Facultad 5

Herramienta para distribuir objetos prefabricados sobre una superficie en Blender

Trabajo de diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autor: Yuniel Castro González

Tutor: Ing. Alexis Echemendía González

Co-Tutor: Ing. Loyda Cárdenas Rey

La Habana, junio 2011

"Año 53 de la Revolución"

El que no aplique nuevos remedios debe esperar nuevos males, porque el tiempo es el máximo innovador.

Sir Francis Bacon

Declaración de autoría

Declaro que soy el único autor del presente trabajo y autorizamos a la Universidad de las Ciencias Informáticas los derechos patrimoniales del mismo, con carácter exclusivo. Para que así conste firmamos la presente a los ___ días del mes de _____del año _____.

_____

Yuniel Castro González

Autor

_____ _____

Ing. Alexis Echemendía González

Tutor

Ing. Loyda Cárdenas Rey

Co-Tutor

Datos de contacto

Tutor: Ing. Alexis Echemendía González.

Edad: 26 años.

Ciudadanía: Cubano.

Institución: Universidad de las Ciencias Informáticas.

Título: Ingeniero en Ciencias de Computación.

Categoría Docente: Profesor Instructor.

E-mail: aechemendia@uci.cu

Graduado de la Universidad de Ciencias Informáticas.

Co-Tutor: Ing. Loyda Cárdenas Rey.

Edad: 24 años.

Ciudadanía: Cubano.

Institución: Universidad de las Ciencias Informáticas.

Título: Ingeniero en Ciencias de Computación.

Categoría Docente: Recién Graduado en Adiestramiento.

E-mail: lrey@uci.cu

Graduado de la Universidad de Ciencias Informáticas.

A mis padres:

Por ser guías en cada uno de los momentos de mi vida, por todo el amor que han sabido darme sin pedir nada a cambio, por velar incondicionalmente por mi correcta formación como persona y profesional. A ellos me debo, por ellos soy y para ellos seré.

A mi familia:

Por apoyarme en todo momento para seguir adelante en mi futuro como profesional, por estar presente en las buenas y las malas situaciones, en especial para mis tías y primas, muchas gracias.

A todos mis amigos:

Por aguantarme y compartir incontables momentos que serán inolvidables con el transcurso del tiempo, por hacer de la universidad la mejor etapa de mi vida como estudiante. Son muchos que pudiera nombrar pero mi hermanito Raidel y Juan Miguel (el flaco) se llevan los mayores méritos, gracias por estar siempre que los necesité, de verdad.

A mi tutor:

Por brindarme incondicionalmente su tiempo y todo su talento para apoyarme en el desarrollo de esta investigación.

A mis padres:

Por darme fuerzas a cada minuto para enfrentarme a todos los retos de mi vida social y estudiantil, por ser ejemplo de tenacidad, sacrificio y modestia en todas las situaciones vividas, por ser mis ídolos.

A mis amistades:

Por contribuir a que mi andar por la vida estudiantil haya sido excelente, por compartir estudios, fiestas, momentos tensos y felices, y cada uno único e inolvidable.

A todos los profes que han puesto su mano para forjarme como profesional.

A la Revolución por darme la oportunidad de estudiar en la UCI y culminar mis estudios en Ciencias Informáticas.

Resumen

Desde los inicios de la creación de software muchos desarrolladores han enfocado su trabajo en busca de herramientas que permitan simular los entornos tridimensionales reales en el mundo virtual. Es por ello que han surgido programas capaces de cumplir con esta ambiciosa tarea y cada uno de ellos buscando optimizar al máximo la creación de los elementos necesarios, en especial los que se repiten con frecuencia en el ambiente natural. El objetivo de este trabajo consiste en elaborar una funcionalidad para distribuir objetos similares sobre una superficie, dando el mayor toque de naturalidad posible.

Para cumplir con los objetivos de esta investigación se hizo un análisis de las diferentes técnicas de dibujado y distribución de objetos presentes en los software de mayor repercusión en el mercado internacional, además se hace un estudio de las diferentes vías que existen para crear una funcionalidad que sea capaz de distribuir objetos sobre una superficie desde Blender.

Como resultado se obtuvo una herramienta que minimiza el tiempo de desarrollo de una escena que requiera elementos distribuidos, así como una mejor precisión y naturalidad en el modelado de la misma, cumpliendo de esta forma con la necesidad existente en la línea de producción “Imágenes y Videos Basados en Rendering” de la facultad 5.

Palabras clave:

Entorno tridimensional, distribución.

Introducción

Desde hace algunos años a escala mundial se vienen utilizando programas capaces de simular escenas de la vida cotidiana en tres dimensiones con un nivel de realismo que cada día alcanza mayor calidad, tratando así de disminuir la frontera existente entre el mundo real y el mundo virtual.

En muchas esferas de la vida moderna se utilizan modelos en tres dimensiones que son llevados al mundo virtual haciendo uso de programas creados para ello, se pueden mencionar los amplios campos del cine y la televisión, el desarrollo de videojuegos, así como el modelado arquitectónico de edificaciones. Cada uno de estos escenarios implica ambientes variados y diversos, tanto naturales como artificiales, donde se pueden divisar gran cantidad de objetos que sin lugar a dudas se repiten de tal forma que logran una imagen de igualdad entre ellos.

Se hace muy común que cualquier entorno tridimensional pueda mostrar elementos distribuidos tales como: hierbas, piedras, cúmulos de rocas, bosques y arena, los cuales guardan similitud entre sí, por esta razón existen herramientas que minimizan el cumplimiento de esta tarea ya que permiten al diseñador realizar un dibujado o distribución con objetos y que todos queden iguales o semejantes según la necesidad existente. Estos programas se han ido desarrollando con el paso de los años hasta alcanzar lo que hoy son, poderosas herramientas, con gran cantidad de funcionalidades que contribuyen a su fácil manejo por parte de los usuarios. Se pueden mencionar algunos programas que han ganado reconocimiento internacional dada la gran diversidad de herramientas que poseen, es el caso de Autodesk 3ds Max, Autodesk Maya, RHINOCEROS, AutoCad, Cinema 4d, Softimage 3d y Blender. Todos poseen funciones básicas para el trabajo con primitivas (por ejemplo: cajas, esferas y conos) y su posterior transformación para llegar a los modelos deseados, además han adquirido con el transcurso del tiempo otras funciones que agilizan y combinan las básicas en una sola.

La Facultad 5 de La Universidad de las Ciencias Informáticas cuenta con una línea de producción denominada “Imágenes y Videos Basados en Rendering”,

que tiene como objetivo el desarrollo de escenarios tridimensionales, muchos de los cuales comúnmente presentan elementos distribuidos entre los que se pueden mencionar personajes, árboles, hierbas y rocas, es por ello que el programa que se utilice para diseñar puede ser más efectivo si se tiene una herramienta integradora capaz de cumplir con el dibujado con elementos prefabricados. Existen tres programas que son los más ajustados a las necesidades del proyecto y a la vez los más utilizados, éstos son: Blender, Autodesk 3ds Max y Autodesk Maya, por sus condiciones, se consideran entre los mejores a escala mundial, pero los dos últimos cuentan con la negativa de ser Software Propietario. En la actualidad la Universidad de las Ciencias Informáticas se encuentra inmersa en un proceso que pretende lograr a corto plazo la migración a programas de Software Libre. Para facilitarle el trabajo a los diseñadores y que los cambios que conlleva este proceso no tomen de sorpresa al colectivo de trabajo de la línea se adopta el software denominado Blender, que posee una gran cantidad de funcionalidades que posibilitan cualquier tipo de desarrollo tridimensional y esto hace que se pueda igualar a la mayoría de las herramientas 3D más costosas del mercado, pero carece de algunas herramientas que bien pudieran contribuir con los objetivos de la línea "Imágenes y Videos Basados en Rendering". Una funcionalidad carente en dicho software y que es motivo de este trabajo es la posibilidad de que el diseñador pueda distribuir un objeto específico ya elaborado sobre una superficie cualquiera, pudiendo manejar los parámetros básicos de diseño de los nuevos objetos generados. Por lo antes mencionado surge la necesidad de crear un plugin que realice una distribución de objetos predeterminados a lo largo de una superficie.

