendrá el ambiente virtual modelado en la etapa de diseño.

Es aquí donde se genera el código que permitirá al ambiente comportarse de cierta manera ante la intervención del usuario o la interacción de unos objetos con otros.

Principalmente se requiere dotar de propiedades a todos los objetos que conforman al ambiente virtual; las propiedades que aquí se determinan se refieren a las restricciones de movimiento, propiedades físicas y algunas propiedades de luminosidad y textura que se agregarán a aquellas definidas en el modelo de ambiente virtual definido en la etapa de diseño.

Por ejemplo, será aquí donde se programarán los tableros de controles para que manejen el vídeo, las imágenes o cualquier otro elemento multimedia que se utilice. También se programará aquí lo referente a la navegación del usuario en el sistema.

2.2.3.2.3 Versiones del proyecto

Una vez terminadas las etapas anteriores, se procederá a integrar todo en una versión del proyecto. Esta versión deberá ser completamente funcional y se le dará el número de versión de acuerdo con la iteración en la que fue generada.

A esta versión se le deberá someter a una serie de pruebas que el desarrollador determinará a fin de decidir si será necesario hacer una siguiente iteración. Estas pruebas estarán encaminadas a detectar posibles errores dentro del sistema, mismos que serán corregidos en la siguiente versión. Otro tipo de prueba es la llamada validación ante el usuario; esta consiste en presentar el sistema ante el usuario final y considerar sus observaciones.

Como se puede observar en la ***Figura 2.8***, al finalizar las pruebas se tiene una liga hacia la etapa de diseño en donde se volverán a realizar estas etapas tomando en cuenta prioritariamente los errores detectados en las pruebas.

Es de destacarse la aplicación que se hace en este punto del concepto de versiones sucesivas estudiado anteriormente.

2.2.3.2.4 Versión final e implementación

Una vez hechas las correcciones necesarias y pasadas las pruebas sin detectar nuevos errores, se presentará la versión final del proyecto. Esta versión deberá cubrir todos los requerimientos iniciales, se presentará ante el usuario, en caso de ser aprobada, será la que se presente e implemente ante el cliente.

2.2.4 Mantenimiento

En esta etapa se realizan las labores de cuidado y asistencia al sistema para un futuro.

Muchos proyectos son entregados y en este momento se termina el compromiso con el cliente y con el proyecto mismo; sin embargo, es importante darle esta asistencia a los proyectos a fin de que se conserven vigentes e inclusive puedan crecer en un futuro.

En todo proyecto debe tomarse muy en cuenta el mantenimiento, ello debido a que representa el 60% del costo del proyecto. Existen cuatro tipos de mantenimiento; éstos son el preventivo, correctivo, adaptativo y perfectivo.

Respecto al mantenimiento preventivo, es de gran importancia ya que permite detectar posibles fallas antes de que se presenten lo que aumenta la fiabilidad del producto.

El mantenimiento correctivo[18]*, tiene que ver con la eliminación de los errores aún presentes en el producto cuando éste es entregado, así como los errores que se suman durante el mantenimiento del software.*

El mantenimiento adaptativo y perfectivo son la verdadera fuente de cambio en el software, éstos motivan la introducción de la capacidad de evolucionar como una cualidad fundamental del software y la anticipación de cambios como un principio general que debe guiar a la ingeniería de software. El mantenimiento adaptativo ocupa cerca del 20% del costo del mantenimiento mientras que el 50% es ocupado por el perfectivo. El mantenimiento adaptativo involucra ajustar la aplicación a los cambios en el ambiente, por ejemplo, una nueva versión de hardware, del sistema operativo o un nuevo sistema de base de datos. En otras palabras, en el mantenimiento adaptativo la necesidad de cambios en el software no pueden ser atribuidas a una característica del software en sí, tales como la presencia de errores acarreados o la incapacidad de proveer de algunas funciones requeridas por el usuario. Más bien, el software debe modificarse debido a que el ambiente en el que trabaja presenta cambios.

Finalmente, el mantenimiento perfectivo involucra cambiar el software para mejorar alguna de sus cualidades. Aquí, los cambios son debido a la necesidad de modificar las funciones ofrecidas por la aplicación, agregar nuevas funciones, mejorar el rendimiento de la aplicación, hacerla más fácil de usar, etc.

[18] Ghezzi, Carlo. Op.Cit. pp.25-26

La petición de realizar el mantenimiento perfectivo puede venir directamente del ingeniero de software, con el fin de mejorar la situación del producto en el mercado o puede venir del cliente para especificar nuevos requerimientos.

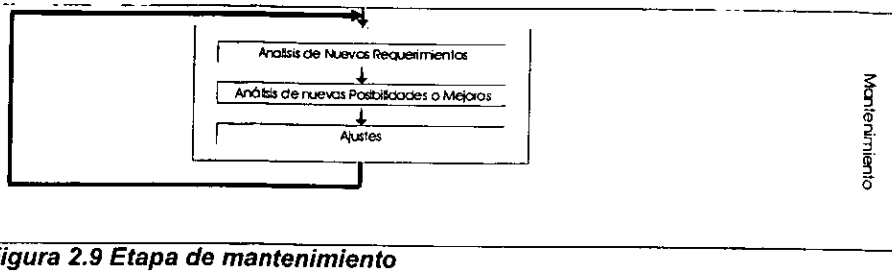


Figura 2.9 Etapa de mantenimiento

2.2.4.1 Análisis de nuevos requerimientos

Una vez teniendo el sistema funcionando, el usuario final podrá proponer posibles requerimientos que no fueron contemplados en el sistema actual.

Estos requerimientos se tendrán en cuenta en una siguiente iteración de un ciclo que se iniciará en la etapa de diseño.

2.2.4.2 Análisis de nuevas posibilidades o mejoras

En esta subetapa lo que se analiza es la posibilidad de incrementar las capacidades del sistema mediante un estudio de las nuevas condiciones de la tecnología o bien simplemente por renovar el sistema en cuanto a estética o funcionalidad.

2.2.4.3 Ajustes

En esta etapa se modifican todo aquello que haya sido contemplado en las etapas que analizan los nuevos requerimientos y las mejoras. Como se puede observar en la ***Figura 2.9***, al finalizar la etapa de mantenimiento se tiene una liga que nos lleva al inicio de la misma. Esto, evidentemente, implica que la etapa se repetirá indefinidamente. En esta etapa, se aplica claramente, el concepto de espiral ya

que se producen versiones mejoradas en cada ciclo. Este ciclo sólo podrá finalizar en caso de que se decida desarrollar un nuevo proyecto que sustituya al presente.

3 Herramientas para el desarrollo del sistema

3.1 Hardware

Para realizar un sistema de realidad virtual, es necesario contemplar los elementos de hardware con los que se cuenta. En este caso explicaremos aquellos con los que se cuenta en el Laboratorio de Interfaces Inteligentes de la Facultad de Ingeniería de la UNAM.

3.1.1 Dispositivos de entrada

El dispositivo de localización utilizado es el Polhemus Fastrack. El Polhemus Fastrack, de la serie 3SPACE, es un dispositivo electromagnético de la compañía Polhemus, el cual proporciona actualizaciones a una velocidad de 120 Hz. La mayoría de los paquetes de software de desarrollo de realidad virtual tiene controladores para los sensores de Polhemus. Algunas desventajas que presenta el dispositivo son; tiene un alcance limitado por la potencia del emisor; su nivel de error es proporcional a la distancia, así como a la presencia de objetos metálicos grandes; además, tiene un tiempo de respuesta a los movimientos del usuario de 10 ms.

El dispositivo de control utilizado para este sistema, y disponible en el Laboratorio de Interfaces Inteligentes es el Division 3D Mouse. Se trata de un joystick con el cual es posible ofrecer al usuario un control de su punto de vista en el entorno, además proporciona la posibilidad de manipular los objetos y provocar algunas

La petición de realizar el mantenimiento perfectivo puede venir directamente del ingeniero de software, con el fin de mejorar la situación del producto en el mercado o puede venir del cliente para especificar nuevos requerimientos.

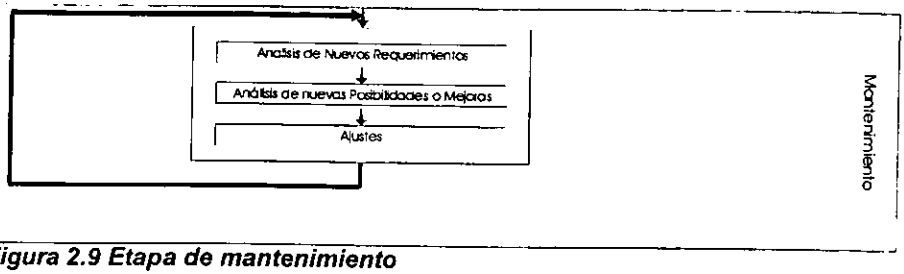


Figura 2.9 Etapa de mantenimiento

2.2.4.1 Análisis de nuevos requerimientos

Una vez teniendo el sistema funcionando, el usuario final podrá proponer posibles requerimientos que no fueron contemplados en el sistema actual.

Estos requerimientos se tendrán en cuenta en una siguiente iteración de un ciclo que se iniciará en la etapa de diseño.

2.2.4.2 Análisis de nuevas posibilidades o mejoras

En esta subetapa lo que se analiza es la posibilidad de incrementar las capacidades del sistema mediante un estudio de las nuevas condiciones de la tecnología o bien simplemente por renovar el sistema en cuanto a estética o funcionalidad.

2.2.4.3 Ajustes

En esta etapa se modifican todo aquello que haya sido contemplado en las etapas que analizan los nuevos requerimientos y las mejoras. Como se puede observar en la ***Figura 2.9***, al finalizar la etapa de mantenimiento se tiene una liga que nos lleva al inicio de la misma. Esto, evidentemente, implica que la etapa se repetirá indefinidamente. En esta etapa, se aplica claramente, el concepto de espiral ya

acciones al disparar eventos. Este dispositivo tiene integrado el sistema de localización electromagnética de la compañía Polhemus[19].

La unidad de control es la encargada de retroalimentar las entradas y resultados del sistema detector de posición y del Division 3D Mouse a la estación de trabajo a través de puertos seriales. En este tipo de dispositivos es importante el concepto de latencia, mismo que se refiere al tiempo que requiere el sistema para responder a los movimientos del usuario.

3.1.2 Dispositivos de salida

El dispositivo de presentación utilizado en el sistema es el casco de visualización VR4 de la compañía Virtual Research. Es un casco de casi un kilogramo de peso que presenta la imagen en dos pantallas de cristal líquido mientras bloquea el contacto visual con el mundo exterior, lo que produce un buen efecto de inmersión. Usando el casco de visualización, la navegación por el ambiente es una operación natural; no es estereoscópico,[20] pues no se cuenta con un equipo de cómputo que soporte dos tarjetas de video, una para cada display. Se considera que con un campo de visión de 80 a 85 grados se obtiene una buena inmersión.

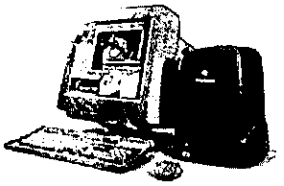
El dispositivo de audio usado está incluido en el casco de visualización VR4 a través de los audífonos. Independientemente de los dispositivos aquí mencionados, existen otros muchos dispositivos que permiten incrementar la sensación de realidad; algunos de ellos son los llamados de retroalimentación táctil, de esta forma es posible percibir el ambiente mediante el sentido del tacto además de los sentidos de la vista y el oído.

---

[19] Stampe, Dave, et al. Realidad Virtual. Creaciones y Desarrollo. Ed. Anaya Multimedia. S.A, Madrid, 1995, pp. 84-85.

[20] La Estereoscopia se refiere a la capacidad de presentar las imágenes que percibe cada uno de los ojos en las pantallas respectivas.

Finalmente, mencionaremos la computadora en la que se realizó el sistema. Se trata de una estación de trabajo O2 de Silicon Graphics. Es ideal para el diseño tridimensional y animación al contar con un sistema que combina el desempeño de Silicon Graphics con un poderoso procesador MIPS R5000 RISC que ofrece un alto desempeño en operaciones de punto flotante y enteros.



La computadora[21] tiene una arquitectura de 64 bits, una velocidad de 180 Mhz, y, sobre todo, una velocidad de rendereo de 380 triángulos por segundo aproximadamente. Esto último es lo que le da el poder para desplegar realidad virtual. Su única limitante se encuentra en su memoria RAM que es de 64 MB.[22]

[21] Imagen obtenida en la página de Silicon Graphics http://www.sgi.com
[22] Canché Rodríguez, Claudia. Creación de un sistema en R.V, Tesis de Ingeniería en Computación, México 1999, F.I. UNAM. pp. 37-41.

3.2 Software

Además de las herramientas de hardware, es importante conocer el software con el que se puede trabajar a fin de poder escoger el más adecuado para el sistema a desarrollar ya que, a diferencia del hardware, existe disponibilidad de una gran cantidad de paquetes.

3.2.1 Edición de imágenes.

El programa Adobe Photoshop es utilizado para la edición de imágenes de mapas de bits. Con él es posible lograr modificar cualquier imagen de acuerdo a las necesidades específicas de la aplicación que las utilizará. También es posible crear ilustraciones con las diferentes herramientas de dibujo de que dispone.

Este programa trabaja mediante los llamados layer o capas de dibujo. Estas capas se encuentran una sobre otra y sólo se puede trabajar sobre una de ellas. El orden en que están colocadas puede ser cambiado y cada capa tiene propiedades como la transparencia. Las mencionadas capas pueden realizar diversos efectos; éstos consisten en sombras de diferentes tipos que actúan sobre todo aquello que esté contenido en esta capa.

Una capa puede ser de texto o de mapa de bits y no está restringida en espacio por el especificado en la imagen de trabajo. Es decir, una capa es una imagen de cierto tamaño independiente al tamaño de la imagen de trabajo; a esta capa se le coloca en cierta parte de la imagen y, aún si excede los límites, se conserva completa al ser guardada en el formato .psd de Adobe Photoshop.

Cuando una capa es de tipo texto, éste permanece editable, de tal forma que se le puede modificar tanto en el contenido mismo como en su formato. Una capa de tipo texto puede ser convertida a tipo mapa de bits, pero dejará de ser editable.

Una o más capas pueden ser integradas entre sí. Un ejemplo de esto es cuando se guarda en un formato de mapa de bits, en este momento se integran todas las capas y se guardan sólo en el espacio delimitado por la imagen de trabajo.

El Adobe Photoshop puede trabajar en distintos sistemas de colores. Se tiene el sistema RGB, el CMYK, la escala de grises, y otros.

Todos estos sistemas son de mapas de bits y consisten en la codificación de cada punto o pixel de la imagen en su respectivo sistema.

Los sistemas RGB y CMYK son los más utilizados en la industria para codificar una imagen a color y se basan en la combinación de los colores primarios para el sistema respectivo. En el sistema RGB se combinan el Rojo, el Verde y el Azul y en el sistema CMYK se combinan el Cyan, el Magenta, el Amarillo y el Negro.

El sistema de escala de grises, como su nombre lo indica, codifica la imagen en niveles de gris y trabaja con 256 niveles.

Finalmente hay otro aspecto que hace especial la forma de trabajar del Adobe Photoshop. Se trata de su navegador, esta herramienta nos permite manejar el acercamiento que tengamos de nuestra imagen de una manera mucho más sencilla ya que te permite ver en qué parte de la imagen se ubica el acercamiento actual de tal forma que no se pierde la perspectiva del conjunto. Además permite desplazar al acercamiento libremente sin necesidad de alejarse y volver a acercarse en un área distinta.

El Adobe Photoshop tiene múltiples herramientas con las que se puede trabajar una imagen, o más bien, una capa de una imagen. Una de las principales es la herramienta de selección. Con esta herramienta se puede escoger una o más zonas especificas de la capa y trabajar sin afectar al resto de la capa. Para esto se cuenta con selección por colores, por áreas geométricas definidas, por zonas elegidas a mano libre o mediante la herramienta de paths o rutas.

Otra de las principales herramientas con las que cuenta el Adobe Photoshop es la goma. En este programa, a diferencia de otros, tiene sentido hablar de goma ya que el sistema de capas le permite tener transparencia en todo el espacio de la capa que no tenga color. Si se trabajara en una sola superficie no podríamos hablar de transparencia ya que una imagen de mapa de bits común y corriente no acepta valores nulos para ningún pixel de la misma. La única excepción a esta regla se presenta con el formato GIF89 en el cual se puede definir un color transparente, pero aún en este caso se tiene la desventaja de que este color ya no puede ser utilizado, esto sin mencionar la baja calidad del formato.

El programa cuenta con algunas herramientas de dibujo con las que se pueden realizar dibujos a mano libre. Estas herramientas pueden ser utilizadas con la brocha o pincel que se desee, a éste se le puede modificar el punto, tanto en tamaño, en intensidad, en color e incluso se le pueden dar efectos a la región afectada por la brocha.

Se tienen también algunas herramientas extra como el gotero, que sirve para tomar un color específico de la imagen, y la cubeta que sirve para iluminar una área definida con un mismo color.

Una de las principales cualidades del programa es su basta cantidad de filtros sin mencionar todos los que se le pueden agregar que son producidos por otras compañías. Estos filtros son formas en que se puede alterar una imagen o parte de una imagen.

3.2.2 Creación de ambiente tridimensional

KINETIX 3D MAX 3.0 se utiliza para la creación de ambientes en tercera dimensión y para crear animaciones dentro de los mismos.

Cuando se trabaja en este programa se pasa por diversas etapas y el programa tiene las capacidades para realizar con gran calidad cada una de ellas.

Estas etapas se pueden dividir de la siguiente manera: **Modelado**, **Texturización** y **Animación**.

El **modelado** consiste en la construcción de la estructura del ambiente. Dentro de esta tarea se encuentran la creación de los objetos y su modificación a fin de lograr las formas deseadas para el ambiente tridimensional.

El programa 3D MAX en su versión 3 cuenta con una basta cantidad de objetos predefinido que el usuario puede utilizar a su gusto. Esto es, por ejemplo, si se desea un cubo no se requiere dibujar un cuadro mediante cuatro líneas y después darle volumen, en realidad sólo se utiliza la herramienta cuadro y se le dan las dimensiones deseadas. De la misma forma que el cuadro, se tienen definidas herramientas para esferas, cilindros, pirámides y hasta una tetera.

Estos objetos pueden ser modificados de muy diversas formas. Una de ellas es mediante su escalamiento y rotación. Otra de ellas es mediante los múltiples modificadores que provee el programa, entre los que se encuentran aquellos que doblan, tuercen y deforman de muchas maneras a los objetos.

Otra forma de crear objetos es mediante una forma o figura en dos dimensiones a la que se le puede dar volumen. Estas formas también son predefinidas por el programa y para usarlas sólo se necesita tomarlas y darles las dimensiones requeridas.

Las formas, bidimensionales, al igual que los objetos tridimensionales, pueden ser deformados mediante los modificadores con los que cuenta el programa.

Hasta el momento, hemos descrito los objetos tangibles que podrían existir dentro del ambiente tridimensional deseado, sin embargo, existen otros objetos de suma importancia para el ambiente que no pueden ser vistos directamente sino a través de sus efectos, se trata de las luces.

Las luces son importantes ya que proveen de una diferente visibilidad y apariencia a los objetos de una forma complementaria a la textura que tengan asignada los mismos. Como en la vida real, independientemente del color que un objeto tenga, nosotros lo percibiremos de diferente manera si lo ilumina un flash intenso de luz blanca, que si apenas le llega una luz neón roja.

El 3D MAX provee tres tipos de luces: la spot, la direccional y la omnidireccional. La primera de ellas se refiere a la típica luz en cono, se define mediante un punto, una dirección y un ángulo de apertura del cono. La luz direccional se produce, no desde un punto, sino desde un área circular y se difunde en una sola dirección y un solo sentido. Finalmente, la luz omnidireccional se define únicamente como un punto del que se emite una luz en todas direcciones.

Otro de los objetos intangibles que pueden formar parte de un ambiente tridimensional en 3DMAX son las cámaras. Estas son objetos que definen un punto de observación de la escena o ambiente tridimensional. Las cámaras pueden tener o no un objetivo fijo. Aquellas que no lo tienen son llamadas libres y son utilizadas en los casos en que será movida durante la animación.

La utilidad de las cámaras estriba principalmente en las distintas perspectivas que nos pueden dar de la escena. Sería imposible mover todos los objetos de una escena para poderlos ver desde una perspectiva distinta cada vez que se quisiese. En lugar de esto se coloca una cámara en cada lugar específico, desde el cual nos interesa observar la escena.

La siguiente etapa de la construcción de ambientes tridimensionales es la **texturización** y se refiere a la elección de las texturas que utilizaran los objetos del ambiente. El 3D MAX cuenta con una herramienta específicamente para esta labor llamada editor de materiales.

En esta herramienta, se definen texturas; esto se hace eligiendo distintos colores o imágenes para ser manejados cada uno de distinta forma y lograr, entre todos, una textura integral que podrá ser asignada al material. Estas texturas no sólo se

refieren a la apariencia que tendrá el objeto sino a ciertas propiedades que definirán la forma en que las luces se comportarán al contacto con los objetos a los que sean asignados.

La ultima etapa para el desarrollo de una animación completa es precisamente la **animación** del ambiente tridimensional ya construido. Para esta etapa el 3D MAX, al igual que la mayoría de los programas que realizan animaciones en dos o tres dimensiones, se ayuda de los llamados keyframes o cuadros llave. Los cuadros de una animación son cada una de las imágenes consecutivas que la conforman al ser presentadas de forma continua. En los cuadros llave se definen posiciones, texturas y otras propiedades especificas para cada uno de los objetos de la escena y el programa calcula, automáticamente y por medio de la interpolación las mismas propiedades para todos los cuadros intermedios.

3.2.3 Edición del audio del sistema

Sound Forge 4.5 es utilizado para la edición de audio. Esto es, en este programa se puede modificar un archivo de sonido y adecuarlo así a las necesidades que se tengan en la aplicación.

El programa tiene capacidad para abrir archivos en distintos formatos. El más común de ellos es el formato .wav. Este formato es el utilizado por Windows y permite conservar el audio con máxima calidad. Otros formatos de uso común son .aif, .au, .snd y .voc. Todos estos formatos son soportados por Sound Forge y es posible trabajarlos en él. Es importante mencionar que existe otro formato muy utilizado para reproducir música que utiliza la extensión .mid. Sin embargo, éste no es un formato de audio propiamente dicho ya que se trata de un código de notas que son interpretadas por algunos programas y reproducen el respectivo sonido de la misma forma que lo hace un sintetizador.

Este programa tiene la posibilidad de seleccionar partes del audio de tal forma que se puedan recortar, pegar, copiar o borrar.

Estas selecciones también tienen la utilidad de que se les pueden aplicar diferentes distorsiones, cambiar la amplitud o ecualizar zonas en específico.

Otro de los aspectos fundamentales del Sound Forgue es su capacidad para manejar dos canales de audio al mismo tiempo, lo que nos permite mezclar dos audios o un audio y una voz en un mismo archivo de sonido.

Algunas otras herramientas que tiene el paquete son la posibilidad de fades o desvanecimientos.

Es necesario destacar la capacidad de cambiar los formatos, velocidad de muestreo y cantidad de bits a utilizar por muestra, de tal forma que se modifica la cantidad de espacio de memoria que utilizará el archivo, al mismo tiempo que se fija la calidad del audio.

3.2.4 Creación y edición de video

Adobe Premiere es utilizado para la edición de video en el ámbito casero y semiprofesional. Sus capacidades son importantes en cuanto al manejo que se puede hacer de imágenes y audio, aunque su principal problema a escala profesional es la falta de calidad en su salida a video y evidentemente la linealidad de su edición. Los programas utilizados para televisión, por ejemplo, normalmente son editores no lineales, característica con la que no cuenta el Adobe Premiere.

A nivel semiprofesional, sin embargo es la herramienta más utilizada. Su funcionamiento se basa, como en el caso del Photoshop, en capas o en este caso los llamados canales de audio y de video.

El Adobe Premiere trabaja con 2 canales de video y 3 de audio. En los canales de video se colocan desde imágenes fijas hasta secuencias de video en diversos formatos. En uno de los canales de que se dispone en Adobe Premiere, es posible colocar 2 videos pero éstos no podrán estar al mismo tiempo. La utilidad de esta

división del canal es la posibilidad de presentar transiciones de un video o imagen a otra.

Las transiciones son efectos que el programa tiene predefinidos para cambiar de una imagen a otra. Algunos de los más populares son las disolvencias o los cambios de página en diferentes estilos.

Como en el caso del Photoshop, los canales de este programa pueden presentar transparencias para dejar ver, en ciertas partes de la imagen, al canal que se encuentra atrás. Para lograr calidad en estas transparencias es extremadamente útil utilizar imágenes que traigan la transparencia en el formato .psd de Photoshop, el cual es plenamente reconocido por el programa por obvias razones.

Los canales de audio, por otro lado, proveen la posibilidad de modificar la amplitud del audio mediante una línea que lo controla.

La línea de tiempo que maneja el programa es sin duda alguna la parte más importante del mismo. En ella se puede estructurar el proyecto ya que es posible controlar todos los canales tanto de audio como de video de manera sencilla. El tamaño del bloque colocado en esta línea nos muestra perfectamente el tiempo durante el cual se mantendrá presente el elemento en cuestión.

Los elementos con que se cuenta para estructurar el proyecto los podemos encontrar en la llamada ventana de proyecto. De esta ventana podemos obtener las propiedades de estos elementos y podemos llevarlos directamente hacia la línea de tiempo para incluirlos en un canal determinado.

Evidentemente el programa tiene su propio formato para guardar los proyectos hechos en el mismo. Cuando un video no ha sido terminado es necesario guardarlo en este formato ya que sólo así se mantendrá la estructura del mismo y se requerirán los elementos que lo conforman por separado.

En el caso de tener un proyecto terminado es necesario exportarlo a video de tal forma que sea posible visualizarlo en cualquier computadora. Algunos de los formatos de video que soporta este programa son .avi, .mov de Quick Time, Gif animados y .flc. Una vez elegido el formato en que se guardará el video, aún falta elegir el compresor de video que determinará la calidad del video que se generará y la calidad de audio, que dependerá de la velocidad de muestreo y la cantidad de bits por muestra tomada. Algunos de los compresores de video más comunes son los de Microsoft, Cinepak y aquellos de Intel.

3.2.5 Programación del ambiente virtual

3.2.5.1 dVISE

A continuación se intenta dar una guía básica sobre el uso del programa dVISE en el cual se realizará el ambiente virtual de este proyecto y se darán los fundamentos para comenzar a utilizarlo de modo que, con la ayuda de los manuales dVISE *User Guide*, *dVS User Guide y dVS Geometry Tools User Guide*, el usuario logre familiarizarse con el programa y explotarlo en todo su potencial.

No se pretende dar una descripción técnica del programa ni del sistema operativo sobre el que se utiliza, para esto se hace referencia a los manuales antes mencionados.

El programa dVISE de la compañía *DIVISION* es un medio ideal para la creación de ambientes virtuales con una programación basada en ventanas y sin la necesidad, aunque es posible, de programar código directamente, más bien generándolo como lo hacen los lenguajes de cuarta generación.

Para utilizar el programa se requieren algunos preparativos:

Primero que nada, se requiere tener acceso a una cuenta con los permisos necesarios, esto es, en el caso del Laboratorio de Interfaces Inteligentes, tener acceso a la cuenta "DVS", la cual permite la ejecución de los programas asociados con DIVISION.

El árbol de directorios del programa está organizado tal y como se muestra en la **Figura 3.1**.

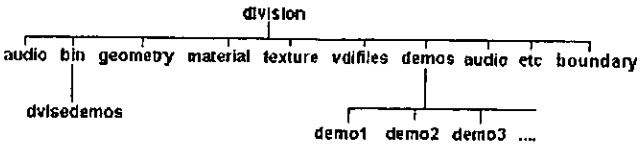


Figura 3.1

En el directorio **audio** se encuentran los archivos de audio que el ambiente virtual podrá utilizar. Para los archivos de radiación se utilizará el formato **.arf** y para los audios en general se utilizará un formato que dependerá del sistema operativo y la configuración del hardware utilizado.

En el directorio **geometry** se encuentran los archivos a utilizar, como objetos y se utilizan en la propiedad *Visual* en el control panel. El formato en este directorio es **.bgf** y son los modelos tridimensionales que por lo general se importan de algún programa de modelado en 3D.[23]

En el directorio **material** se encuentran los archivos que definen las texturas que los objetos de *Geometry* llevarán. Esto se refiere al color, brillo, transparencia, etc. que el objeto tendrá. El formato en el que se encuentran estas texturas es **.bmf**.

En el directorio **texture** se encuentran las imágenes que se asignan a los objetos en vez de una textura del directorio *material*, en este caso los archivos llevan el formato **.vtx** que por lo general son importados de formatos como **.bmp** y **.tga** mediante las herramientas de dVISE.

En el directorio **vdifiles** se encuentran los archivos que describen el ambiente virtual. Este archivo es el único indispensable para llamar al ambiente virtual ya que en él se define todo, incluidas las rutas y nombres de los archivos de los demás directorios que son utilizados. Evidentemente el formato de estos archivos es **.vdi.** Al convertir archivos de escenas en **.3DS** mediante las herramientas de *DVS* se puede obtener directamente un archivo **.vdi** y el árbol de directorios sobre

[23] *DVS Geometry Tools User Guide*.http:// www.division.com.

el cual se puede empezar a trabajar. En este mismo directorio se encuentran las librerías que este mundo virtual utilizará (las librerías serán descritas en el capítulo referente al *Control Panel*) y éstas son guardadas en un formato **.vdi**.

En el directorio **etc** se encuentran archivos de configuración que mapea el software en la configuración del hardware; estos archivos son descritos en el *DVS User Guide*. En el directorio se encuentran también los archivos *body* cuyo formato es **.bod**. Éstos definen la forma en que el usuario interactúa con el ambiente virtual en cuanto a los dispositivos con los que recibe y envía información al usuario, estas configuraciones son las desplegadas en el menú de inicio al correr el *dvisedemos* (este directorio no deberá ser incluidp en el árbol de directorios de cada ambiente, ver directorio *demos*).

En el directorio **boundary** encontramos los archivos que definen los límites de una zona utilizada (las zonas serán descritas en el apartado referente al *Control Panel*) en el ambiente. El formato en este tipo de archivos es el mismo que en los archivos en *geometry*, es decir **.bgf**.

El directorio **demos** es un directorio que arbitrariamente colocamos para guardar los ambientes virtuales que se realizarán. En realidad cada usuario deberá colocar su directorio demos en el directorio raíz de su propia cuenta.

El programa dVISE trabaja a partir de un archivo llamado dvisedemos que busca a partir de donde se encuentra, los archivos con extensión **.dvsdemo**. Esta búsqueda se realiza a partir de un directorio que se especifica en el código del archivo *dvisedemos* en la siguiente línea:

set demo_dir = /usr/people/zaldivar/mundos

Por esto se debe tener un directorio único en el que se guarden todos los ambientes a mostrar en el *dvisedemos*; cada ambiente deberá estar contenido en un directorio independiente a fin de no mezclarlos. En el directorio de cada ambiente se requiere, por lo tanto, un archivo **.dvsdemo** en el cual se establece la

descripción del ambiente, el nombre del directorio en que se encuentra, el nombre del archivo .vdi. El archivo .dvsdemo tiene las siguientes líneas:

```
set description = " descripción del ambiente "
set demo_dir_name = "nombre_del_directorio_del_mundo"
set vdi_file = "vdifiles/nombre_del_archivo.vdi"
set dvise_exe = DEFAULT
set toolbox_file = DEFAULT
```

En las últimas dos líneas se determinan dos variables de ambiente cuyo valor deberá dejarse en *DEFAULT*.

A partir del "directorio del mundo" se deberá desarrollar el mismo árbol de directorios descrito anteriormente.

En el directorio **BIN** del árbol de directorios de DIVISION se encuentra el programa *dvisedemos* (este directorio no deberá ser incluido en el árbol de directorios de cada ambiente, ver directorio demos) que al ser ejecutado presenta un menú como el siguiente:

```
1) tutorial - Demonstration of dVISE features (Audio)
2) camaro - Camaro car showroom (Audio)
3) car_suspension - Car chassis suspension demo
4) dmi - Dismounted Infantry Demo (Audio).
5) engine - Combustion engine animation using dVISE keyframes
6) house - House architectural walkthrough
7) kitchen - Interactive kitchen model
8) lab - Manikin Demonstration
9) molecule_room - Molecular modeling
10) navy_train - Fire sprinkler training procedure
11) porsche - Porsche car & engine
12) production_line - A production line of dVS bottles (Audio)
13) tank - Programmable tank demo (Audio)

Menu Control:

a) dVISE control panel = off   -> select 'a' to switch on
b) Toolbox          = on    -> select 'b' to switch off
c) VR display       = window -> toggle: dvisor/vr4/nvision/boom
d) Tracking Device  = edit2d -> toggle: mouse/ipu/s.ball/s.mouse
e) Audio Device     = off   -> toggle: default/dbi300/atron2
q) Quit

Select option:
```

La opción 1 siempre será la del tutorial ya que esta no puede ser eliminada de este menú. Este tutorial es de gran ayuda a fin de comprender el funcionamiento del programa sin necesidad de crear antes un ambiente.

Las opciones del 2) al 13) son los ejemplos que provee el programa. Como se mencionó anteriormente, esta parte del menú lista los ambientes existentes a partir de la ruta definida en el archivo *dvisedemos*. Por lo tanto, en realidad listará los ambientes de cada usuario.

Las opciones de la **a)** a la **e)** se cambian de valor con la letra indicada y presionando **RETURN**, de tal forma se pasa a la siguiente opción de las indicadas frente al inciso.

La opción **a)** se refiere al control panel y si se desea trabajar con él, lo cual es recomendable al empezar a conocer el programa, se debe tener en **ON**.

La opción **b)** se refiere al **toolbox** y es recomendable colocarla en la opción **OFF** mientras se familiariza con el programa.

La opción **c)** se refiere a la salida visual del programa, es aquí donde se determina si se utilizará el casco o la pantalla (**window**). Si el dispositivo que aquí se escoge es el casco **(vr4)** se le estará habilitando para tomar las coordenadas del usuario a partir de lo que indique el sistema de localización del casco. Si por el contrario, se elige **window** se estará indicando que la localización será tomada por otro medio, esto es independientemente de que la salida del vídeo puede ser dirigida al casco por medio del sistema operativo. La localización determinada en este inciso se refiere únicamente en cuanto a la rotación ya que lo referente al avance o retroceso se indica en el siguiente inciso.

La opción **d)** se refiere a la entrada que recibirá el programa en cuanto a la distancia hacia delante o hacia atrás. La opción que se utiliza para el joystick es

edit2d, sin embargo, mientras se trabaja en el modelado del mundo es adecuado utilizar la opción Mouse para utilizar el ratón.

La opción **e)** se refiere al dispositivo mediante el cual se emitirá el audio.

La opción **q)** sólo deberá ser utilizada si se desea salir del programa.