La presente investigación tiene como **problema científico**: ¿Cómo distribuir objetos prefabricados en una superficie de manera automática?

Para solucionar esta interrogante se plantea como **objeto de estudio** las herramientas de dibujado y distribución de software dedicados al desarrollo de entornos 3D y como **campo de acción** las funciones para dibujar y distribuir a partir de primitivas en superficies tridimensionales.

El **objetivo** es implementar un plugin para Blender que permita distribuir objetos 3D en cualquier escena y sobre cualquier superficie, dando esto la posibilidad de desarrollar el arte de los diseñadores de la línea de producción.

Para dar cumplimiento al objetivo planteado, se trazaron las siguientes tareas:

- Descripción de software más usados en el desarrollo de entornos tridimensionales.
- Descripción de funcionalidades de dibujado con objetos prefabricados presentes en software propietarios de alto impacto en el desarrollo de juegos y cine.
- Descripción de funcionalidades de distribución con objetos prefabricados presentes en software propietarios de alto impacto en el mercado.
- Fundamentación de la propuesta de solución, basado en la búsqueda de los conceptos relacionados al dominio del problema.
- Descripción de las metodologías y lenguajes a utilizar para este tipo de desarrollo.
- Identificación de las herramientas y lenguajes para el desarrollo del componente.
- Definición de la metodología de desarrollo de software a utilizar.
- Implementación del plugin que distribuya objetos prefabricados.

El documento se encuentra estructurado en 3 capítulos. En el primer capítulo, “Fundamentación Teórica”, se hace una análisis bibliográfico de algunas de las mejores herramientas de diseño 3D del mercado actual para estudiar los antecedentes del plugin a implementar, además se hacen descripciones de las posibles metodologías y lenguajes a utilizar en el desarrollo del plugin. En el segundo capítulo, “Características del Sistema”, se describen las posibles técnicas que pudiera adoptar la solución propuesta para seleccionar la más adecuada para el problema en cuestión. En el tercer capítulo, “Diseño e Implementación”, se hace un análisis más profundo de la ingeniería a utilizar en la solución (modelos, captura de requisitos, diagramas) y de la implementación final del módulo. Por último se brinda un glosario de términos para ayudar al lector en la comprensión del lenguaje técnico utilizado en la investigación.

Capítulo 1

Fundamentación teórica

En este capítulo se realiza un análisis de las herramientas y metodologías propuestas para llevar a cabo el trabajo, además se hace un breve bosquejo del estado del arte relacionado con la investigación y se muestran los conceptos fundamentales estudiados que circundan el entorno de trabajo en cuestión. También se hace referencia a las metodologías de desarrollo y lenguajes de programación estudiados para el desarrollo del sistema. Además se presentan las principales herramientas de diseño que son utilizadas en nuestra facultad para la realización de los diferentes proyectos.

1.1 Software dedicados al desarrollo de entornos tridimensionales

1.1.1 Descripción de Autodesk 3ds Max

Este software es catalogado como la herramienta preferida por los profesionales que son considerados líderes a nivel mundial en el desarrollo de juegos, televisión, cine y composición digital que quieren una solución 3D completa de modelado, animación, renderización y composición con resultados inmediatos [1].

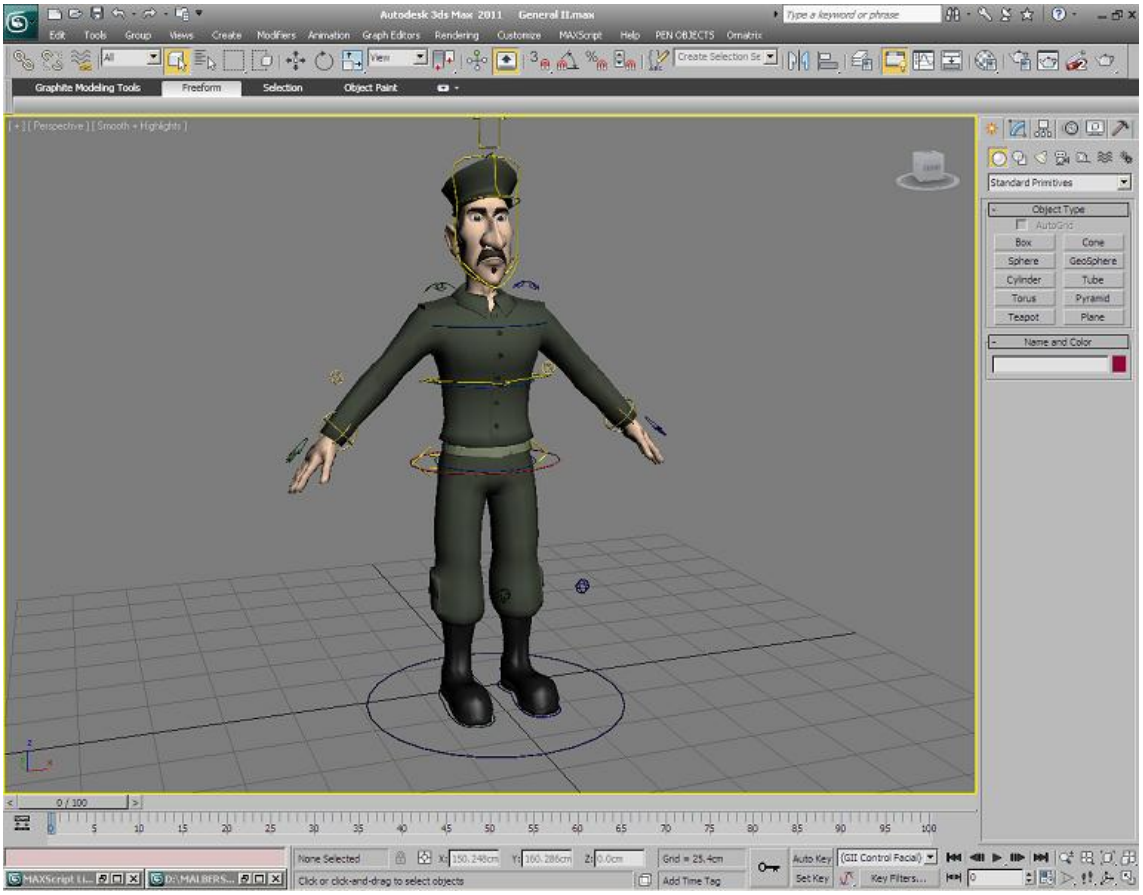


Figura 1 Interfaz de Autodesk 3ds Max

Entre sus características más notables podemos encontrar [2]:

- Amplio juego de herramientas de modelado 3D: Más de 100 herramientas avanzadas para modelado poligonal y diseño de formas libres en 3D.