Mientras el usuario se empieza a familiarizar con el programa, la forma en que las opciones deben ser ajustadas es la siguiente:

Menu Control:

```
a) dVISE control panel   = on       -> select 'a' to switch off
b) Toolbox               = off      -> select 'b' to switch on
c) VR display            = window   -> toggle: dvisor/vr4/nvision/boom
d) Tracking Device       = mouse    -> toggle: ipu/s.ball/s.mouse/edit2d
e) Audio Device          = off      -> toggle: default/dbi300/atron2
q) Quit
```

Una vez ajustando estas opciones se está listo para correr cualquiera de los ejemplos que se presentan en el menú y de esta forma, al teclear el número deseado y el **RETURN**, se ejecuta un mundo virtual en el cual se puede conocer en buena medida el uso del programa dVISE.

Si se desea, se pueden modificar las opciones elegidas por *default* a fin de no tener que cambiarlas cada vez que se entra al programa. Para modificarlas es necesario cambiar el código del archivo *dvisedemos* en las siguientes líneas:

```
set dvise_control = "on"
set toolbox       = "on"
set audio         = "off"
if ($OS != "UNIX_SV")    then
set display        = "windows"
set tracking      = "mouse"
else
set display       = "dvisor"
set tracking      = "ipu"
endif
```

Las primeras tres líneas definen si se habilitan el control panel, el *toolbox* y el dispositivo de audio. Las siguientes líneas definen dispositivos de despliegue y de desplazamiento de acuerdo al sistema operativo en el que se encuentra.

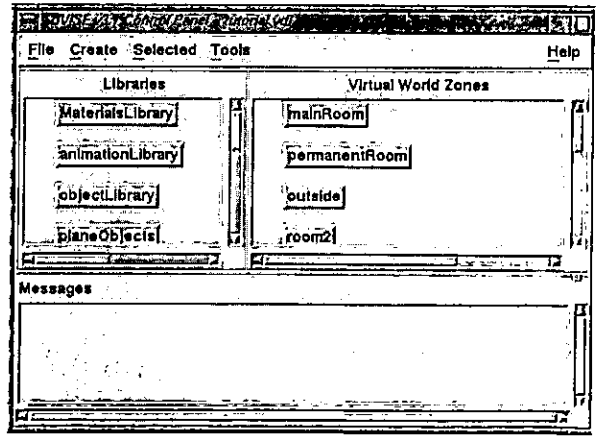
Una vez dentro de cualquier ambiente virtual se puede salir de él presionando la tecla CTRL + C en la ventana de la sesión del *shell* desde donde se ajustó el menú.

3.2.5.1.1 Control panel

El mundo virtual en este programa puede ser manejado por medio de dos herramientas básicas, el *control panel* y el *toolbox*. Inicialmente explicaremos el funcionamiento del primero de ellos.

Este programa se basa en objetos, propiedades, eventos y acciones; con estos cuatro elementos se puede describir básicamente la estructura del mismo. Un mundo virtual construido en dVISE se conforma por tres ventanas y un menú principal.

Las ventanas son: La ventana de librerías, la de zonas del mundo virtual y la de mensajes.



En la primera de ellas encontramos la librería de materiales así como librerías que son utilizadas para las llamadas instancias, las cuales describiremos posteriormente.

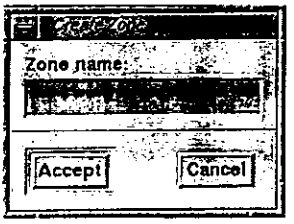
En la segunda ventana se encuentran las diferentes zonas que tendrá el mundo. Las zonas a su vez tendrán objetos en ellas y éstos pueden ser vistos al dar *doble click* en la zona deseada.

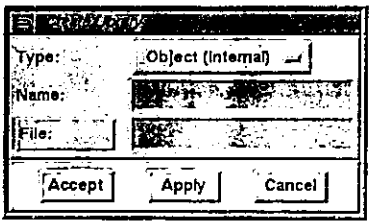
En la tercera zona se encontrarán los mensajes que se enviarán desde las acciones que serán programadas mediante los eventos de que dispone el programa y que serán vistos a detalle posteriormente.

En *el menú* se tienen las siguientes opciones: *file*, *create*, *selected* y *tools*.

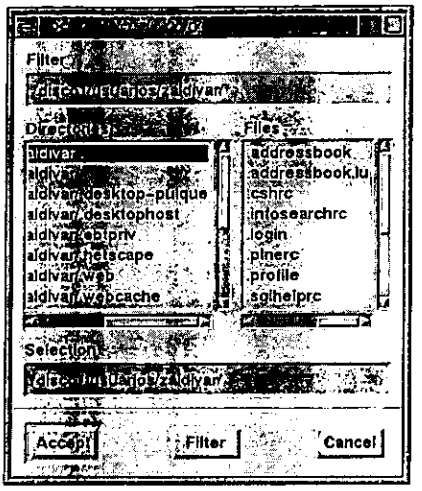
En el menú *file* se encuentran las opciones tradicionales como *save*, *save as*, *close* y *exit*.

En el menú *create* encontramos las opciones zone y library que evidentemente sirve para crear zonas y librerías respectivamente.



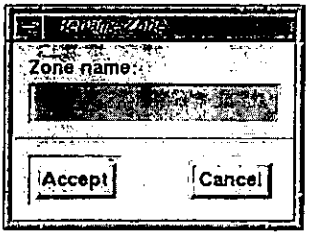


En caso de querer elegir una librería desde un archivo se despliega la siguiente ventana.

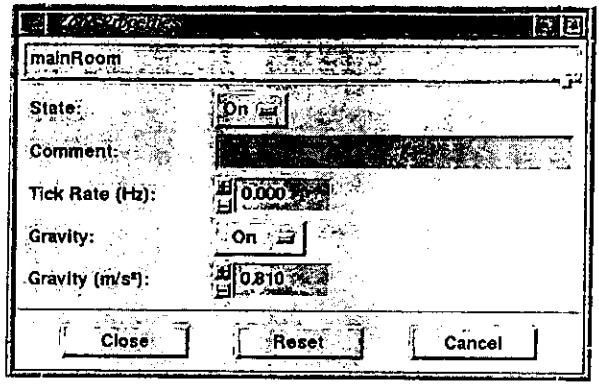


En el menú *Selected* tenemos las opciones *View*, *Rename*, *Behavior*, *Properties*, *Send event* y *Close*. Con estas opciones, que se aplican a la zona o librería seleccionada podemos:

- *View*.- Ver la zona o librería a detalle con el editor respectivo.
- *Rename*.- Renombrar la zona o librería.

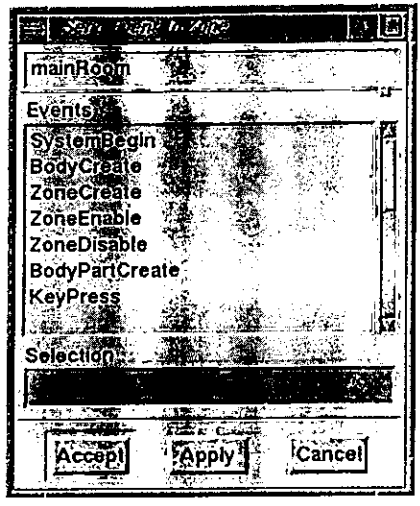


- *Behavior*.- Adjudicar acciones a los eventos que pueden darse con la zona seleccionada. Esta ventana será explicada a detalle dentro del menú selected del editor de zonas.
- *Properties*.- Atribuir valores a las propiedades de la zona. Estas propiedades son: State, Comment, Tick Rate, Gravity, Gravity (m/s^2).



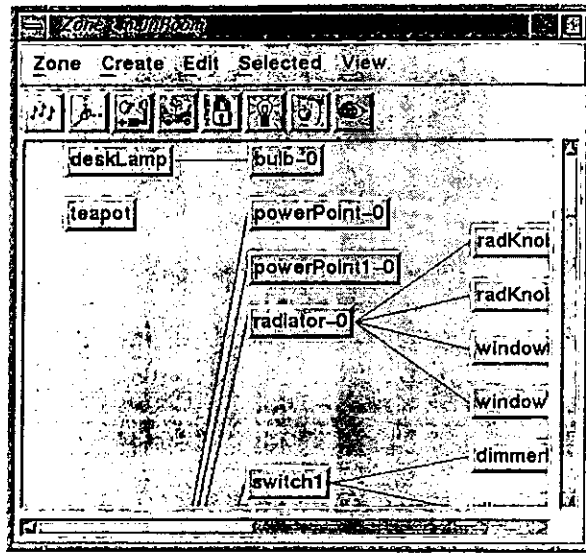
- ✓ *State:* Se refiere a activar o desactivar la zona. Cuando ésta se deshabilita, todos los objetos en ella se deshabilitan y se dispara el evento *ZoneDisabled*. Si algún usuario se encuentra dentro de la zona, ésta no podrá ser deshabilitada.
- ✓ *Comment*: Se utiliza para agregar un comentario.
- ✓ *Tick Rate*: Se utiliza para determinar la velocidad de refresco de la pantalla. Cuando existe una animación es conveniente colocarla en el valor *default*, 0, ya que esto indica que se debe refrescar la pantalla cada vez que se renderea un cuadro de la animación; si se da una velocidad mayor, se estarán desperdiciando procesos, mientras que si se le da menos se podrían perder cuadros. Cada vez que se actualiza la pantalla se dispara también un evento *Tick*, por lo cual la velocidad asignada aquí también representa la frecuencia con la que este evento será disparado.
- ✓ *Gravity*: Mediante esta opción se determina si en la zona en cuestión existirá una acción gravitatoria o no.
- ✓ *Gravity (m/s^2)*: Aquí es donde, en caso de existir la acción gravitatoria, se define el valor que tendría. En un ambiente real este valor sería de 9.81 m/s^2.

➢ *Send Event.*- Mediante esta opción se dispara el evento que se desee. Este evento es escogido en la caja de diálogos y será asociado a la acción que se desee.



En el caso de utilizar la opción *view* del menú *selected*, que es equivalente a dar *doble click* sobre la librería o zona deseada, se abre el *editor* respectivo ya sea de *zonas* o de *librerías*.

Editor de zonas.



En el editor de zonas podemos observar los diferentes objetos que la componen en una ventana.

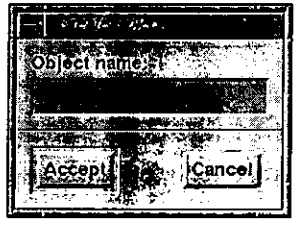
El menú de este editor se compone de las siguientes opciones: *zone*, *create*, *edit*, *selected* y *view*.

- En el submenú *Zone*, tenemos las opciones *Load*, *Behavior*, *Properties*, *Send event* y *Close*.
  - La opción *load* se refiere a cargar una zona de otro ambiente para traerla a éste.
  - La opción *Behavior*, tiene la misma función que la opción *Behavior* en el submenú *Selected* de la ventana principal del panel de control, teniendo seleccionada la zona en cuestión. Las opciones de esta ventana se explican en la opción *Behavior* de las propiedades de un objeto y funciona de la misma manera para el caso de una zona.
  - La opción *Properties* nos permite adjudicar valores a las propiedades de la zona. Esto es equivalente a la opción *Properties* en el submenú *Selected* de

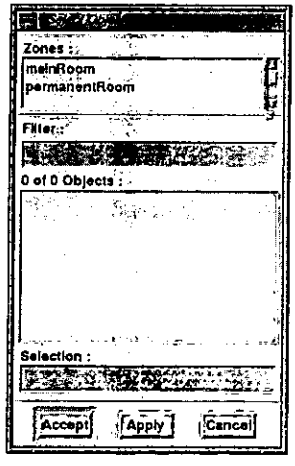
la ventana principal del panel de control, teniendo seleccionada la zona en cuestión.

- La opción *send event* envía un evento al cual se le pueden asignar acciones. Esto es equivalente a la opción *send event* en el submenú *selected* de la ventana principal del panel de control, teniendo seleccionada la zona en cuestión
- La opción *close* cierra el editor de zonas.

- En el submenú *create* se tiene la única opción llamada *object*. Sirve para agregar un objeto a la zona en que se está trabajando. Este nuevo objeto entra con todas sus propiedades deshabilitadas por *default*.

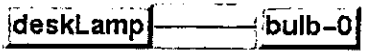


- En el submenú *edit* se tienen las opciones *select*, *cut*, *copy*, *paste* y *delete*.
  - La opción *select* nos presenta la lista de objetos para seleccionar alguno. Esta opción es la alternativa a seleccionar el objeto con el mouse.



  - Las opciones *cut*, *copy*, *paste* y *delete* son por todos conocidas.
- En el submenú *selected* se tienen las opciones *rename*, *link*, *unlink*, *toggle visibility*, *properties* y *send event*.

- La opción rename sirve para cambiar el nombre al objeto seleccionado.
- La opción *link* sirve para ligar dos objetos. Para lograr esto se debe seleccionar primero al objeto que se desea que quede como hijo, después ir a la opción *link* y finalmente seleccionar al objeto que quedará como padre. Una vez hecho esto, el objeto hijo se moverá a la derecha del padre en la ventana y aparecerá una línea uniendo a ambos objetos.

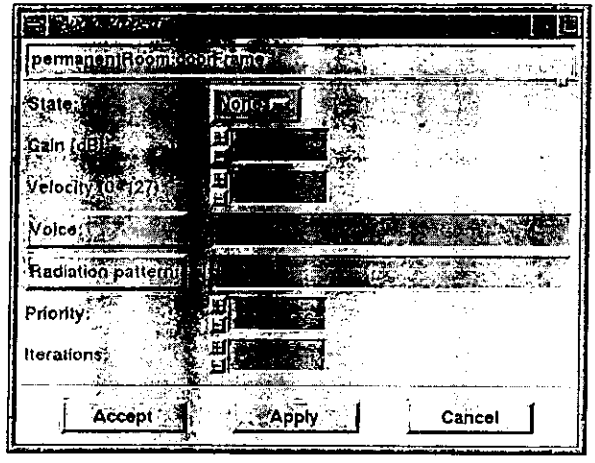


- La opción *unlink* se utiliza de la misma forma que la opción *link* y sirve para eliminar la liga entre dos objetos. Hecho esto desaparece la línea que los une y el hijo se reubica de acuerdo a las demás ligas que tenga. En caso de no tener otras ligas el objeto queda solo.
- La opción *toggle visibility* permite intercambiar entre tener el objeto habilitado o deshabilitado. Esto es equivalente a poner en *off* la opción *visibility* en la propiedad *visual* del objeto, sin embargo, los cambios hechos con el *toggle visibility* no son guardados en el archivo **.vdi**, es decir son temporales.
- La opción *properties* nos permite adjudicar valores a las propiedades del objeto. En dVISE, todos los objetos tienen todas las propiedades, sólo que algunas pueden estar deshabilitadas dependiendo del tipo de objeto.

Propiedades de los objetos

Las propiedades que un objeto puede tener son todas aquellas características que lo describen. En dVISE, las características o propiedades que se le pueden asignar a un objeto son las siguientes:

- *Audio*: Aquí se definen los efectos de audio que un objeto puede tener.



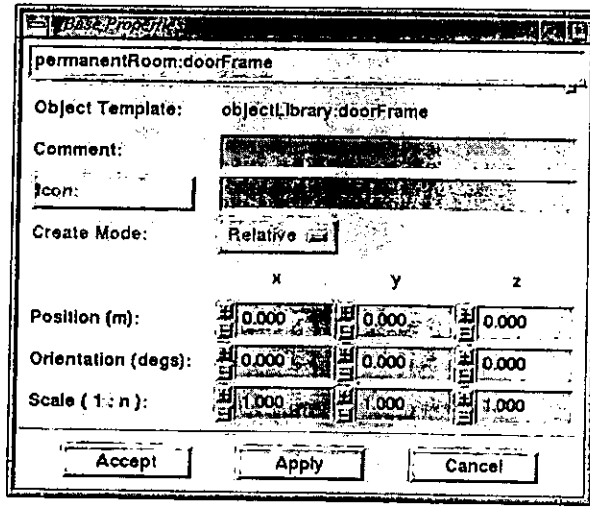
Los valores a asignar son los siguientes:

- *State*: Donde se define si tendrá o no efectos de sonido.
- *Gain*: Se asigna el volumen del audio en decibeles. El valor por *default* es 1.0.
- *Velocity*: Se refiere a la velocidad en que una nota es emitida en el caso de las secuencias MIDI. Los valores aceptados van entre 0 y 127. Por *default* la velocidad es de 63.
- *Voice*: En este campo se define el nombre del archivo de audio a reproducir. El formato del mismo dependerá del sistema, *véase DVS User Guide*. Para seleccionar el archivo se utiliza la ya conocida ventana de diálogo de dVISE.
- *Radiation Pattern*: En este campo se define el nombre del archivo de radiación. El archivo de radiación define los niveles de sonido que se irradian desde el objeto hacia fuera tomando el eje z como origen. El

formato de este archivo es **.arf** y su contenido es descrito en el *DVS User Guide*. Al igual que la opción voice, permite seleccionar el archivo con la ventana de diálogo acostumbrada.

- *Priority*: En este campo se asigna la prioridad que este sonido tendrá con respecto a otro, en caso de ser simultáneamente disparados. La prioridad puede tener valores desde 0 hasta 127, de mayor a menor prioridad. Existe un valor especial que es el –1 que determina que este audio no puede ser interrumpido. El valor por *default* es 0.
- *Iterations*: En este campo se determina el número de veces que un sonido debe ser reproducido. Cuando se da el valor de –1 el sonido será reproducido continuamente. El valor de *default* es 1.