- Sombreado y texturización: Gran variedad de opciones para pintar texturas, mapearlas y asignarlas a capas.
- Animación: Herramientas muy avanzadas para crear personajes y animaciones 3D de alta calidad.
- Dinámica, efectos y simulación: Herramientas de producción con alto rendimiento para crear efectos y dinámica.
- Potente funcionalidad de renderización 3D
- Integración en la estructura productiva: Importación de datos de numerosos orígenes y transferencia de información de 3ds Max y 3ds Max Design entre archivos, aplicaciones, usuarios y ubicaciones.
- Flujos de trabajo colaborativos: Recopilación y uso compartido de datos en escenas complejas, para que múltiples usuarios puedan colaborar en el flujo de trabajo.
- La nueva versión 3ds Max 2011 incluye una poderosa herramienta de dibujado con objetos prefabricados (Object Paint), que posibilita una fácil creación de elementos distribuidos en un entorno tridimensional.

1.1.2 Descripción de Autodesk Maya

Es un software dedicado a la animación, modelado, efectos visuales, renderización y composición en 3D. Ofrece flujo de trabajo creativo completo, con todas las herramientas para realizar animación, modelado, simulación, efectos visuales, renderización, rastreo de movimiento y composición en 3D dentro de una plataforma de producción sumamente ampliable. Toda esta funcionalidad se reúne en una única aplicación que aporta un valor excepcional a los artistas gráficos. Proporciona una interfaz de usuario de alto nivel, gran interacción con las ventanas gráficas, funciones de edición en 3D, gestión del color integrada y excelente animación de personajes [3].

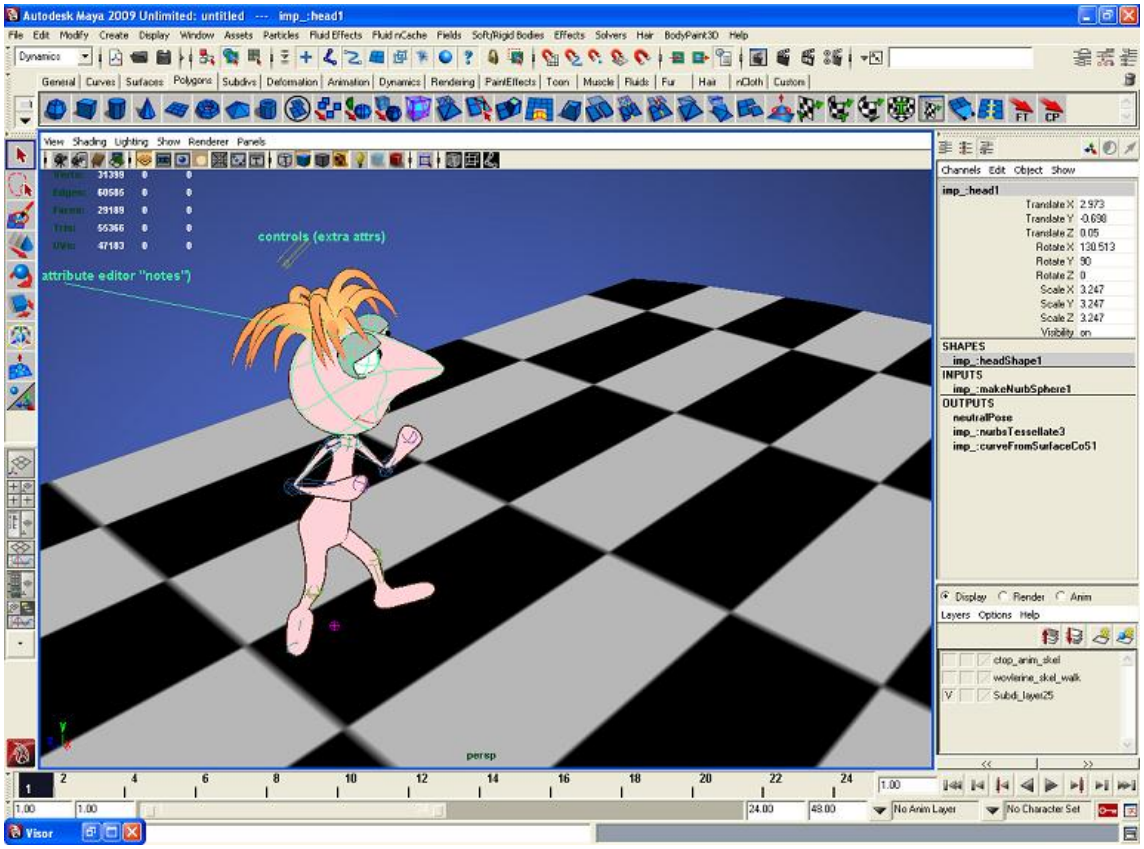


Figura 2 Interfaz de Autodesk Maya

Entre sus características más novedosas podemos encontrar [4]:

- Interfaz de usuario actualizada: experiencia de uso coherente y mejorado en todas las plataformas.
- Nuevos flujos de trabajo y herramientas de aplicación de piel: creación más rápida de personajes con mejor piel y deformaciones más realistas.
- Nuevo flujo de trabajo no destructivo al reasignar objetivos: la reutilización, corrección y mejora de la captura de movimiento y otros datos de animación son más fáciles y rápidas gracias al nuevo flujo de trabajo no destructivo de reasignación de objetivos.
- Nuevas funciones de edición en 3D: aceleran la previsualización y la producción cinematográfica virtual.
- Mejor referenciación de archivos y activos: es más fácil segmentar, reutilizar e intercambiar datos con y sin referenciación de archivos.

- Posee una herramienta de dibujado con objetos prefabricados (Paint Effects) que tiene una gran gama de utilidades en este aspecto.

1.1.3 Descripción de Cinema 4D

Cinema 4D es un programa de diseño y animación 3D desarrollado originariamente para Commodore Amiga por la compañía alemana Maxon, y portado posteriormente a plataformas Windows, Linux y Macintosh. Satisface las necesidades profesionales de modelado 3D, texturizado, animación y renderizado. Ofrece una configuración personalizable adecuada para cualquier industria y a todos los niveles de cualificación [5].

Figura 3 Interfaz de Cinema 4D

Se pueden mencionar características sobresalientes, tales como [6]:

- El programa dispone de un potente modelador y de una amplia gama de funciones y efectos especiales para la presentación de proyectos de arquitectura e ingeniería, el diseño en 3D a cualquier nivel, la animación fotorrealista, la simulación científica, el desarrollo de entornos virtuales y la realización de efectos especiales para el cine y la televisión.
- Está basado en algoritmos de cálculo que le dotan de una incomparable velocidad de renderizado, y que unidos a un intuitivo sistema de modelado, convierten al programa en una herramienta de muy alto rendimiento.
- Permite realizar, mediante imágenes fotorrealistas y animaciones, convincentes presentaciones virtuales de diseños y proyectos.
- La creación e inserción de luces puede realizarse seleccionando entre diferentes tipos de fuentes luminosas y distintos parámetros, incluyendo luces superficiales, volumétricas.
- Se pueden utilizar efectos especiales como la profundidad de campo, desenfoque por movimiento, efectos ambientales como niebla o viento, sonido estéreo, entre otros.
- Dadas sus prestaciones, versatilidad y facilidad de uso, este programa puede considerarse entre los mejores en la relación calidad-precio del mercado.
- Es capaz de importar y exportar una gran variedad de formatos (DXF, 3DS, VRML...), y genera directamente QTVR.
- Permite modelado (primitivas, splines, polígonos), texturización y animación. Sus principales virtudes son una muy alta velocidad de renderización, una interfaz altamente personalizable y flexible.

1.1.4 Descripción de Blender

Es un software multiplataforma, dedicado especialmente al modelado, animación y creación de gráficos tridimensionales. Cuenta con un intérprete de Python por lo que se puede programar mediante scripts. Tiene una muy peculiar interfaz gráfica de usuario, que se critica como poco intuitiva, pues no se basa en el sistema clásico de ventanas; pero tiene a su vez ventajas importantes sobre éstas, como la configuración personalizada de la distribución de los menús y vistas de cámara [7].

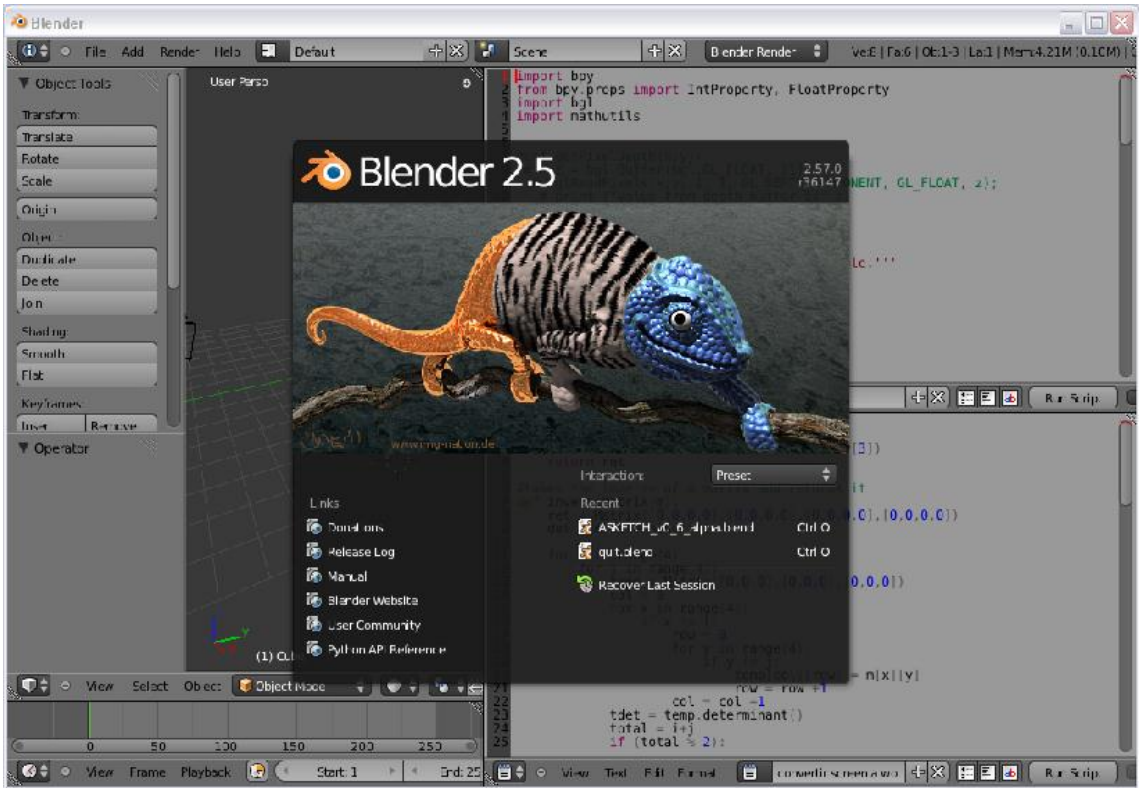


Figura 4 Interfaz de Blender 2.57

Posee características novedosas como [7]:

- Multiplataforma, libre, gratuito y con un tamaño de origen relativamente pequeño.
- Capacidad para una gran variedad de primitivas geométricas, incluyendo curvas, mallas poligonales, vacíos, NURBS.

- Junto a las herramientas de animación incluye cinemática inversa, deformaciones por armadura o cuadrícula, vértices de carga y partículas estáticas y dinámicas.
- Edición de audio y sincronización de video.
- Detección de colisiones, recreaciones dinámicas y lógica para juegos.
- Posibilidades de renderizado interno versátil e integración externa con potentes trazadores de rayos libres.
- Lenguaje Python para automatizar o controlar varias tareas.
- Motor de juegos 3D integrado.
- Simulaciones dinámicas para partículas y fluidos.
- Modificadores apilables, para la aplicación de transformación no destructiva sobre mallas.
- Sistema de partículas estáticas para simular cabellos y pelajes.

1.2 Herramientas de dibujado con objetos prefabricados presentes en otros software de diseño tridimensional.

De reconocida calidad en el mundo del diseño tridimensional se pueden mencionar los software Autodesk 3ds Max y Autodesk Maya, los cuales poseen entre sus módulos potentes herramientas de dibujado que pudieran considerarse antecedentes del plugin que se pretende incorporar a Blender, es por ello que a continuación se detallan algunas de sus características.

1.2.1 Object Paint de Autodesk 3ds Max

Object Paint es una función que resulta muy útil para crear grupos de objetos, ya sea distribuidos o en un patrón particular. Esta herramienta convierte a uno o varios objetos determinados en un pincel. Facilita la creación de objetos repetidos, texturizados, por el escenario, para realizar un entorno tridimensional más rápido [8].

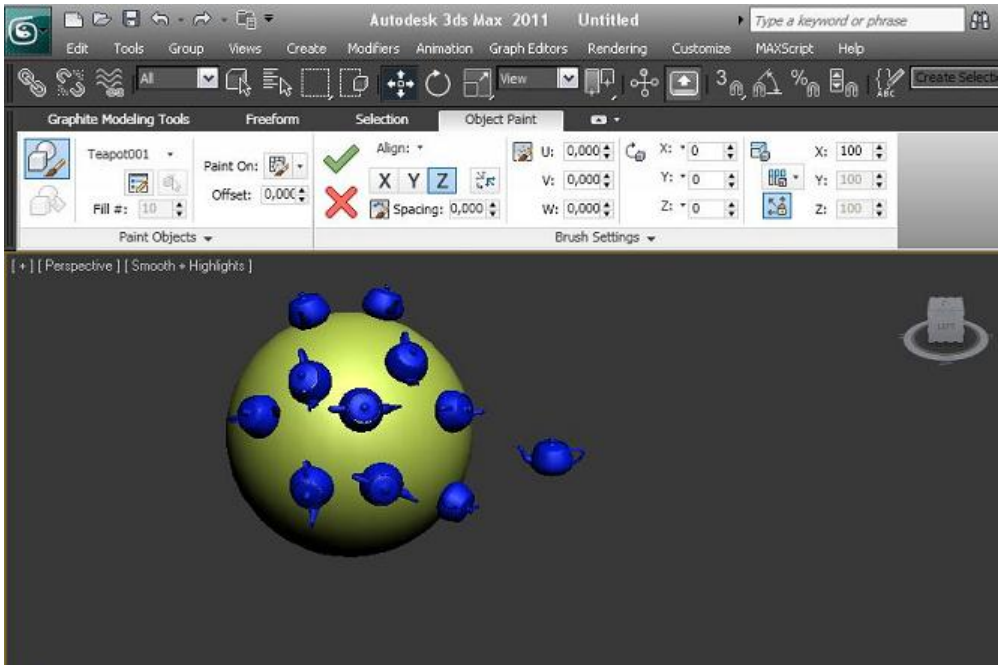


Figura 5 Herramienta Object Paint de Autodesk 3ds Max

Esta herramienta se utiliza para pintar objetos hacia la escena en cuestión o sobre objetos específicos, ubicando los objetos en una trayectoria trazada a pulso y/o a lo largo de una superficie seleccionada por el diseñador.