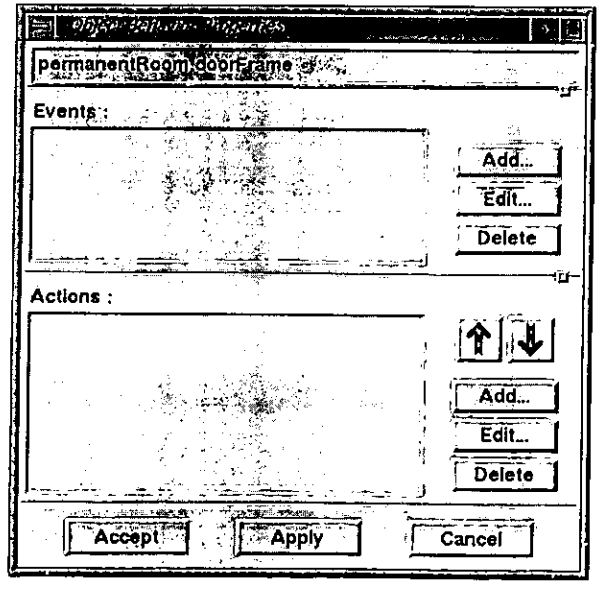
- *Base*: Aquí se define la ubicación del objeto dentro del mundo, tanto su posición como su orientación, también se define la escala a la cual el objeto se encuentra; esto es equivalente al tamaño.



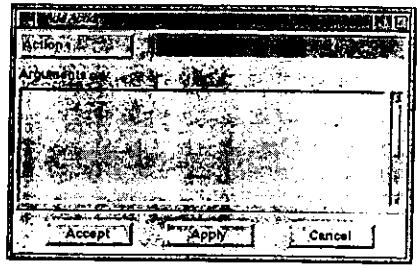
- *Comment*: Aquí se puede colocar un comentario respecto al objeto, de uso exclusivo del programador.
- *Icon*: En caso de estar en el editor de Librerías se tienen los llamados templates. En estos casos es posible adjudicar un icono a fin de que éste aparezca en la ventana de librerías. Nuevamente, la selección del archivo es con la ventana de diálogo de dVISE.

- *Create Mode*: En el caso de tener un template, en este campo se especifica si al crear una instancia del mismo se deberán tomar las coordenadas dadas de forma absoluta o relativas al mouse pointer.
- *Position*: Evidentemente aquí se definen las coordenadas en metros de los objetos. En dVISE todas las coordenadas de objetos son absolutas.
- *Orientation*: En este campo se define el ángulo respecto a cada uno de los tres ejes que tiene el vector dirección del objeto. Esto es importante en caso de que el objeto emita una luz direccional o cuando el objeto no es simétrico. Los valores se dan en grados.
- *Scale*: En este campo se dice el factor de escala del objeto respecto a cada uno de los ejes coordenados.

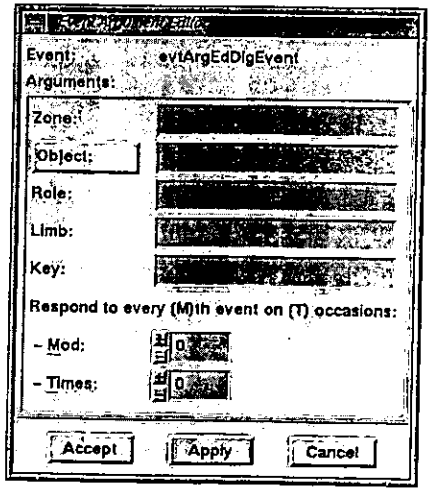
- *Behavior*: Es una de las propiedades más importantes tanto para las zonas como para los objetos ya que en ella se define el comportamiento que tendrán ante determinados eventos. Al igual que en el caso de las zonas nos permite asociar acciones a los diferentes eventos que el objeto en cuestión dispara.



La ventana de diálogo de esta propiedad presenta en la parte superior a los eventos que dispara el objeto. En la parte inferior se despliegan las acciones que el evento seleccionado tiene asociados. Por *default*, esta propiedad no tiene eventos ni acciones seleccionadas, éstas se seleccionan con el botón "add" tanto en el caso de los eventos como en el de las acciones.



Si se desea crear un evento no listado por dVISE en la caja de diálogo, se escribe el nombre del nuevo evento en la caja de texto de selección.

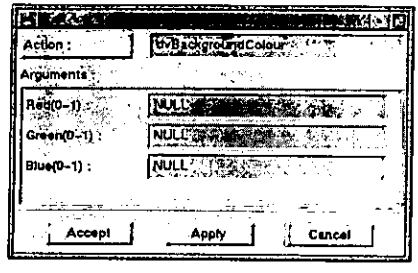


Los eventos tienen varios parámetros mismos que son ajustados mediante la ventana que se obtiene al presionar *edit*. Los parámetros de un evento son los siguientes:

- *Zone*: Nombre de la zona en que se encuentra el objeto.
- *Object*: Nombre del objeto al que se le está asociando el evento.

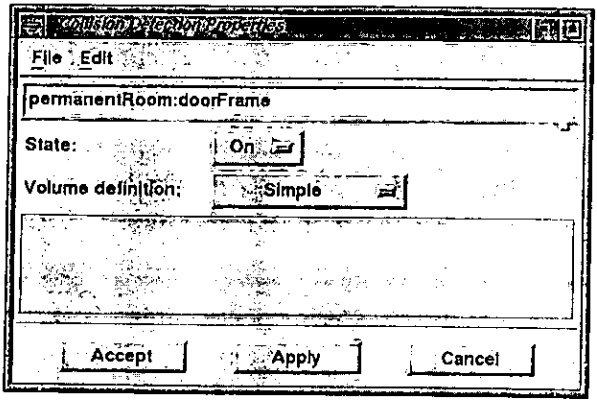
- *Role*: Se coloca el nombre del rol que se asigna a este evento. De acuerdo con el rol que esté jugando el usuario se disparará o no el evento. Para mayores detalles sobre los roles *ver el* dVISE *User Guide*.
- *Limb*: Se coloca el nombre de la parte del cuerpo a la que se asocia el evento. Para detalles respecto a la definición de *bodies ver el DVS User Guide*.
- *Key*: Se coloca el carácter del teclado al que responderá el evento. Este campo sólo se utiliza en los eventos KeyPress y KeyRelease.
- *Mod*: Restringe la frecuencia con la que una acción será disparada por el evento.
- *Times*: Especifica qué cantidad de veces será disparada la acción por el evento.

Cuando se agrega una acción a un evento la ventana de diálogo también requiere ciertos parámetros:



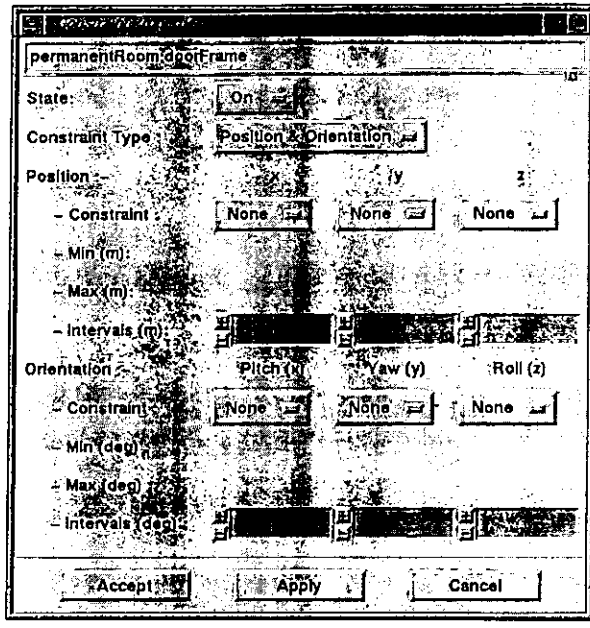
- *Action*: Se escribe o selecciona la acción a tomar.
- El resto de los argumentos depende exclusivamente de la acción seleccionada. Los eventos y las acciones definidas en dVISE se explicarán más adelante.

- *Collision*: Aquí se definen las propiedades en cuanto a que un objeto pueda ser considerado sólido, tangible o manipulable. Esto responde tanto en el caso de contacto con otros objetos como en el contacto con el usuario. Se definen en la caja de diálogo los siguientes datos:

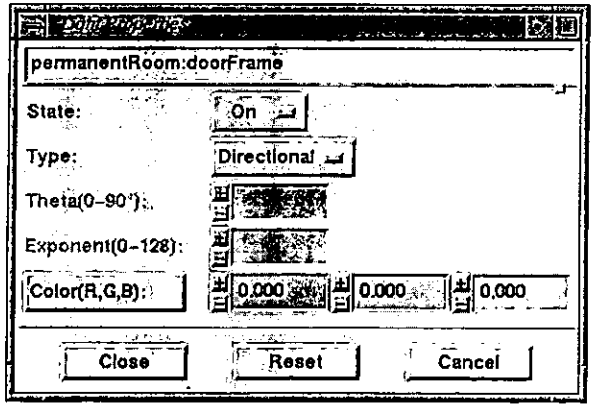


  - *State*: Se define si el objeto tendrá la propiedad o no.
  - *Volume definition*: Se define si el objeto será definido poligonalmente, esféricamente, rectangularmente, etc. Todo esto para efectos de las colisiones. Dependiendo de la selección hecha en este campo se muestran opciones adicionales. Algunas posibilidades en este campo son Simple, Bounding Box, Explicit Box, Explicit Sphere, etc.
- *Constraint*: Esta propiedad define las restricciones en el desplazamiento o rotación del objeto. En su ventana de diálogo se define, como siempre, si tiene o no la propiedad, las restricciones en posición y en orientación para cada eje coordenado y finalmente el intervalo, si lo hay, en que puede desplazarse.

ESTA TESIS NO SALE
DE LA BIBLIOTECA



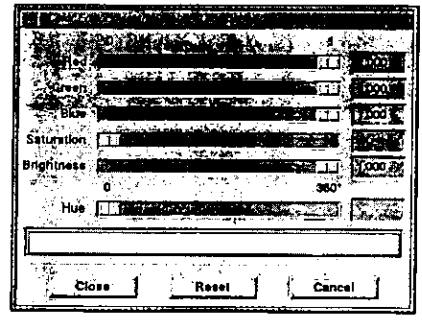
- *Light*: En esta propiedad se le dan las características de iluminación a un objeto, en el caso de ser una luz únicamente, ésta sería la única propiedad que se le daría al objeto.



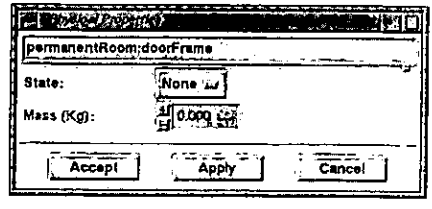
Esta ventana tiene los siguientes campos:

- *State*: En este campo se define si existe o no esta propiedad.

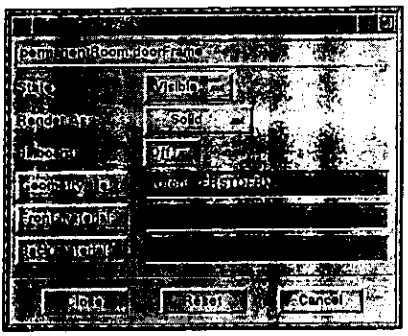
- *Type*: Aquí se define el tipo de luz que el objeto emitirá. Estos tipos pueden ser *ambient*, para luces ambientales, *directional*, que se emite en la dirección en que se orienta el objeto, *point*, que se emite desde el objeto hacia todas direcciones y *spot*, que hace un cono desde el objeto en la dirección con que está orientado y con una apertura theta.
- *Theta*: Es el ángulo con el que la luz de tipo *spot* abre su cono.
- *Exponent*: En las luces tipo *spot* define el exponente de la función que define el desvanecimiento que la luz tendrá. Este exponente tiene un rango de valores de 0 a 128.
- *Color*: En este campo se define el color de la luz mediante el *editor de color*.



- *Physical*: En esta propiedad se definen, por un lado, si el objeto tendrá propiedades de gravedad o no, y por otro, qué masa tendrá.



- *Visual*: En esta propiedad se define si el objeto será visible o no y bajo qué parámetros lo será. Estos parámetros son los siguientes:



  - *State*: Sólo define si el objeto será visible o no.
  - *Render As*: Se determina la forma en que el objeto será desplegado, ya sea de forma *sólida*, es decir con su textura, o bien en *wireframe*, esto significa que sólo las líneas que definen al objeto se despliegan.
  - *Billboard*: Existe un tipo especial de forma de despliegue que no se encuentra en las opciones de *Rendering Mode*, en este caso el objeto se desplegará en la misma forma independientemente de donde se encuentre el usuario, ya que no hace caso de las sombras que se generen sobre él. Al colocar esta opción en *ON* la opción elegida en *Rendering Mode* se deshabilita.
  - *Geometry File*: En este campo se define la estructura geométrica del objeto en cuestión. El formato de este tipo de archivo es **.bgf**.
  - *Front Material*: En este campo se coloca el nombre del material que el objeto tendrá en la superficie de enfrente. El formato de estos materiales es **.bmf**.
  - *Back Material*: En este campo se coloca el nombre del material que el objeto tendrá en la superficie de atrás. El formato de estos materiales es **.bmf**. Estas últimas tres opciones se seleccionan mediante la ventana de diálogo que ya conocemos.

- La opción *send event* envía un evento al cual se le pueden asignar acciones. Esta opción a su vez tiene dos posibilidades, *XPick* y *Other*. La opción *Xpick* define un evento especial y sólo es enviado al seleccionar esta opción. Ya que se le pueden asociar acciones como a cualquier otro evento, esta opción es útil al desarrollar ambientes virtuales para probar algunas acciones en el momento que se desee. En el caso de *Other* la opción funciona de la misma forma que en menús anteriores, es decir, envía un evento seleccionado como si éste se estuviera llevando a cabo.

- El submenú *view* tiene las opciones *toggle tree* y *local*.
  - La opción *toggle tree* permite switchear entre desplegar o no el árbol de objetos en la ventana del editor. Cuando se tiene el árbol desplegado, esta opción lo contrae al nivel raíz del árbol; por el contrario, si se encuentra contraído, esta opción lo expande nuevamente.
  - La opción *local* sirve para desplazar la ventana a fin de que el objeto seleccionado aparezca en ella.

Eventos y acciones

El programa dVISE reconoce múltiples eventos y acciones que a continuación clasificaremos y describiremos en forma general. Para una información detallada de los eventos y acciones favor de dirigirse a la documentación oficial.

Eventos

Los eventos que se reconocen son de tres tipos fundamentales. Aquellos que tienen que ver con lo que le sucede a los objetos, aquellos que se refieren a eventos sobre el usuario y finalmente a eventos que se producen en el ambiente virtual.

Acciones

Las acciones se dividen de manera similar a los eventos. Se tienen acciones que se refieren a cambiar propiedades de los objetos. Con este tipo de acciones se puede, por ejemplo, cambiar la textura de un objeto o accionar una animación. También se pueden definir nuevas acciones que cambien las propiedades de los objetos.

Existen también acciones que manipulan animaciones previamente definidas. Con estas acciones se pueden hacer, por ejemplo, controles de reproducción de algunas animaciones.

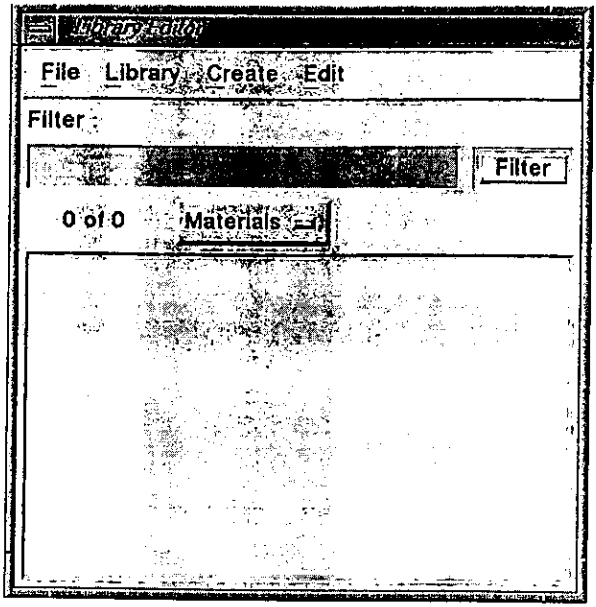
Otro tipo de acciones son aquellas con las que se manejan las propiedades del usuario, algunos ejemplos de estas propiedades son, la ubicación de inicio dentro de la zona, o el tamaño del usuario.

El ambiente también tiene propiedades que pueden ser manipuladas por las acciones. Algunas de ellas son la niebla que existe en el ambiente, el color de fondo, la gravedad, etc...

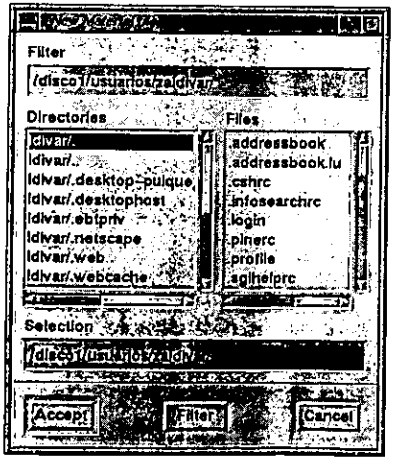
Finalmente existe un último grupo de acciones que maneja tanto variables como expresiones condicionales a fin de enviar eventos, ya sea a objetos, a zonas, o al sistema mismo, todo esto de acuerdo con lo evaluado en las condiciones. También es posible asignar valores a variables que posteriormente pueden ser evaluadas.