Para realizar el dibujado se requiere la creación de un objeto (o varios) que será el replicado en la escena, si se quiere que la distribución sea de manera libre hay que señalar el Grid, en caso de que la necesidad sea que los nuevos objetos estén adheridos a una superficie pues hay que crearla previamente, acto seguido se marca cual será el objeto a clonar y se distribuye por las trayectorias escogidas por el diseñador. Es importante aclarar que existe la posibilidad de ajustar algunos parámetros después de realizar el dibujado y antes de concluir, o sea, se varían los parámetros de posición, tamaño, etc., escogidos y luego de satisfecho el diseñador pues concluye con un clic

derecho del mouse. Además, cada uno de los nuevos objetos creados puede ser tratado o no como una instancia del original, o sea, puede ser modificado por separado o todos los objetos a la vez, lo que constituye una gran ventaja para la confección de la escena deseada.

A grandes rasgos la parte lógica de funcionamiento de la herramienta Object Paint es la siguiente:

- Selección del o los objetos determinados a actuar como pincel y guardarlos en variables destinadas para ello.
- (OPCIONAL) Selección de la superficie donde se desplegarán los clones y se guarda en una variable destinada para ello.
- Manejo de los parámetros mostrados en la interfaz, los cuales serán almacenados para tenerlos en cuenta en la creación de cada uno de los clones, dígase escala, rotación, alineación con los ejes, entre otros.
- Se traza la trayectoria deseada por el diseñador y se almacenan cada uno de los puntos que componen dicha curva para en ellos ubicar los nuevos clones.
- En caso de que la distribución se haga sobre una superficie se hacen los cálculos del vector normal correspondiente a cada uno de los puntos de la curva, lo cual denotará la inclinación y rotación del clon que se ubique en cada uno de estos puntos.

1.2.2 Paint Effects de Autodesk Maya

Paint Effects es un poderoso y complejo módulo que posibilita al diseñador la realización de innumerables tareas de dibujado, se hace uso de un pincel para generar cualquier geometría imaginable, además de muchos efectos.

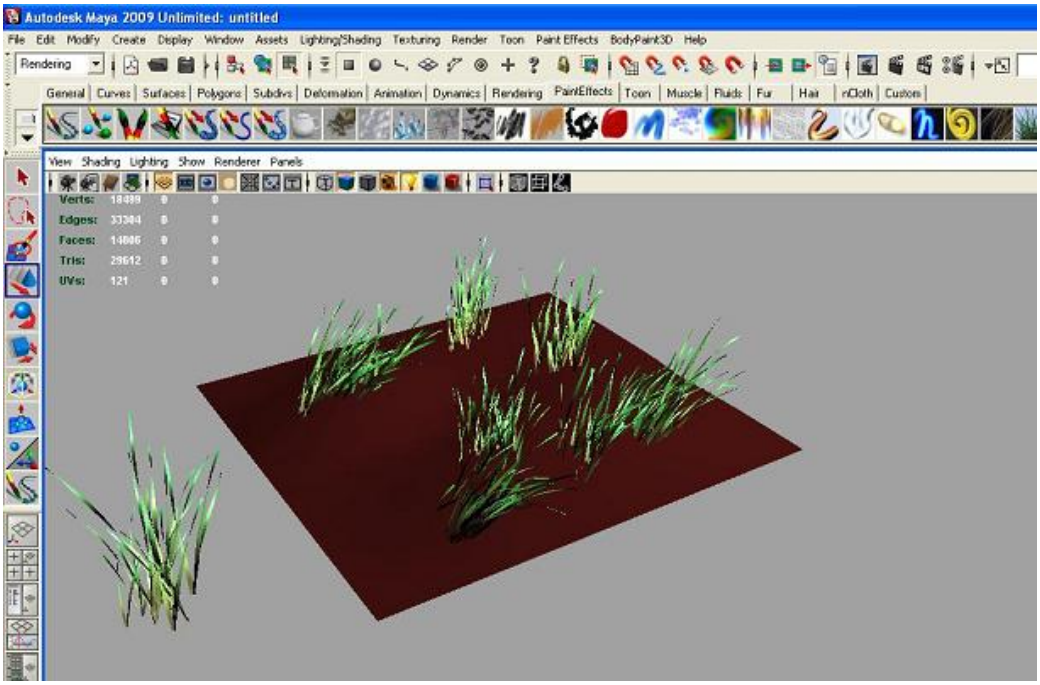


Figura 6 Herramienta Paint Effects de Autodesk Maya

Pudieran mencionarse algunas de las funciones útiles de la herramienta: permite pintar trazos por trayectorias definidas a decisión del diseñador, para ello están implícitos una serie de pinceles que poseen atributos predefinidos y de acuerdo a éstos se aplica la pintura o efecto deseados; se pueden distorsionar pinturas distribuidas en la escena o en superficies, creando también sombras distorsionadas; es capaz de suavizar la apariencia de pinturas distribuidas en la escena o en superficies, creando también sombras borrosas; permite describir una trayectoria para eliminar los pixeles pintados y mantener intactos los colores anteriores de esa región, esto puede ser muy útil para borrar los agujeros en una textura o una escena; se pueden crear muchos líneas o tubos con parámetros modificables que pueden reducir el tiempo y trabajo a la hora de crear muchas escenas, pudiendo mencionarse en el caso de tubos finos la inclusión y detallado de pelaje a algún objeto y el caso de tubos gruesos pudiera simular conjuntos de manchas; es posible representar efectos de pintura que utilizan tubos triangulados, esto da lugar a la geometría cónica precisa con texturas sobre superficies; pueden crearse efectos de pintura de árboles y plantas que no son convincentes sólo desde la distancia, sino también de cerca; es posible crear formas tales como la geometría de bordes duros y de iluminación por píxel en la malla; otra posibilidad que brinda

la herramienta es la simulación de superficies reflectantes además que para no sobrecargar memoria los triángulos son generados en tiempo de render [9].

A grandes rasgos la parte lógica de funcionamiento de la herramienta Paint Effects es la siguiente:

- Selección del pincel deseado y se guardan los valores por defecto del mismo en variables destinadas para ello.
- Ajuste a conveniencia del diseñador de los parámetros del pincel antes, durante o después de realizado el dibujado con el mismo.
- Se traza la trayectoria deseada por el diseñador y se almacenan cada uno de los puntos que componen dicha curva para en ellos ubicar las pinceladas realizadas.
- Si el dibujado se hace sobre una superficie se calcula el vector normal de cada uno de los puntos de la curva que coincidan con la superficie y esto determina la inclinación y rotación de la pincelada en cada uno de estos puntos. Este parámetro es modificable posteriormente.