Editor de librerías de materiales

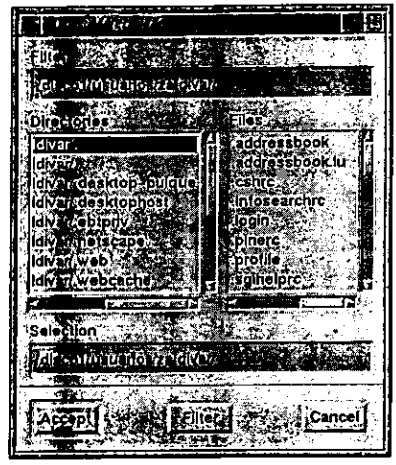
Este editor, que surge en el momento que se desea seleccionar materiales de una librería o bien crear un nuevo material presenta una ventana con el siguiente menú:



- *File*, en este submenú tenemos las opciones:
  - *New*: Sirve para crear un nuevo material

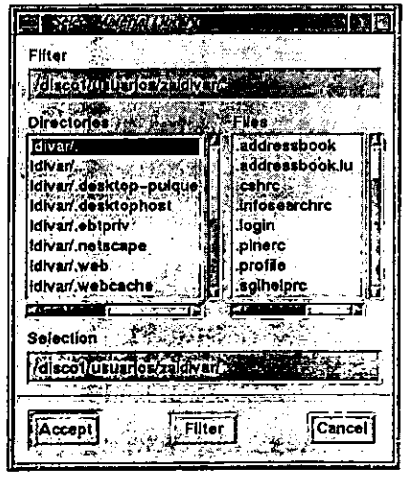


- *Open*: Con esta opción se abre un material existente en la librería.



- *Save*: Sirve para guardar los cambios hechos a un material.
- *Save* as: Sirve para guardar el material modificado en un nuevo archivo.[24]
- *Close*: Con esta opción se cierra la ventana del editor.

- *Library*, en este submenú se tiene el siguiente comando:
  - *Load*: Sirve para abrir una librería de materiales.

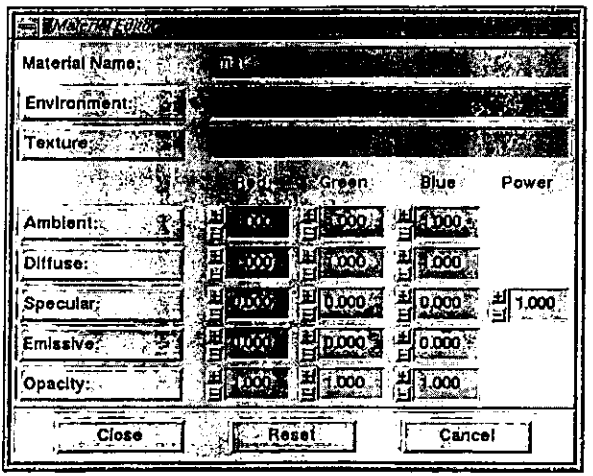


[24] La ventana de diálogo es exactamente igual para las opciones Open y Save As.

- *Create*, en este submenú las opciones son las siguientes:
  - *Material*: Para crear un nuevo material dentro de la librería que está siendo editada.
  - *Texture*: Para crear una nueva textura.
- *Edit*, en este submenú las opciones son las siguientes:
  - *Editor*: Esta opción a su vez da la posibilidad de abrir ya sea el editor de materiales o el editor de texturas.
  - *Duplicate*: Hace un duplicado del material seleccionado y lo coloca en la parte más baja de la lista de materiales de la librería.
  - *Delete*: Elimina el material o textura seleccionado de la librería.

Editor de materiales

En esta ventana se definen las propiedades de un material.
Se conforma de los siguientes campos:

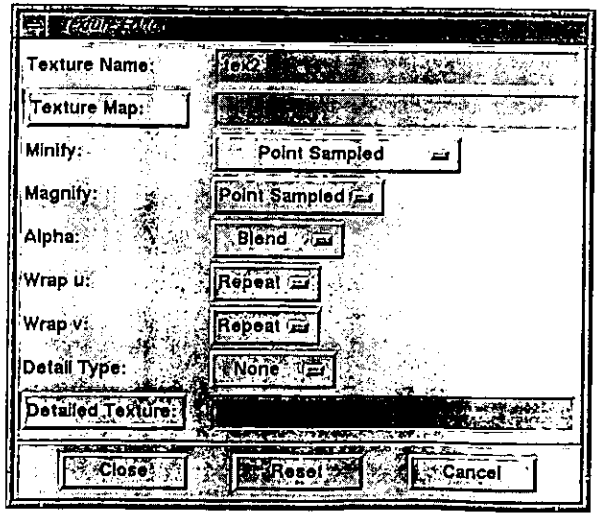


- *Material Name*: En este campo se coloca el nombre del material.
- *Environment*: En este tipo de mapeo, la textura se ve afectada por las sombras de tal forma que su rendereo dependerá de la posición del usuario.
- *Texture*: En este tipo de mapeo, la textura le es aplicada al objeto independientemente de la posición del usuario.

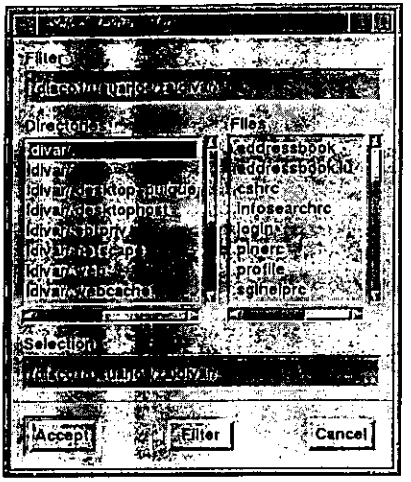
- *Ambient*: Este campo sirve para definir la luz ambiental que el objeto creará, generalmente sólo se aplica a un objeto especial en cada zona.
- *Diffuse*: En este campo se define el color de la luz que el objeto reflejará en todas direcciones; esta luz se reflejará independientemente de la posición del usuario.
- *Specular*: En este campo se define lo brillante que un objeto puede ser, es decir se define si la superficie reflejará toda la luz, como un espejo, o nada de luz, como los colores mate. Los valores cercanos a 1 en los tres colores reflejaran mucho; los colores tendientes a 0 serán opacos. El valor *Power* define la potencia y dispersión de la luz reflejada. Con un valor de 1 la luz se dispersa completamente y con un valor de 128, que es el máximo, la luz se concentra en un punto.
- *Emissive*: En este campo se define el color de la luz emitida por el objeto independientemente de si recibe alguna luz.
- *Opacity*: Se define la transparencia del material. Un valor de 0 hace transparente al objeto, un valor de 1 lo hace completamente opaco. El rango de valores es evidentemente de 0 a 1.

Editor de texturas

Con este editor es posible cambiar los parámetros que definen una textura. Los campos en esta ventana son los siguientes:



- *Texture Name*: Se coloca el nombre de la textura.
- *Texture Map*: Se define un archivo a ser mapeado en el objeto. Estos archivos pueden estar en los formatos **.tga**, **.int** e **.inta**, **.rgb** e **.rgba** o bien en **.vtx**.



- *Minify*: Se utiliza este campo para definir el algoritmo a ser utilizado en los casos en que el mapeo de pixeles de una textura no corresponda exactamente al del rendereo. En los casos en que dos o más pixeles deben ser mapeados en uno se utiliza alguna de las técnicas siguientes:
  - *None*
  - *Point_Sampled*
  - *Bilinear*
  - *Trilinear*
  - *Mip_Map_Linear*
  - *Mip_Map_Bilinear*
  - *Mip_Map_Trilinear*
  - *Mip_Map_Quadlinear*
- *Magnify*: En los casos en que un pixel debe ser mapeado en más de un pixel se utilizan las siguientes técnicas:
  - None
  - *Point_Sampled*

- *Bilinear*
- *Trilinear*
- *Bicubic*
- *Sharpen*

- *Alpha*: En este campo se escoge la forma en que el componente *alpha* o de transparencia de una textura es combinada con la imagen descrita en el archivo *geometry*. Las opciones a escoger son las siguientes:
  - *None*
  - *Blend*, que mezcla las dos texturas.
  - *Cut*, elimina la textura del archivo *geometry*.
  - *Blend_Cut*, deja la mezcla entre las opciones anteriores.
- *Wrap U*, y *Wrap V*: En los casos en que la superficie del objeto es más grande que la textura a asignar se define una técnica para ajustarla. Esta técnica se define separadamente para el eje **U** y **V**. Se tienen las siguientes técnicas para tal efecto:
  - *Repeat*, que sólo repite indefinidamente la imagen
  - *Clamp*, que sólo aplica la textura una vez y la amplia hasta cubrir la superficie.
  - *Select*, que sólo aplica la textura una vez y deja el resto de la superficie sin textura.
- *Detailed Type*: La textura definida en *Detailed Texture* será aplicada con alguna de las técnicas que en este campo se presentan:
  - *Add*, que suma las texturas original y de detalle.
  - *Modulate*, que mezcla ambas texturas.
  - *None*, que deshabilita la opción de textura de detalle.
- *Detailed Texture*: La textura que aquí se defina será aplicada al objeto cuando se le hace un acercamiento y la textura original no pueda ser magnificada a tal extremo.

3.2.5.1.2 Toolbox

En dVISE existen, como ya se mencionó, dos herramientas fundamentales para su programación, el Control Panel, ya descrito, y el Toolbox. Este último se diferencia del otro principalmente por poder ser utilizado dentro del ambiente virtual. El Toolbox es mucho más limitado que el Control Panel en cuanto a sus alcances; sin embargo, dentro de sus límites, es una herramienta mucho más amigable.

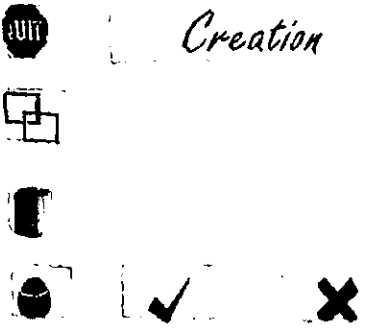
A continuación describiremos la forma en que está constituido, sus posibilidades y su forma de utilización.

Menú principal

El menú principal del Control Panel puede ser llamado desde cualquier parte del ambiente mediante el botón de Control Panel, si se tiene control por joystick o con las teclas Ctrl + T.

Este menú contempla las siguientes opciones:

Creación de objetos



Esta herramienta presenta las siguientes posibilidades rotuladas con el título de ***Creation***:

En la parte inferior del menú ***creation*** se encuentran dos botones con una paloma y un tache respectivamente, éstos sirven para confirmar o cancelar los cambios que se hayan hecho con la herramienta respectiva desde la última confirmación. Al salir de cada submenú, los cambios se confirman automáticamente.

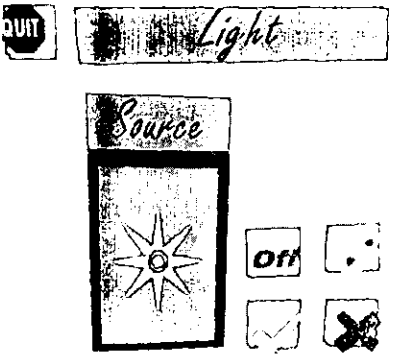
De abajo hacia arriba, los botones de la izquierda son para crear, borrar y copiar nuevos objetos. El primero de ellos, con el mismo icono mostrado en el menú anterior, permite crear instancias o copias de los objetos que se tienen definidos. Al seleccionar esta opción el menú cambia por un conjunto de iconos que simbolizan a los objetos que se tienen definidos, al seleccionar alguno de ellos, se creará una nueva copia del mismo que se colocará junto al usuario. Es posible definir una posición específica para que aparezcan todas las copias de un objeto dado.

El segundo icono, en forma de un bote de basura, sirve para eliminar objetos del ambiente. Para lograr esto, es necesario seleccionar el botón, seleccionar los objetos por eliminar y volver a seleccionar la herramienta borrar. Es hasta este momento que los elementos desaparecen del ambiente virtual, mientras este

último paso no se haya dado, es posible cancelar la acción mediante el botón cancelar explicado anteriormente.

Finalmente, el botón copiar sirve para duplicar un objeto dado. Para lograrlo, hay que seleccionar este botón con lo que se cambia a modo de copiado, en este modo, todo objeto que se seleccione será duplicado; debido a que el nuevo objeto tomará la misma posición que el original, es recomendable moverlo desde el mismo momento en que es creado a fin de no confundirlos. El modo de copiado finaliza cuando se selecciona el botón copiar nuevamente.

Luces



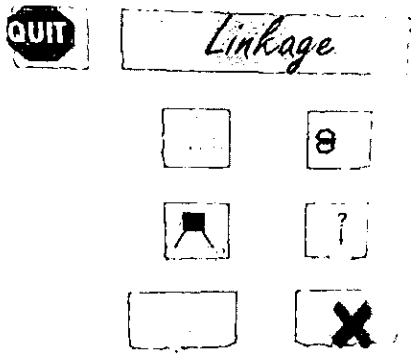
Este menú sirve, como su nombre lo indica, para modificar las luces que el ambiente contiene. No es posible crear luces desde esta herramienta.

Una vez seleccionada la herramienta, el menú titulado Source aparece, y en él se despliegan los diferentes tipos de luces que se encuentran en la zona actual.

Como se menciona en la propiedad respectiva en el control panel, los tipos de luces son: puntual, direccional y ambiental, las cuales son representadas en este menú por un foco, una lámpara de mano y un sol respectivamente.

Al seleccionar alguna de estas luces aparecen las opciones respectivas; en el caso de la luz puntual, sólo es posible prenderla o apagarla, en el caso de la luz ambiental, es posible cambiar el color mediante el editor de color. Finalmente, la luz direccional puede ser reorientada girando una lámpara de mano que se presenta en pantalla.

Ligado



El siguiente paso que se puede dar es crear relaciones entre los objetos. Para esto existe la herramienta ligado.

Al seleccionar esta herramienta aparece un menú con las siguientes opciones:

Crear una jerarquía de objetos, romper una jerarquía de objetos, ver todos los objetos menores a una jerarquía y finalmente ver todos los objetos superiores en una jerarquía.

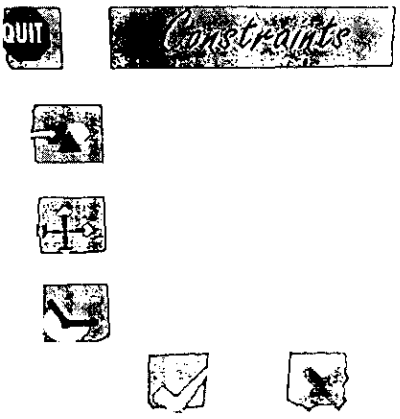
La primera de estas herramientas, el botón de ligado, sirve para unir dos objetos virtuales y formar una jerarquía. Primero se selecciona al padre de la relación, después se seleccionan los hijos, el caso de cambiar de opinión respecto al padre es necesario deseleccionar lo previamente escogido a fin de iniciar nuevamente el proceso. Para finalizar el proceso de ligado se presiona el botón de confirmar.

Para romper una relación es necesario elegir el botón de desligar y enseguida seleccionar a cada hijo que se desea desligar. Una vez seleccionados éstos es necesario confirmar la operación.

La tercera herramienta sirve para localizar los hijos de un determinado objeto. Cuando la herramienta se encuentra activa y se selecciona un objeto, éste toma un color rojo en caso de tener hijos y éstos se muestran en verde, en caso de que el objeto no tenga hijos, éste se mantendrá gris cuando sea seleccionado. Para deshabilitar esta opción sólo hay que seleccionarla de nuevo.

La última herramienta funciona de la misma forma que la anterior, únicamente con la diferencia de que en este caso se selecciona al hijo y el que aparece en rojo es el padre.

Restricciones

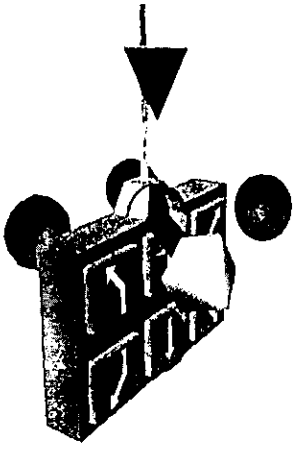


Mediante esta herramienta se limita la libertad de movimiento de los objetos.

Al seleccionar esta herramienta se presenta un menú con las siguientes herramientas:

El primero de ellos es el botón de Selección de Objeto, que evidentemente es usado para determinar el objeto a restringir. Una vez seleccionado el objeto es posible restringirlo en todo tipo de movimientos.

Restricciones de traslación.



Para evitar que el objeto se traslade, se selecciona la herramienta respectiva con lo cual se presentan los ejes coordenados en 3D.

Para restringir los movimientos de un objeto, es necesario arrastrar estos ejes hasta el objeto en cuestión y soltarlos ahí. A partir de este momento los ejes coordenados se alinean con el objeto y toman forma de acuerdo con las restricciones que el objeto ya tenía. Cuando un objeto no tiene restricción en determinado eje, en determinado sentido, el mencionado eje presenta una esfera en el extremo. Para establecer una restricción en ese eje, en esa dirección, es necesario seleccionar la esfera y moverla a la distancia del centro que se desee limitar el movimiento del objeto; al soltar la esfera ésta se convierte en un cono que apunta hacia afuera indicando así que existe una restricción en ese eje.

Si se desea eliminar cualquier movimiento en esa dirección se requiere soltar la esfera en el centro de los ejes coordendos, en ese momento, la esfera (o cono) se convierte en un cono que apunta hacia el centro. Finalmente, para eliminar cualquier restricción se toma el cono, se atraviesa el centro con él, se regresa al centro y se suelta ahí; de esta manera el eje vuelve a su forma original con la esfera en un extremo.

Existe también la herramienta Grid, esta herramienta permite establecer una rejilla tridimensional que forza los movimientos del objeto a tomar posiciones de acuerdo

con ésta. Para establecer o eliminar el Grid es necesario seleccionar primero, el botón de cambio de traslación y después el botón de Grid que apagará o prenderá alternativamente el Grid.

Cambio de Traslación

Grid

Restricciones de rotación

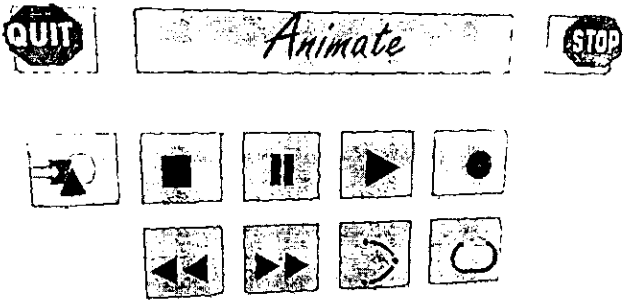
Para restringir la rotación de un objeto (una vez seleccionado el mismo), se selecciona el botón de rotación.

Botones de cada uno de los tres ejes aparecen. Cuando algún botón está en gris significa que no hay restricciones en ese eje, de otro modo el botón se encontrará en verde. Para colocar o eliminar una restricción sólo se requiere seleccionar el botón.

En el Control Panel se pueden establecer límites para estas rotaciones, es importante hacer notar que si se eliminan las restricciones en el Toolbox, los límites establecidos en el Control Panel también son eliminados.

La herramienta de Grid para rotaciones funciona de la misma forma que el Grid para traslaciones.

Nota: Tanto el Grid de rotación como el de traslación pueden ser establecidos en cuanto a su definición usando el Control Panel.

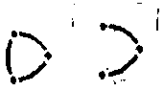


Las animaciones de objetos son definidas mediante una serie de posiciones que

llamaremos keyframes, las cuales deberá recorrer el objeto a velocidades dadas.

El menú Animate está conformado por una serie de controles tales como reproducción, pausa, parar, grabar, avanzar y retroceder. También se tiene la herramienta de selección de objetos vista anteriormente.

Se tienen dos herramientas auxiliares para la definición de keyframes.



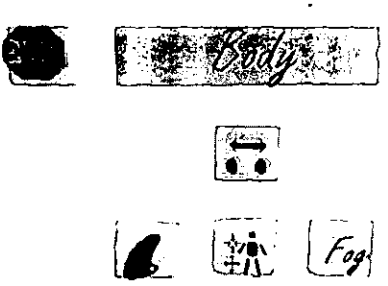
El primero de ellos, el botón de ajuste, forza la posición de un keyframe a la posición exacta de otro en caso de encontrarse cerca de él. Esto se realizará mientras el botón se encuentre activado.

Finalmente el botón repetir tiene la finalidad de definir si la secuencia se mantendrá en un ciclo infinito o sólo se ejecutará una vez.

Para grabar una animación se requiere, una vez seleccionado el objeto, escoger el botón de grabar, de esta forma aparece una copia transparente del objeto seleccionado, para marcar la siguiente posición se requiere mover esta copia

hasta la posición deseada y soltarlo ahí; también se puede ajustar la rotación del objeto en esta posición girando a la copia transparente. Juntas, la posición y la rotación del objeto conforman el llamado keyframe. Para crear el resto de los keyframes sólo es necesario arrastrar las copias transparentes que se crearán cada vez que se termine un keyframe. Una vez que se tengan todos los keyframes necesarios se requiere presionar nuevamente el botón grabar de tal forma la animación queda concluida y puede ser reproducida con el control correspondiente.

Body



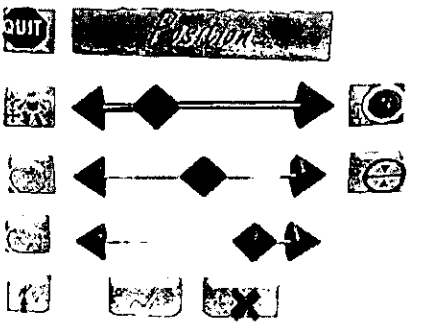
Mediante esta herramienta se pueden establecer las condiciones iniciales para el usuario al momento de ingresar al ambiente virtual, así como para el mismo ambiente virtual.

Al seleccionar esta herramienta en el menú principal se presenta el menú Body que incluye cuatro herramientas.

Position

Sirve para determinar la posición y velocidad en que el usuario se encuentra al ingresar a la zona en cuestión.

El menú *Position* presenta varias herramientas con la primera de ellas se puede cambiar el tamaño que el usuario tendrá al ingresar al ambiente, esto se hace mediante una barra de desplazamiento. Al determinar el tamaño del usuario se determina también su velocidad de desplazamiento, ya que mientras más pequeño sea más lentamente podrá desplazarse.

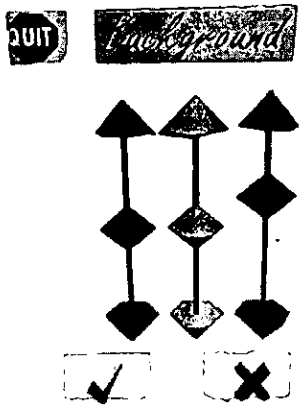


Para determinar la posición inicial que tendrá el usuario al ingresar a una zona se tiene el botón *body*. Antes de usar este botón se requiere colocarse en la posición deseada. Recuérdese que el *toolbox* puede ser visto en cualquier posición que se encuentre presionando Control + T. Para determinar la velocidad en que se mueve el usuario en el ambiente, es posible usar el botón Slow y mover la barra de desplazamiento hacia la derecha, para acelerar o a la izquierda para hacer lo contrario. También se tiene una herramienta para determinar la velocidad en el estado de alta velocidad. Este estado se tiene mientras se presiona el botón respectivo.

La herramienta de nivel de vuelo evita que el usuario se mueva en el eje Z.
Con la herramienta compass el usuario obtiene una referencia de los ejes coordenados en la esquina inferior derecha de su visión a fin de poderse ubicar en caso de perder la perspectiva en el ambiente.

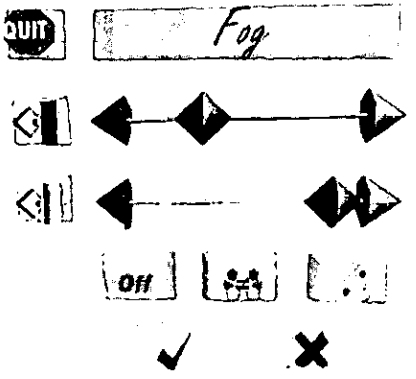
Background

Sirve para determinar el color del fondo del ambiente y utiliza el editor de color.



Fog

Se pueden introducir efectos atmosféricos para aumentar el realismo del ambiente virtual. Se puede determinar la distancia a la cual la atmósfera empieza a ser nebulosa y la distancia a la cual la niebla es tan densa que no se puede ver a través de ella.



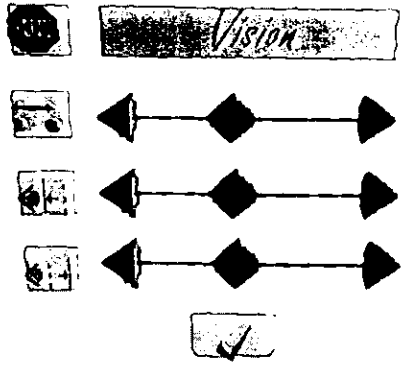
En el menú de esta herramienta se presentan diversas opciones, con las primeras dos de ellas ubicadas a la izquierda del menú se determinan las distancias mencionadas anteriormente, esto mediante las barras de desplazamiento que se comportan de forma logarítmica.

El botón On/Off sirve para habilitar o deshabilitar la opción de niebla.

El botón de paleta sirve para elegir el color que la niebla tendrá, en caso de desearlo así, se puede elegir directamente el color del fondo del ambiente utilizando la herramienta Color de Niebla

Vision

Finalmente, esta herramienta es utilizada para ajustar los parámetros visuales. Estos parámetros son: la distancia interocular y el rango visual.



La primera herramienta, de arriba hacia abajo sirve para ajustar la distancia entre los ojos del usuario. Esto significa que entre mayor sea la distancia ajustada con la barra de desplazamiento mayor será la diferencia entre lo que ve un ojo y el otro debido al ángulo de incidencia que cada uno tendrá en el ambiente.

La otra herramienta permite establecer las distancias más cercana y más lejana que el usuario podrá apreciar. Con la primera barra de desplazamiento se ajusta la

distancia a la que el usuario comienza a tener visibilidad; con la segunda se ajusta la distancia a la que el usuario vuelve a perder la visibilidad.

Save

El botón Save sirve para guardar los cambios hechos con el resto de las herramientas del Toolbox. Mientras esta herramienta no sea seleccionada ninguno de los cambios hechos en el ToolBox será guardado en el archivo vdi y por lo tanto no tendrá efecto la próxima ocasión en que se cargue el ambiente. Esta herramienta es necesaria independientemente de la confirmación de los cambios que se requiere en cada una de las herramientas y de la opción save en el control panel.

Con esto concluímos lo que se refiere a las herramientas tanto de software como de hardware que podrían ser utilizadas para realizar un sistema en realidad virtual. Este capítulo no pretende decidir cuáles son las herramientas más adecuadas para realizar un sistema, más bien se intenta describir las características de éstas, a fin de que el desarrollador decida cuáles se ajustan al sistema deseado.

4 Aplicación de la metodología

El sistema que desarrollaremos tiene como objetivo presentar una aplicación práctica de la metodología propuesta, de tal forma que la teoría se vea sustentada en un sistema que presenta a la cultura Maya mediante elementos virtuales.

Es importante exponer las distintas posibilidades que tiene este tipo de sistemas, tanto en interactividad con el usuario, como en lograr su inmersión en el ambiente. Finalmente es destacable la importancia de mostrar la capacidad de los sistemas en realidad virtual para exponer temas culturales como la antropología, la historia, etc.

4.1 Análisis

En esta etapa realizamos el aprendizaje de herramientas, determinamos el contenido del sistema y realizamos una investigación general del tema a abordar; todo esto siguiendo la metodología propuesta en esta tesis de tal forma que se pueda constatar la eficiencia de la mencionada metodología.

4.1.1 Análisis, elección y aprendizaje de las herramientas.

4.1.1.1 Análisis de las herramientas de hardware

Existen varias herramientas de hardware para la realización de sistemas en realidad virtual. Las herramientas de entrada se dividen en dispositivos de localización y dispositivos de control. Los primeros se refieren a los dispositivos mediante los cuales se determina la posición y movimientos del usuario dentro del ambiente virtual, existen de tipo electromagnéticos, electromecánicos, ultrasónicos y ópticos. Los segundos se refieren a aquellos dispositivos que permiten al usuario tener interactividad con el ambiente, es decir, poder manipular objetos, etc.

Las herramientas de salida se dividen en dispositivos de presentación, de audio, realimentación táctil y móviles. Los dispositivos de presentación se refieren a la forma en que el usuario captará visualmente al ambiente virtual, los de audio se refieren, evidentemente, a la forma en que el usuario captará sonidos, los de realimentación táctil se refieren a la posibilidad del usuario de utilizar el sentido de tacto; finalmente, los dispositivos móviles son espacios físicos que rodean al usuario y presentan el despliegue visual de tal forma que el usuario se ve inmerso en el ambiente.

4.1.1.2 Elección de las herramientas de hardware

Para este sistema en particular no se tenían los recursos como para elegir entre varias herramientas por lo que nos limitamos a utilizar las herramientas que provee el Laboratorio de Interfaces Inteligentes.

4.1.1.3 Análisis de las herramientas de software

En lo que se refiere al software se analizaron las características de varios programas aunque el motivo fundamental por el que se eligieron estas herramientas fue la disponibilidad de las mismas. Aún así fueron analizadas otras herramientas: por ejemplo, se analizó la posibilidad de utilizar AutoCAD, Soft Image, New Tech Light Wave o Alias para la construcción del ambiente tridimensional. Se pensó también en utilizar Corel Photo Paint, para la edición de imágenes o inclusive el Sense8[25] para la programación del ambiente Virtual. Respecto a las herramientas de hardware, no se hizo un análisis, esto debido a que los recursos con los que se cuenta dentro del laboratorio son reducidos y teníamos que limitarnos a ellos

[25] Sense8 Web page. http://www.sense8.com

4.1.1.4 Elección de las herramientas de software

Las herramientas utilizadas para la realización de este proyecto fueron principalmente las siguientes:

- 3DMAX® v. 3.0. Para la realización de los modelos tridimensionales a utilizar en el ambiente virtual. También se utilizó este programa para asignar texturas e imágenes a los modelos tridimensionales.

Algunas de las razones fundamentales por las que se eligió este programa fueron la conjugación de una potencia importante con una facilidad de uso que se acentuó con un conocimiento previo que se tenía en el paquete 3D Studio®, predecesor directo del 3D MAX®. Fue importante también la compatibilidad que presenta el programa dVISE ® con el formato 3DS, el cual también es manejado eficazmente por el 3D MAX®.

- ADOBE PHOTOSHOP® v. 5.0. Para la edición de imágenes que se utilizarán para ser asignadas, ya sea como texturas o directamente como imágenes dentro del ambiente virtual.

Las razones para la elección de esta herramienta son muchas, entre ellas es de destacarse la compatibilidad que presenta este programa con otros utilizados para el sistema debido al soporte de la empresa que lo produce; Adobe® es, sin duda, un respaldo importante para cualquier programa. También es importante la enorme superioridad que esta herramienta ha mostrado con respecto a sus competidores en el mercado. De la misma forma es de considerarse el conocimiento previo que teníamos los tesistas en el manejo de la misma.

- ADOBE PREMIERE® v. 5.1. En la edición de videos que podrían ser utilizados dentro del sistema.

A nivel universitario este programa tiene la suficiente potencia para la calidad de videos necesaria. Además de esto cuenta, como en el caso de Photoshop®, con el respaldo de Adobe® y su enorme compatibilidad con el formato de Photoshop® permite excelente calidad en la incrustación de imágenes. Para utilizar algún programa de mayor calidad sería necesario equipo demasiado caro.

- SOUND FORGE® v. 4.5. Para la edición de audio, ya sea música, sonidos o cualquier tipo de efecto auditivo que se pueda requerir en el sistema.

Una vez más la principal razón para elegir este programa fue la sencillez con que se maneja además de la potencia que tiene. Es importante su capacidad de trabajo en máquinas no tan potentes de tal forma que evita gastos excesivos en hardware.

Estos paquetes fueron aprendidos mediante el método de prototipos. Esta técnica se basa en la realización de pequeños proyectos sobre los cuales se aprendieron las diversas posibilidades que los programas presentaban. Una vez habiendo dominado las características básicas del programa en cuestión se procedía a realizar proyectos un poco más complejos sobre los cuales nuevamente se aprendian herramientas más completas de las que el programa dispone. Con este proceso se continúa hasta conocer el programa lo suficiente como para realizar algunos proyectos ya encaminados a las necesidades del sistema en general.

Evidentemente, los programas utilizados son bastante complejos como para pretender dominarlos por completo. Es por ello que cobra importancia el saber diferenciar cuando se tiene el suficiente dominio del programa para alcanzar los objetivos del sistema y continuar el aprendizaje resultaría innecesario.

En lo que se refiere al aprendizaje de las herramientas tanto de hardware como de Software, no se considera necesaria su documentación.

4.1.2 Contenido del sistema

En esta etapa se eligió el tema de "La Cultura Maya". Inicialmente se decidió abarcar la antropología debido al interés que teníamos en utilizar la tecnología de la computación en un aspecto independiente a la misma. Además de ello, consideramos que la antropología es una ciencia que, al estudiar las culturas del mundo en diferente espacio y tiempo, nos provee de herramientas para afrontar el presente y futuro de nuestra propia cultura. En específico determinamos abordar las culturas prehispánicas mesoamericanas debido principalmente a que son éstas las culturas de las que surge la sociedad en que vivimos, su estudio cobra vital importancia en los tiempos actuales debido a la evidente necesidad de reforzar la identidad nacional. Este tema; sin embargo, resultaba aún demasiado amplio para ser abarcado en un sistema de esta naturaleza, es por ello que se tomó la decisión de enfocar la investigación a la cultura Maya, decisión tomada con base en el interés de los tesistas en el tema y en la basta información con la que se cuenta. La cultura Maya, además de interesante, es, junto con la cultura Azteca una de las culturas que más ha sido estudiada dentro de Mesoamérica. Esto provoca que la información disponible sea demasiado extensa para ser contemplada en el sistema, lo que nos obliga a hacer una nueva selección de temas a abordar. Esto último se hará en la etapa correspondiente.

4.1.2.1 Investigación general del tema

La información para el tema fue obtenida en libros, revistas, conferencias, visitas a zonas arqueológicas, exposiciones, Internet y entrevistas con especialistas en la materia (antropólogos, historiadores, arqueólogos, etc.).

Se leyeron varios libros sobre la cultura Maya para tener una visión general sobre la misma y así poder definir, posteriormente los temas más representativos. Se leyó también el libro del Popol Vuh para tener una visión de cómo veían los Mayas el mundo y apoyarnos en esto también para la selección de temas. Así mismo, se consultaron varias revistas de Mundo Maya a fin de obtener temas específicos

sobre los que se podría tratar. Finalmente se asistió a conferencias y seminarios impartidos por el INAH en los que se obtuvieron ideas para algunos temas a abordar.

En estos seminarios, se logró también la cooperación del proyecto "La pintura Mural Prehispánica en México" dirigido por la Dra. Beatriz de la Fuente del Instituto de Investigaciones Estéticas de la Universidad Nacional Autónoma de México. De esta colaboración se obtuvieron las fotografías de los murales de Bonampak digitalizadas.

4.2 Diseño

En esta etapa se definió la forma en que el sistema trabajará en cuanto a los mecanismos que utilizará para mostrar la información. Se definió también la arquitectura del ambiente virtual a utilizar y finalmente se definieron los temas específicos de la cultura Maya que se abordarían.