1.3 Herramienta de distribución de objetos prefabricados usada para el diseño tridimensional.

El software Autodesk 3ds Max posee entre sus módulos una potente herramientas de distribución que bien pudiera considerarse lo más cercano a la idea del plugin a implementar para Blender, es por ello que a continuación se detallan algunas de sus características.

1.3.1 Scatter de Autodesk 3ds Max

El Scatter de 3ds Max es una poderosa herramienta que realiza una dispersión al azar de un objeto seleccionado, ya sea una distribución tipo arreglo o bien sobre una superficie, también seleccionada [10].

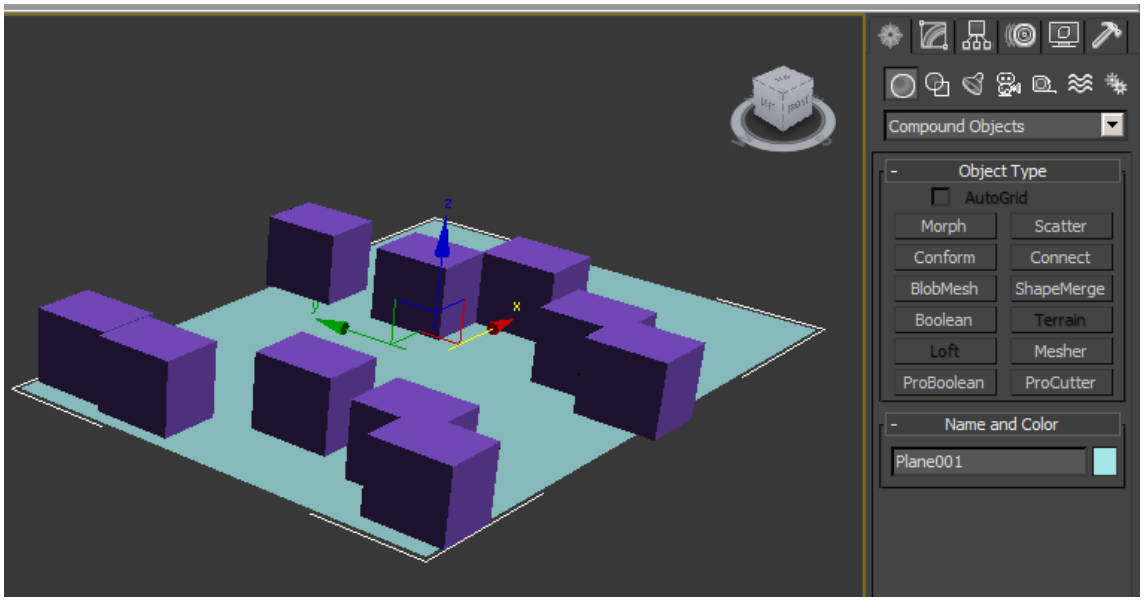


Figura 7 Herramienta Scatter de Autodesk 3ds Max

La mecánica de funcionamiento es sencilla, se crea un objeto que será el escogido para distribuir, opcionalmente se selecciona una superficie sobre la cual realizar la dispersión de objetos y se ejecuta la acción “Scatter”. El objeto de origen debe ser un objeto de malla o un objeto que se pueda convertir en un objeto de malla. Si el objeto seleccionado no es válido, el botón de dispersión no estará disponible [10].

Posee además gran cantidad de opciones en cuanto a modificación de parámetros como: cantidad de copias; rotación; escala; si se hace copia, instancia o referencia; si se ejecuta distribución o transformación de objetos; si la distribución se desea sea perpendicular haciendo caso omiso al cálculo de la normal de la superficie donde se va a colocar cada duplicado que indica la inclinación del objeto; entre otros. Todo esto se puede ejecutar antes o después de realizada la dispersión de objetos [10].

1.4 Descripción de posibles metodologías a utilizar

En el mundo informático actual se persigue el objetivo de que la mayoría de los proyectos terminen con éxito, para ello se hace necesario que el proceso de desarrollo de cada uno de ellos se realice de manera eficiente. Para lograr un software con calidad es requisito indispensable que en cada una de sus etapas el proceso mantenga un estricto control de las actividades a realizar.

En los últimos años los procesos de desarrollo han aumentado considerablemente en cantidad y variedad, surgiendo a su vez dos corrientes fundamentales, los llamados métodos ligeros y métodos pesados. La principal diferencia entre ambos es que mientras los métodos ligeros tratan de mejorar la calidad del software por medio de una comunicación directa e inmediata entre las personas que intervienen en el proceso, los métodos pesados intentan conseguir el objetivo común por medio de orden y documentación. Puede considerarse a XP como la metodología líder en la vertiente de métodos ligeros y RUP en la de métodos pesados.

1.4.1 Extreme programming (XP)

Sin dudas una metodología de desarrollo de software muy utilizada en la actualidad, eficiente preferiblemente para proyectos cuyo equipo de desarrollo es pequeño y/o el plazo de entrega es corto. Consiste en una programación rápida o extrema y se caracteriza por tener como parte del equipo de desarrollo al usuario final, lo cual constituye un factor vital para el éxito del proyecto [11].

Que plantea XP?

- El manejo del cambio se convierte en parte sustantiva del proceso.
- El costo del cambio no depende de la fase o etapa.
- No introduce funcionalidades antes que sean necesarias.
- El cliente o el usuario se convierte en miembro del equipo.

1.4.2 Rational Unified Process (RUP)

El Proceso Unificado es un proceso de software genérico y su objetivo es asegurar la producción de software de alta calidad y que satisfaga las necesidades del usuario final, en un marco de tiempo determinado y un presupuesto predecible. Utiliza UML (Lenguaje de Modelado Unificado) en la elaboración de los planos para el sistema a desarrollar. Sus rasgos característicos están capturados en tres conceptos esenciales [12]:

Dirigido por casos de uso: un caso de uso es una pieza en la funcionalidad del sistema que le da al usuario un resultado de valor, por lo que capturan los requerimientos funcionales, además dirigen el diseño del sistema, implementación y pruebas. Todos los casos de uso juntos constituyen el modelo de casos de uso el cual describe la funcionalidad completa del sistema. Son desarrollados a la par con la arquitectura del sistema y ambos maduran conforme avanza el ciclo de vida del proyecto.

Está centrado en la arquitectura: la arquitectura surge de las necesidades de la empresa tal y como están reflejadas en los casos de uso. También está influenciada por la plataforma de software en la que se ejecutará, la disponibilidad de componentes, consideraciones de instalación, requerimientos no funcionales, entre otros. La arquitectura es la vista del diseño completo con las características más importantes y dejando los detalles de lado. Debe enfocarse en las metas correctas, tales como claridad, flexibilidad en los cambios futuros y reutilización.

Es Iterativo e Incremental: es práctico dividir el trabajo en pequeños pedazos o mini-proyectos. Cada mini-proyecto es una iteración que debe ser planeada y que finaliza en un incremento en el producto. En cada iteración se identifican y especifican los casos de uso relevantes, se crea el diseño, se implementa el diseño en componentes y se verifican que satisfacen los casos de uso. Si una iteración no cumple sus metas los desarrolladores deben revisar sus decisiones previas y probar un nuevo enfoque.

1.5 Lenguaje de modelado

El desarrollo de un proyecto de software pudiera causar confusiones y malas interpretaciones en los requerimientos necesarios, ya sea por la abundancia de notaciones, metodologías o conceptos que hace que los desarrolladores no se pongan de acuerdo en las ideas de lo que realmente están elaborando.

1.5.1 UML

El Lenguaje de Modelado Unificado (UML) es un lenguaje estándar para escribir planos de software. Tiene como objetivo comunicar ideas a los desarrolladores, servir de apoyo en los procesos de análisis de un problema, además que sea independiente del lenguaje de implementación, de tal forma que los diseños realizados usando UML se puedan implementar en cualquier lenguaje [13].

1.5.2 Diagramas UML

Diagrama de Clases, modela la estructura estática de las clases en el sistema [14].

Diagrama de Componentes, modela los componentes de una aplicación, sistema o empresa.

Diagrama de Objetos, modela la estructura estática de los objetos en el sistema.

Diagrama de Actividades, modela el comportamiento de los casos de uso, objetos operaciones.

Diagrama de Comunicaciones, modela interacciones entre objetos.

Diagrama de Secuencias, representa una interacción, poniendo el foco en la secuencia de los mensajes que se intercambian.

Diagrama de Casos de Uso, muestra las relaciones entre los actores y el sujeto (Sistema) y los Casos de usos.

1.6 Formas de implementación de herramientas

Para desarrollar la implementación sobre herramientas pueden aplicarse varias vías, en el caso específico de programar sobre Blender hay dos vertientes fundamentales: programar a base de scripts utilizando el intérprete de Python adjunto al software y programando en C directo al código fuente del software.

1.6.1 Script

Es un guión o conjunto de instrucciones. Permite la automatización de tareas creando pequeñas utilidades. Es ejecutado por un intérprete de línea de comandos y usualmente es un archivo de texto. Puede considerarse también una alteración o acción a una determinada plataforma, algo así como aplicar trucos que se usan para alterar acciones y conseguir resultados extras. Es sumamente utilizado para programar sobre otros programas, es decir, manipular explícitamente los datos del proyecto activo, precisamente lo que se pretende hacer sobre la herramienta Blender. Existen varios lenguajes de scripting como Visual Basic Script, JavaScript, WScript, Batch Script, C++, Python, entre otros [15].

Dependiendo de su funcionalidad, los scripts para Blender se clasifican en [16]:

Ejecución manual: sirven para realizar una tarea en el diseño del proyecto. Son los más sencillos, se ejecutan dejando el cursor sobre la ventana de texto y pulsando las teclas ALT-P. A su vez pueden tener un entorno gráfico, usando instrucciones OpenGL, y funcionando por eventos.

Ejecución automática: se ejecutan cada vez que Blender redibuja el entorno, o se cambia de frame (fotograma). Así se puede aplicar instrucciones a un objeto, por ejemplo, para darle una localización dependiendo del tiempo.

Ladrillos de Lógica: se emplean en el motor de tiempo real para la toma de decisiones, aplicación de atributos, entre otros.

1.7 Lenguajes, herramienta case e IDE a utilizar en el desarrollo del plugin

Para llevar a cabo el desarrollo de la aplicación se pueden tomar como opciones dos potentes lenguajes de programación que en la actualidad tienen gran eficiencia y aceptación por parte de los programadores a escala mundial, ellos son el C++ y Phyton, también se hace necesario el uso de herramientas case para apoyar la creación del sistema en general y un IDE que responda a los lenguajes seleccionados.

Lenguajes:

1.7.1 Descripción de C++

C++ es un lenguaje de programación orientado a objetos. Se suele decir que es un lenguaje híbrido, ya que permite la programación estructurada, además aumenta la eficiencia al no incorporar verificación de errores en tiempo de ejecución. Minimiza la sobrecarga con la implementación de polimorfismo y la expansión de patrones. Es un lenguaje de nivel intermedio, pudiéndose utilizar tanto para escribir software de bajo nivel, drivers y componentes de sistemas operativos. Apoya el desarrollo de clases genéricas con parámetros de tipo y de tamaño, lo cual es la base de la Biblioteca de Patrones Estándar, STL, que contiene una gran cantidad de patrones de clase contenedor, como mapas, conjuntos, pilas y colas, y una amplia variedad de algoritmos que pueden especializarse para tipos de datos provistos por el usuario. Posee sobrecarga de operadores y permite construir elegantes mecanismos y jerarquías de clase que controlen correctamente la creación y destrucción de objetos [17].

No puede dejar de mencionarse que el C++ es la base principal de funcionamiento del software Blender ya que el código fuente que lo respalda está escrito en este lenguaje, dejando así para su tratamiento todo el código bruto o fuerte con que cuenta este programa.

1.7.2 Descripción de Python

Python es un lenguaje de scripting independiente de plataforma y orientado a objetos. Es un lenguaje interpretado, lo que significa que no se necesita compilar el código fuente para poder ejecutarlo, lo que ofrece como ventaja la rapidez de desarrollo. Es interactivo pues dispone de un intérprete por línea de comandos en el que se pueden introducir sentencias y cada una produce un resultado visible, que ayuda a entender mejor el lenguaje y probar los resultados de la ejecución rápidamente. Permite programación orientada a procedimientos así como orientada a objetos. En lenguajes orientados a procedimientos, el programa está construido sobre funciones reutilizables y en lenguajes orientados a objetos, el programa es construido sobre objetos los cuales combinan datos y funcionalidad. Puede combinarse con C para que una pieza de código se ejecute muy rápido. Posee gran cantidad de librerías, tipos de datos y funciones incorporadas que ayudan a realizar muchas tareas habituales sin necesidad de tener que programarlas desde cero [18].

El programa Blender trae consigo un intérprete de Python, o sea, puede leer código escrito en Python, por lo cual con este lenguaje se pueden llevar a cabo gran cantidad de modificaciones en la estructura y funcionamiento de cada una de las herramientas incorporadas por defecto en dicho software, además de contar con la posibilidad de adicionarle innumerables funcionalidades adicionales que sirvan al propósito de los usuarios.

Herramienta case:

1.7.3 Descripción de Visual Paradigm

Es considerada a nivel mundial como una herramienta muy eficiente y fácil de utilizar, siendo una de sus ventajas más notables el hecho de ser multiplataforma. Utiliza como lenguaje de modelado el UML y proporciona interoperabilidad con otras aplicaciones. La instalación y actualización de la misma suelen ser procesos relativamente sencillos. Brinda una interfaz amigable y fácil de usar que tiene un sustento en varios idiomas, además de

que ofrece a los usuarios la capacidad de desarrollar la ingeniería tanto directa como inversamente [19].

IDE de desarrollo:

1.7.4 Descripción de Python IDLE

IDLE es el IDE de Python construido con el kit de herramientas GUI Tkinter. Entre sus características generales se pueden mencionar que es codificado puramente en Python, utilizando el kit de herramientas GUI Tkinter; es multiplataforma; como editor puede manejar varias ventanas de texto a la vez y brinda al usuario múltiples acciones deshacer; aporta escala de colores en la sintaxis para un mejor entendimiento por parte del usuario, así como la indexación en sus sentencias; posee un intérprete interactivo y se pueden establecer puntos de interrupción a la hora de depurar los códigos [20].