4.2.1 Aspecto técnico

4.2.1.1 Definición de la estructura general del sistema

Se consideraron varios elementos para mostrar la información. Un punto importante fue la determinación de utilizar textos para explicar las imágenes debido a lo inviable que resultaría el uso de audio para esta labor. Esto debido a la gran cantidad de espacio en disco y en memoria RAM que se requeriría utilizar.

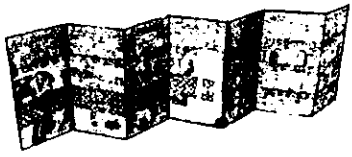
Otro aspecto importante fue que se determinó no utilizar video debido a la complejidad de mostrar imágenes sucesivas y a que el programa dVISE no permite asignar video como textura para un objeto.

Finalmente, debido a la poca capacidad de memoria con que se cuenta y a la gran cantidad de memoria que ocupa cada imagen, se determinó limitar la información.

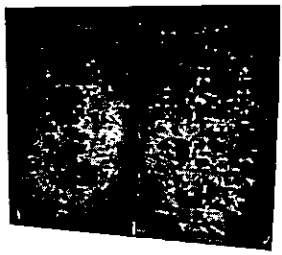
Ciudades Mayas

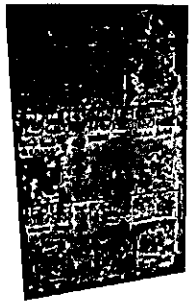
Se tendrá una pantalla para presentar imágenes; mediante ellas se presentará un panorama general de la cultura, éstas serán acompañadas con textos que explicarán el tema respectivo. La pantalla será controlada mediante botones que permitirán al usuario intercambiar las imágenes.

Se mostrarán también, a manera de museo, algunas piezas de la cultura Maya; ejemplos de ello son los códices Madrid, Dresden y Paris. El códice Madrid fue modelado mediante placas rectangulares unidas en sus cantos como se muestra a continuación.

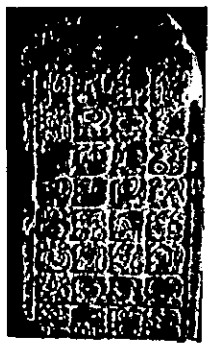


Segmentos de los códices Dresden y Paris son mostrados en placas verticales.





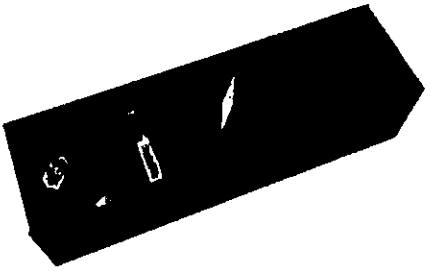
Placas similares son utilizadas para exponer la Estela de Yashilán y un ejemplo del arte en papel amate realizado por el pueblo Maya actual.

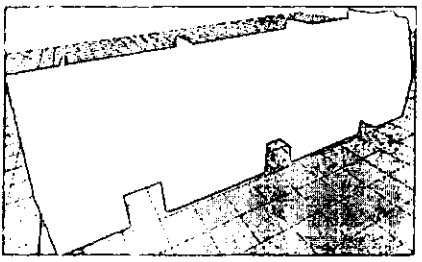


4.2.1.2 Modelo general del ambiente virtual

Se utilizarán tres secciones dentro del ambiente.

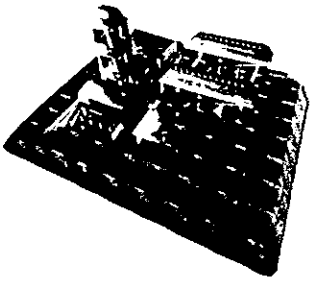
La primera de ellas es un cuarto rectangular, en su interior se encuentran las piezas mencionadas en la etapa anterior así como la pantalla y las estelas.

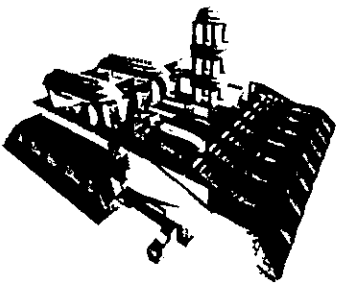




Por otro lado, en la segunda sección se presentan los famosos murales de Bonampak, éstos fueron aproximados en tercera dimensión e ilustrados con fotografías gracias a la colaboración del Seminario de Pintura Mural Prehispánica y del Lic. Ricardo Alvarado Tapia. Estos están contenidos en tres cuartos con base rectangular y techo en forma piramidal. Su interior consiste, únicamente, en una banqueta alrededor del cuarto.

También contaremos con una sección donde se podrá visitar "El Palacio", conjunto de edificios de la zona arqueológica de Palenque.





4.2.2 Aspecto del contenido del sistema

4.2.2.1 Definir alcances de la información

Una vez investigado todo lo referente al tema a exponer y con el conocimiento necesario de las herramientas que se tienen, se obtuvieron conclusiones importantes sobre la información que puede ser incluida y la forma en que se presentará.

4.3 Desarrollo

4.3.1 Aspecto del contenido del sistema

4.3.1.1 Investigación profunda del tema

La información obtenida en "Investigación General del Tema" de la etapa de análisis fue ampliada a fin de lograr un documento lo suficientemente completo para mostrar una panorama de la cultura Maya. No consideramos pertinente incluir este documento en la presente tesis debido a que no pertenece al ámbito ingenieril. Aún así el documento está disponible y puede solicitarse a la dirección osornohm@hotmail.com.

Esta información fue resumida en las imágenes y textos presentadas en el sistema. Las imágenes de Bonampak fueron presentadas en los cuartos respectivos dentro del mismo sistema.

4.3.1.2 Desarrollo de elementos multimedia

Se digitalizaron y editaron múltiples imágenes obtenidas de la etapa anterior para explicar los temas. También se editaron imágenes a fin de asignar texturas a los diversos objetos que conformarán el ambiente.

4.3.2 Fases de integración

En las etapas aquí contempladas se unen los aspectos del contenido del sistema y técnico que hasta el momento se habían manejado por separado. En este momento es evidente la necesidad de que las etapas anteriores, de ambos aspectos, hayan concluido para poder ser integradas.

4.3.2.1 Incorporación de los elementos multimedia

En este sistema el elemento multimedia fundamental es la imagen, éste tuvo tres propósitos fundamentales. Por un lado existen imágenes utilizadas para dar una textura a los objetos del ambiente. Estas texturas permiten que la base de una pieza del museo parezca hecha de piedra, que las paredes de los cuartos parezcan de madera, o que el piso parezca ser una alfombra negra. La textura asignada a la zona arqueológica de Palenque, por ejemplo, sirve únicamente para aparentar que está hecho de piedra.

Por otro lado, se encuentran las imágenes fotográficas, con textos o sin ellos, que se asignan a determinados objetos a fin de ser apreciadas por el usuario. En este caso se encuentran las imágenes asignadas en los cuartos de Bonampak – famosos precisamente por estos murales -. Otras imágenes que se encuentran en esta categoría son las asignadas a la pantalla de navegación que explican los diferentes temas de la cultura Maya. Dentro de este tipo de imágenes encontramos a las asignadas a las piezas del museo, evidentemente no tendría sentido colocar piezas sin imágenes asignadas.

Algunas imágenes sirven para que simples polígonos simulen formas conocidas como botones de control, switches o lámparas.

Las imágenes fueron incorporadas al ambiente virtual mediante la propiedad visual de los objetos respectivos. Esta propiedad permite asignar una imagen a los objetos. Esta imagen es suficiente en el caso de las texturas o de objetos como

botones o switches; sin embargo, en el caso de las imágenes más significativas es necesario asignar una imagen de detalle con una mayor resolución que la imagen que se ve a simple vista para permitir que el usuario la aprecie bien a cualquier distancia, esto también se hace en la propiedad visual de los objetos.

4.3.3 Aspecto técnico

4.3.3.1 Programación de interactividad en el ambiente virtual

La programación en el sistema dVISE se basa en el establecimiento de las propiedades de cada uno de los objetos que conforman al ambiente virtual, de la definición de las relaciones entre ellos y del establecimiento de las diferentes zonas del ambiente virtual, así como de las propiedades de las mismas.

Las propiedades de los objetos así como de las zonas ya fueron descritas en capítulos anteriores. Para esta aplicación en particular utilizamos primordialmente las propiedades de *Behavior*, *Constrain*, *Light*, *Visual*, *Audio* y *Collision* así como la definición de las zonas que conforman el ambiente.

La propiedad *Visual* tuvo básicamente una aplicación. Ésta fue la de cambiar el material de la pantalla respondiendo a los eventos disparados por los controles de la misma, es de esta manera como se muestran los diferentes temas que explican parte de lo que es la cultura Maya.

La propiedad *Light* únicamente fue utilizada para lograr la iluminación del ambiente, se creó un objeto al cual llamamos "luz", y se le asignó, dentro de la propiedad *Light*, una luz ambiental blanca. También se le asignó esta propiedad al objeto lámpara pero, en este caso, una luz puntual (spot light).

La propiedad *Collision* fue utilizada en prácticamente todos los objetos que conforman el ambiente. Esto es debido a que casi siempre es necesario reconocer cuando el usuario toca a un objeto; ya sea para que no lo penetre (como en el

caso de las paredes), o para realizar funciones (como en el caso de los controles de la pantalla o los interruptores de luz).

La propiedad *Constraint* también fue muy utilizada debido a que por la naturaleza misma del ambiente, los objetos son todos inmóviles.

En lo que se refiere a los audios, fueron implementados para casos como: apretar botones, accionar interruptores, alarmas o sonidos ambientales, la mayoría de estos casos se programaron mediante la propiedad *Behavior*.

La propiedad más importante para la programación del ambiente, es la propiedad *Behavior*, ya que en ella se definen los comportamientos que tendrán algunos objetos de acuerdo con los eventos que sufran (sean tocados, seleccionados, arrastrados, etc.).

Un comportamiento bastante sencillo es el que se refiere a los interruptores de luz. Este sólo establece que al darse el evento de que el objeto sea seleccionado (pick), se dispare la acción dvObjectLight, dirigida a la luz puntual "lámpara" y con el parámetro *toggle* a fin de que la luz se prenda y apague alternativamente.

Otro comportamiento utilizado es el que impide que el usuario salga de una zona definida. En este caso fue utilizada para que el usuario no pudiese salir del museo, englobado en la primera zona. Este comportamiento se logra mediante la acción *dvBoundaryCheck*. Para establecer los limites, es conveniente definir un plano en un archivo .bgf y determinar las alturas mínima y máxima en los parámetros *floor* y *ceiling* de la acción *dvBoundaryCheck*, al hacer esto, los valores en Y del archivo son reemplazados.

Esta acción es ejecutada al darse el evento ZoneCreiate que, como su nombre lo indica, sucede al entrar a la zona lo cual, en este caso, sucede al iniciar el programa.

El comportamiento que logra que el usuario sea llevado a otra zona mediante la puerta es similar al anterior, con la diferencia de que en vez de responder enviando al usuario a una posición determinada dentro del ambiente, responde cambiándolo a una zona distinta, esto se logra mediante la acción *dvzone*.

NOTA: Los archivos bgf (binary geometry file) son incomprensibles a simple vista debido a que se encuentran en modo binario, sin embargo, el dVISE provee de una herramienta que permite convertirlos a archivos ASCII, perfectamente legibles. Esta herramienta es el comando bgfchange que tiene diferentes posibles parámetros:

- ✓ **-fb2, sirve para convertir los archivos de ASCII a binario**
- ✓ **-fv2, sirve para convertir los archivos binarios a ASCII**
- ✓ **-fb1 y –fv1 tienen las mismas funciones para versiones anteriores de dVISE**

Los archivos vgf tienen la siguiente estructura:

Encabezado, definición de límites y geometría del objeto. El encabezado define únicamente el número con el cual se podrá encontrar el archivo además de definir datos como la versión del formato del archivo, la fecha de creación y la unidad de medida utilizada.

En la sección de Boundary o "límites" se define el volumen tridimensional del objeto con lo cual quedan definidos los mínimos y máximos que se alcanzan en cada uno de los ejes coordenados.

Finalmente, en la sección Object se definen los objetos utilizando alguna de las siguientes representaciones:

- ✓ **Tristrip**
- ✓ **Polistrip**
- ✓ **Pmesh**
- ✓ **Point list**
- ✓ **Sphere list**
- ✓ **Line**
- ✓ **Text**
- ✓ **Polygon**

Los primeros dos definen al objeto mediante colecciones de triángulos; Pmesh define un conjunto de vértices y un conjunto de conexiones entre ellos. Point list

sólo describe un conjunto de puntos con valores de color. Sphere list define una lista de esferas y Line define una línea mediante vértices en serie. Text contiene una lista de caracteres ASCII y finalmente Polygon describe una lista de n vértices que definen un polígono plano de n caras.

El comportamiento más complejo es el que se utiliza para darle funcionalidad a los controles de la pantalla. El código de esta programación es explicado en el Anexo de la presente tesis; la forma en que esto se programa, mediante el control panel, se explica a continuación.

Para lograr el cambio de la textura de la pantalla, fue necesario el uso de una variable que se incrementará o decrementará de acuerdo con los "click´s" que se hagan en los botones de avanzar y retroceder; de acuerdo con el valor de esta variable será disparada la acción *dvobjectVisual*, con la textura correspondiente como parámetro. El uso de la variable se logra mediante las acciones *dvAssign*[26], que asigna un valor a la variable y *dvCall* que evacua una expresión y lanza un evento de acuerdo con el resultado de esta evaluación.

Evidentemente, lo que se requiere es que, de acuerdo con la expresión evaluada se dispare una acción y no un evento; sin embargo, la única posibilidad de evaluar una variable es *dvCall*, por lo que se requirió definir nuevos eventos para cada una de las texturas posibles y, asociar a cada uno de ellos la acción *dvObjectVisual* que efectivamente cambie la apariencia de la pantalla. El programa dVISE no permite que un objeto lance eventos sobre si mismo por lo que fue necesario asignar estos eventos a un objeto externo[27],de tal forma que son llamados desde el objeto A pero están asignados en el objeto B.

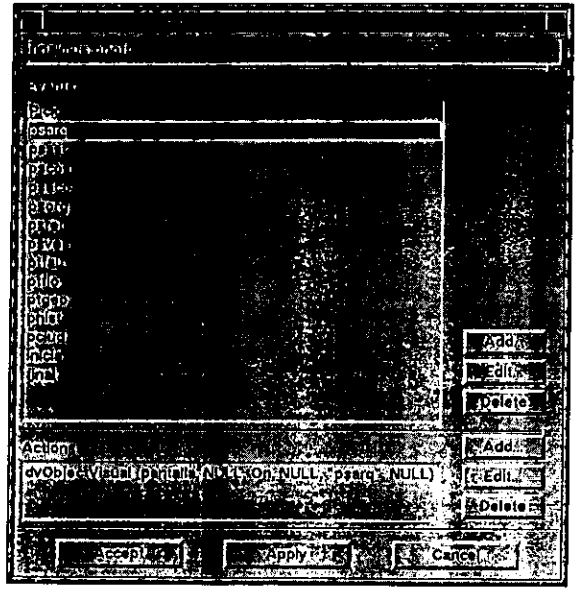
Por otro lado, la organización de los eventos y acciones de los controles de la pantalla quedaron definidos de la siguiente manera:

[26] Dvs Users Guide. http:// www.division.com.
[27] Esto no afecta en nada al objeto en cuestión debido a que estos eventos sólo podrán ejecutarse por llamados explícitos (en programación este tipo de eventos se llaman subrutinas o funciones)

- ✓ Se definieron materiales para cada una de los temas que se mostrarían en la pantalla con sus respectivas texturas.

- ✓ En la zona se definió un comportamiento en el que se le indica que al darse el evento *dvBodyCreate* se dispare el evento *inicia* que se define en el objeto *On/Off*.

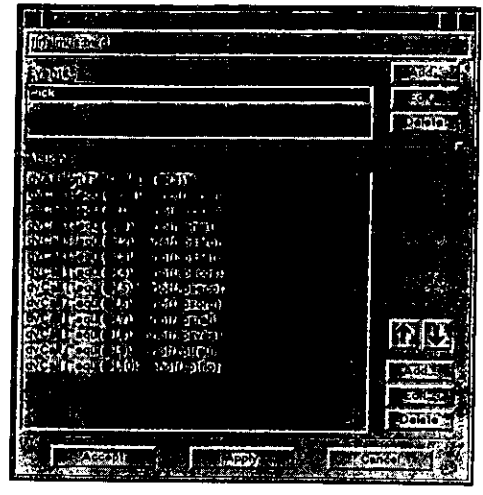
- ✓ En el objeto *On/Off* se definieron los eventos respectivos para cada una de los temas a mostrar.



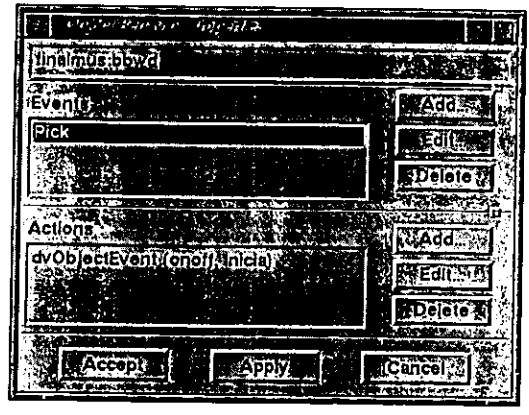
- ✓ Se definieron también eventos para lograr el manejo cíclico de la pantalla al llegar al inicio o al final de la secuencia; por ejemplo, cuando la variable se encuentra al final de la secuencia y se intenta avanzar, se llama al evento *inicia*, que a su vez coloca la variable en 0 y asigna la textura correspondiente. En el evento *Pick*, se definió una acción *dvObjectVisual* para aparecer o desaparecer la pantalla alternativamente.

- ✓ En el evento pick de los objetos *Fwd* y *Bwd*, la variable es incrementada o decrementada mediante la acción *dvAssign* usando las expresiones *Add*(),

o *Sub*() respectivamente; enseguida se colocan acciones para cada uno de los posibles valores de la variable que corresponden a los temas a mostrar. Estas acciones son del tipo *dvCall* y tienen como expresión a evaluar, mediante la función *equ*(), la comparación de "i" con el valor respectivo; el evento a llamar es el correspondiente a este valor.



✓ En el evento *pick* de los botones *Ffwd* y *Bbwd* se colocan acciones *Send event* que llaman a los eventos final e inicia respectivamente, ambos definidos en el objeto *on/off*.



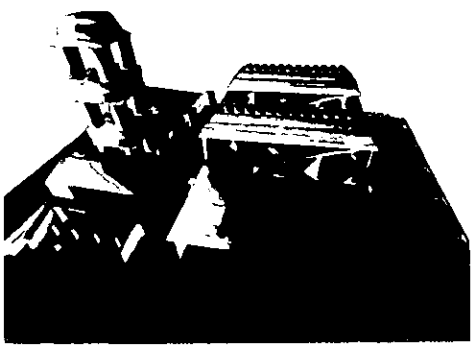
4.3.3.2 Versiones del proyecto

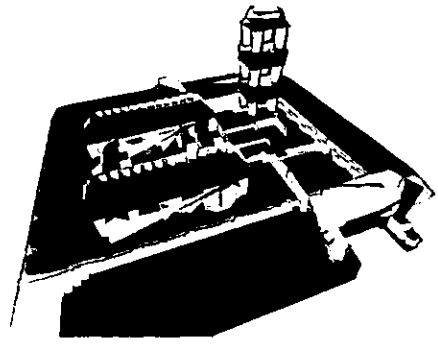
La primera versión que tuvimos del proyecto fue un intento por lograr el ideal que consistía en reunir a las 3 secciones como tres zonas de un mismo ambiente virtual. Para esta versión se utilizaron imágenes a resoluciones de aceptable calidad de tal forma que sus tamaños no excedieran los 200KB en formato vtx ni los 500KB en formato tga. Este intento fue del todo fallido debido a que al tratar de manejar todas las geometrías y sobretodo todas las texturas, excedimos por mucho la capacidad de la computadora, sobretodo en cuanto a su memoria RAM, en un caso óptimo, la memoria RAM debería ser de por lo menos 256 MB. Lo que sucedió fue que la gran cantidad de accesos a disco que tenía que hacer para manejar memoria virtual le hizo el trabajo casi imposible al grado de que difícilmente alcanzó a cargar todas las texturas y por supuesto nunca logró desplegar el ambiente en pantalla.

La segunda versión del ambiente fue un intento por reducir al máximo el tamaño de las imágenes utilizadas a fin de aligerar la carga de información en memoria y permitir así que el sistema trabajara con fluidez. Esta versión nuevamente resultó fallida debido a que con el tamaño de imágenes tan reducido, la calidad visual del ambiente era pésima.

Finalmente, en la tercera y última versión se logró una fluidez adecuada con una calidad aceptable en las imágenes, esto sólo fue posible, separando las zonas del ambiente virtual en ambientes independientes. Esto provocó que el comportamiento que se tenía contemplado para las puertas del museo no pudiese ser utilizado.

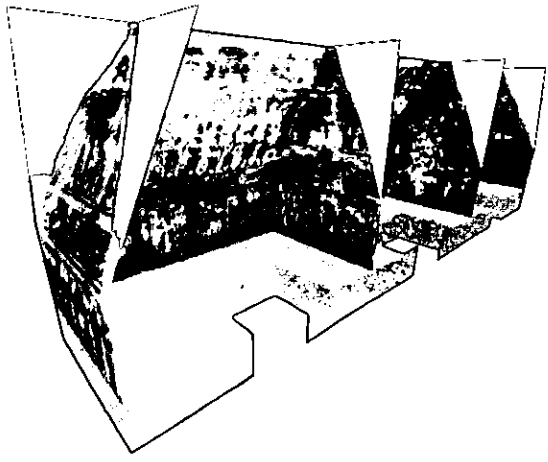
4.3.3.3 Versión final del proyecto e implementación



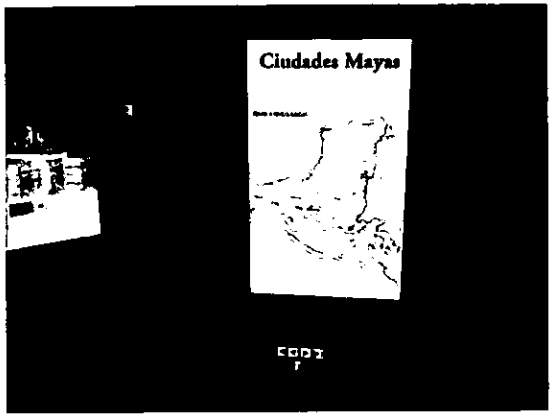


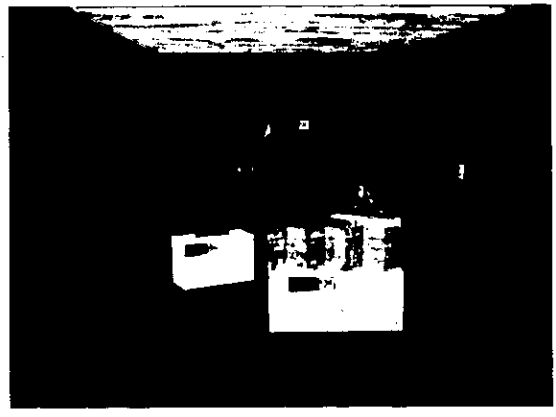
El proyecto quedó finalmente integrado por tres ambientes virtuales. Resulta interesante recorrer el primero de ellos, "el Palacio" de la zona arqueológica de Palenque. Este ambiente nos muestra las posibilidades que se tienen en cuanto a la complejidad que puede tener una estructura.

El segundo, los cuartos de Bonampak, muestra los famosos murales del sitio arqueológico. Este ambiente permite tener una visión muy cercana a la que se tendría si se visitaran estos mismos cuartos en el sitio arqueológico. Con este ambiente se logra mostrar las posibilidades que se tienen para asignar texturas a una estructura



De la misma forma, se tiene un museo que muestra diversas piezas de la cultura Maya. En este ambiente se muestran las posibilidades de interactividad que tiene el programa.





4.4 Mantenimiento

Una vez concluido el sistema, es de gran importancia el mantenimiento, tanto correctivo como adaptativo y perfectivo. En cuanto al mantenimiento correctivo, cualquier sistema es susceptible de ser mejorado, éste no es la excepción.

En lo referente al mantenimiento adaptativo, se pueden mencionar dos posibilidades. Por un lado, consideramos que, en caso de lograrse una ampliación en la capacidad de memoria RAM del equipo, los ambientes creados podrían integrarse en uno solo, a fin de lograr la versión que se manejó inicialmente, en la cual se incluyen las capacidades de tránsito entre zonas.

En lo que a la actualización se refiere, sería adecuado actualizar la computadora a las últimas versiones de la misma. Esto es, una configuración de un procesador MIPS R12000A a 400 Mhz con 256 Mb de memoria RAM a cambio del R5000 a 180 Mhz con 64 Mb de RAM que actualmente posee. En caso de no ser posible lo anterior, podría ser suficiente un procesador R12000 a 300 Mhz con 128 Mb de RAM. La información sobre las configuraciones más actualizadas y los precios correspondientes se puede encontrar en www.sgi.com/internacional/mx.

Por otro lado, la transición natural del sistema es hacia el Sense8 esto debido a la capacidad de éste para trabajar también en PC´s.

Finalmente, en cuanto al mantenimiento perfectivo, éste dependerá de la creatividad de los involucrados en el proyecto a fin de decidir qué capacidades o características podrían ser agregadas al ambiente, todo esto sujeto a las herramientas de software y hardware con la que se cuente en ese momento.

5 Conclusiones

Con base en la experiencia de desarrollar la metodología aquí presentada, así como su aplicación, podemos afirmar que la tecnología de realidad virtual tiene un enorme campo de desarrollo en el área educativa; esto debido a que, por sus características, las aplicaciones generadas resultan fáciles de usar para gente sin conocimientos de cómputo. La inmersión que permiten los dispositivos de realidad virtual (Casco y Joystik en este caso) logra que el usuario pueda recorrer con facilidad los ambientes de una manera intuitiva.

En la presente tesis se optó por desarrollar una nueva metodología basada en los modelos existentes, pero que toma en cuenta las diferencias que se tienen con respecto a los sistemas tradicionales. La metodología planteada es aplicada en un ambiente que muestra aspectos de la cultura Maya; sin embargo, es posible aplicarla para todo tipo de ambientes que muestren, desde algo tan complejo como una proyecto de ciudad en Marte hasta algo tan sencillo como un rompecabezas tridimensional.

La realidad virtual va mucho más allá que el programa dVISE en específico, por esto, la metodología aquí presentada no se limita al mencionado programa, más bien, es flexible para ser utilizada en el desarrollo de esta tecnología en cualquier programa que se utilice.

Podemos concluir también que el desarrollo de la realidad virtual irá a la par de las exigencias del mercado, con esto nos referimos a que las investigaciones en este campo del conocimiento, serán estimuladas de acuerdo con las aplicaciones que de ella se obtengan. Para ello es importante que áreas del conocimiento tan aparentemente alejadas de la computación, como es el caso de la antropología, deberán ser contempladas para el desarrollo de los nuevos sistemas a fin de promover este tipo de tecnología.

Bibliografía

SALAZAR PERALTA, ANA MARÍA. Antropología Visual. Universidad Nacional Autónoma de México, Instituto de Investigaciones Antropológicas. 1ª Edición. México, 1997.

CASEY LARIJANI, L. Realidad Virtual,
McGraw Hill.

SOMMERVILLE, IAN. Ingeniería de Software,
2ª Edición Addison Wesley, México, 1988

CANCHÉ RODRIGUEZ, CLAUDIA. Creación de un sistema en RV,
Tesis de Ingeniería en Computación, México 1999, F.I. UNAM.

FAIRLEY, RICHARD E. Ingeniería de Software
Mc Graw Hill, México, 1988

PRESSMAN, ROGER. Software Engineering, A Beginner's Guide
Mc Graw Hill, E.U.A., 1988

STAMPE, DAVE. Realidad Virtual. Creaciones y Desarrollo.
Ed. Anaya Multimedia. S.A, Madrid, 1995

PHILIPPE QUÉAU. Lo Virtual, Virtudes y Vértigos
Ed. Paidos. Madrid, 1995

GHEZZI, CARLO, ET AL. Fundamentals of Software Engineering
Ed. Prentice Hall, E.U.A. 1991

DIVISION Incorporated

www.division.com

- ✓ DVS User Guide
- ✓ DVS Geometry Tools User Guide
- ✓ dVISE User Guide

SENSE8 Web Page

http://www.sense8.com/

AGUIRRE ROMERO, JOAQUÍN Mª.

"Artes de la memoria y realidad virtual."

http://www.ucm.es/info/especulo/numero2/memoria.htm

GONZÁLEZ GUIZAR, ALBERTO. Club VR.

"Historia de la Realidad Virtual"

http://exodus.dcaa.unam.mx/virtual/history1.html

UGALDE CORRAL, HÉCTOR.

" En realidad ... "

http://amadeus.spin.com.mx/~hugalde/en-real.html

Anexo. Código de interactividad

En el presente anexo, presentamos las líneas de código que muestran la programación de la interactividad de los controles de la pantalla que se encuentra en el museo del ambiente. Esta sección es la más interesante en cuanto a interactividad se refiere, ya que involucra la programación de eventos predefinidos en el ambiente y eventos creados específicamente para el ambiente.

Los controles que aquí se programan permitirán al usuario visualizar en la pantalla la imagen que desee o, incluso, apagar la pantalla.

```
System {
  UserEvent { Ciudades }
  UserEvent { UndefinedEvent }
  UserEvent { psarq }
  UserEvent { psast }
  UserEvent { pscos }
  UserEvent { pseco }
  UserEvent { psorg }
  UserEvent { psrel }
  UserEvent { psves }
  UserEvent { ptfau }
  UserEvent { ptflo }
  UserEvent { ptgeo }
  UserEvent { phist }
  UserEvent { pciud }
  UserEvent { inicia }
  UserEvent { final }
}
```

En este segmento de código, se definen los eventos mediante los cuales se dispararán los cambios de textura de la pantalla; al darse cada uno de ellos, la textura de la pantalla cambiará de acuerdo con el evento.

Los eventos definidos deberán pertenecer a algún objeto. Por su carácter de definidos por el usuario, sólo se ejecutarán cuando sean explícitamente llamados por lo que el objeto a que sean asignados no tiene importancia, en este caso, los eventos fueron incluidos en el botón *onoff* que, además, tiene asignado el evento

pick (predefinido por el sistema y que se dispara al ser seleccionado) mediante el cual cambia la propiedad visual de la pantalla de tal forma que se apague o prenda alternativamente.

```
Object (Name=onoff) {
  Position {155.884, 156.489, -206.611}
  Scale {0.00791406, 0.00791405, 0.00791406}
  Orientation {90, 0, -135}
  Visual {
   Geometry {"finalmus42"}
  }
  Collision {
   State {On }
  }
  Constraints {
   LockX
   LockY
   LockZ
   LockRoll
   LockPitch
   LockYaw
  }
  Event {
   Pick { dvObjectVisual(pantalla, NULL, Toggle, NULL, NULL, NULL);}
   psarq { dvObjectVisual(pantalla, NULL, On, NULL, "psarq", NULL);}
   psast { dvObjectVisual(pantalla, NULL, On, NULL, "psast", NULL);}
   pscos { dvObjectVisual(pantalla, NULL, On, NULL, "pscos", NULL);}
   pseco { dvObjectVisual(pantalla, NULL, On, NULL, "pseco", NULL);}
   psorg { dvObjectVisual(pantalla, NULL, On, NULL, "psorg", NULL);}
   psrel { dvObjectVisual(pantalla, NULL, On, NULL, "psrel", NULL);}
   psves { dvObjectVisual(pantalla, NULL, On, NULL, "psves", NULL);}
   ptfau { dvObjectVisual(pantalla, NULL, On, NULL, "ptfau", NULL);}
   ptflo { dvObjectVisual(pantalla, NULL, On, NULL, "ptflo", NULL);}
   ptgeo { dvObjectVisual(pantalla, NULL, On, NULL, "ptgeo", NULL);}
   phist { dvObjectVisual(pantalla, NULL, On, NULL, "phist", NULL);}
   pciud { dvObjectVisual(pantalla, NULL, On, NULL, "pciu", NULL);}
   inicia {
    dvAssign("@i", "0");
    dvObjectVisual(pantalla, NULL, On, NULL, "pciu", NULL);
   }
   final {
    dvAssign("@i", "12");
    dvObjectVisual(pantalla, NULL, On, NULL, "ptgeo", NULL);
   }
  }
}
```

En este código, se aprecia también la definición de algunas propiedades del objeto como su posición, tamaño, orientación y restricciones de movimiento, además de la propiedad *collision* que le permite detectar cuando el usuario lo toca o selecciona.

Una vez definidos los eventos mediante los cuales la pantalla cambia de textura, es necesario dispararlos mediante los eventos *pick* de los botones de control, esto se hace mediante la acción *send event*. Es de destacarse que, para determinar que evento corresponde lanzar, es necesario tener registro de la imagen que se encuentra desplegada, esto se logra mediante el lanzamiento condicional de los eventos de tal forma que solo se envía aquel cuyo índice corresponda al contador. Este contador es incrementado (en el caso de *fwd*) mediante la acción *dvAssign* que se lanza invariablemente.

```
Object (Name=fwd) {
  Position {155.794, 156.842, -206.7}
  Scale {0.00791406, 0.00791405, 0.00791406}
  Orientation {90, 0, -135}
  Visual {
   Geometry {"finalmus44"}
  }
  Constraints {
   LockX
   LockY
   LockZ
   LockRoll
   LockPitch
   LockYaw
  }
  Collision {
   State {On }
  }
  Event {
   Pick {
    dvAssign("@i", "add(@i,1)");
    dvCall("equ(@i,1)", onoff, phist);
    dvCall("equ(@i,2)", onoff, psarq);
    dvCall("equ(@i,3)", onoff, psast);
    dvCall("equ(@i,4)", onoff, pscos);
    dvCall("equ(@i,5)", onoff, pseco);
    dvCall("equ(@i,6)", onoff, psorg);
```

```
      dvCall("equ(@i,7)", onoff, psrel);
      dvCall("equ(@i,8)", onoff, psves);
      dvCall("equ(@i,9)", onoff, ptfau);
      dvCall("equ(@i,10)", onoff, ptflo);
      dvCall("equ(@i,11)", onoff, ptgeo);
      dvCall("equ(@i,12)", onoff, inicia);
    }
   }
  }
```

Por último, para lograr colocar la secuencia de imágenes al inicio o al final de la misma se lanzan los eventos *inicia* y *final*, que también fueron definidos dentro del objeto *onoff* y que colocan la textura correspondiente en la pantalla además de establecer el valor adecuado en el contador. Estos eventos son lanzados en los botones *bbwd* y *ffwd*. A continuación se muestra el caso del *ffwd*.

```
Object (Name=ffwd) {
  Position {155.579, 156.842, -208.915}
  Scale {0.00791406, 0.00791405, 0.00791406}
  Orientation {90, 0, -135}
  Visual {
   Geometry {"finalmus43"}
  }
  Constraints {
   LockX
   LockY
   LockZ
   LockRoll
   LockPitch
   LockYaw
  }
  Collision {
   State {On }
  }
  Event {
   Pick { dvObjectEvent(onoff, final);}
  }
 }
```