1.8 Consideraciones del capítulo

En este capítulo se ha plasmado la fundamentación teórica del trabajo donde se manejan los conceptos básicos y necesarios para el dominio del tema. Se analizan algunas herramientas de dibujado y distribución similares a la que se pretende implementar, tomando para el estudio las presentes en reconocidos software de desarrollo 3D. También se aborda sobre los lenguajes y herramientas case a utilizar y las posibles metodologías a seguir en el proceso de desarrollo del plugin.

Capítulo 2

Características del sistema

A partir del estudio teórico realizado en el primer capítulo se propone una solución específica para incorporar a Blender la herramienta que cubra las necesidades que posee este software de diseño y que es motivo de desarrollo de este trabajo.

2.1 Metodología a utilizar

Para desarrollar la propuesta que presenta este trabajo, se ha decidido utilizar como metodología el Proceso Unificado de Modelado (RUP), primeramente por ser objetivo de la institución, además de que se considera que XP es una metodología muy joven. Además, luego de haber desarrollado todo un estudio sobre diferentes metodologías, se llegó a la conclusión de que RUP es la óptima para lograr desarrollar una aplicación que satisfaga completamente los requisitos planteados.

2.2 Restricciones del sistema

En el desarrollo de un producto de software se manejan un conjunto de reglas que regulan algunos aspectos importantes del proceso, éstas se convierten en una serie de restricciones a la hora de cumplir alguna actividad y es conveniente evaluar su relevancia dentro del campo de acción que se modela. Pueden mencionarse las restricciones siguientes:

- Para poder utilizar el plugin se hace necesario tener instalado el Blender 2.5pues se utiliza el módulo ‘bpy’ que sólo trabaja en esta versión del software y no en anteriores.
- Tiene que haber algún objeto geométrico seleccionado en la escena para que pueda ser clonado y también una superficie activa para realizar la distribución del objeto.

2.3 Tipos de clase utilizados para programar sobre Blender

A la hora de programar sobre Blender los comandos son escritos sobre el intérprete de Python que trae implícito este software. El mismo acepta sus propios tipos de clase para ejecutar las acciones programadas, estas son: operadores, paneles y menús, que por defecto vienen predeterminadas en los módulos del código fuente.

Operadores: son utilizadas para manipular el contexto, o sea, realizan cambios directos a las propiedades de los objetos presentes en la escena en cuestión, pudiendo modificar temporal o permanentemente los parámetros de datos accesibles de cada componente.

Menús: destinadas a hacer agrupaciones de elementos, contenidos o funcionalidades similares, dando la posibilidad de mantener una mejor organización en la estructura de la programación.

Paneles: son las encargadas de construir y mostrar interfaces al usuario. Posee muchas opciones de configuración visual y de funcionamiento, ya que pueden conectarse con clases operadoras y clases menús para mostrar los resultados que éstas puedan devolver.

2.4 Pasos para realizar una distribución de objetos en Blender

Para realizar un duplicado de objetos en cualquier escena y sobre una superficie, en Blender, pueden adoptarse dos vías fundamentales. Una de ellas es haciendo uso del sistema de partículas implícito en los módulos que trae incorporados el software, la otra es desarrollar la distribución de manera manual.

2.4.1 Distribución usando el sistema de partículas

El sistema de partículas es una herramienta que posee el Blender para instaurar un conjunto de puntos a una superficie para luego colocar en cada uno de ellos un objeto específico y de esta forma realizar una distribución aleatoria de objetos, con ello se consigue la problemática de gran cantidad de repeticiones de cálculos en cada iteración lo cual lo hace un mecanismo que recarga mucho la memoria del sistema, además de ser bastante complejo a la hora de usarlo.

2.4.2 Distribución manual

La figura representa un ciclo que describe cada uno de los pasos necesarios para desarrollar una distribución a conveniencia del diseñador.

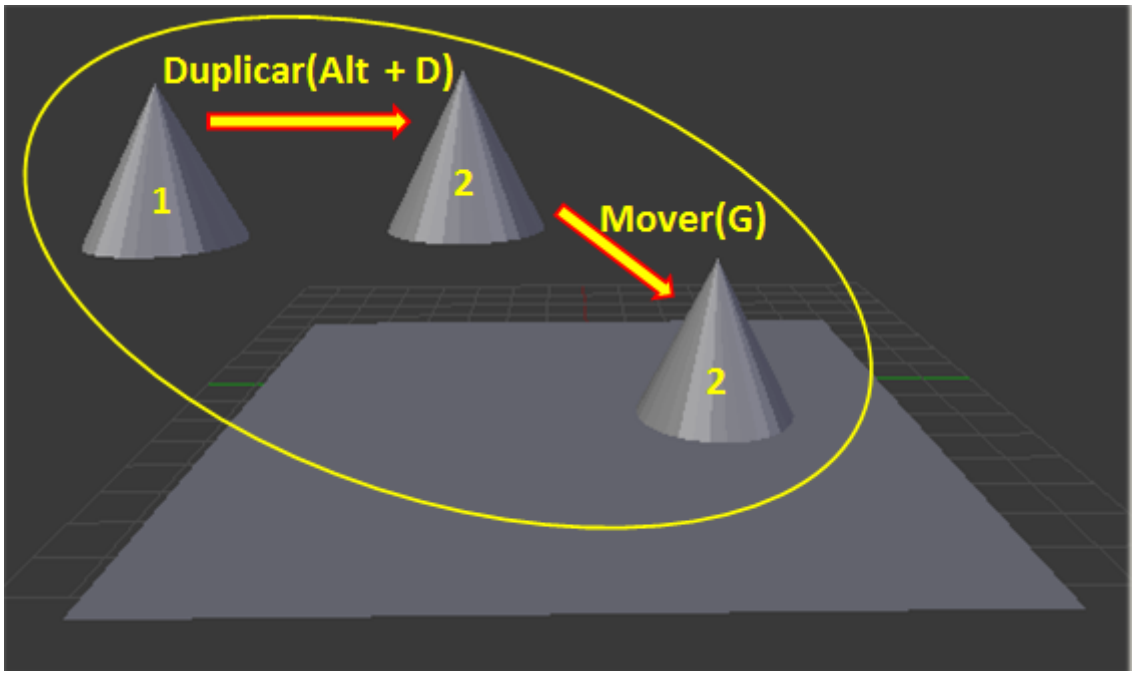


Figura 8 Ciclo para realizar distribución manual

Paso 1: duplicar el objeto deseado aplicando las opciones de interfaz que brinda Blender para ello o bien con las teclas Alt+D.

Paso 2: mover el nuevo objeto para la posición relativa a la superficie deseada y aplicar modificaciones a la rotación y escala del mismo a gusto del diseñador.

Estos pasos se repiten tantas veces como copias se quieran hacer del objeto seleccionado, convirtiéndose este proceso en una tarea muy tediosa para el diseñador dado el tiempo de desarrollo a emplear y el poco nivel de exactitud en el manejo de los parámetros escogidos.

2.5 Descripción de la solución propuesta

Para automatizar la compleja tarea de realizar una distribución de objetos sobre una superficie en Blender se proponen aspectos que circundan la idea de crearla con un alto nivel de realismo y que pueden considerarse la columna vertebral del plugin a desarrollar.

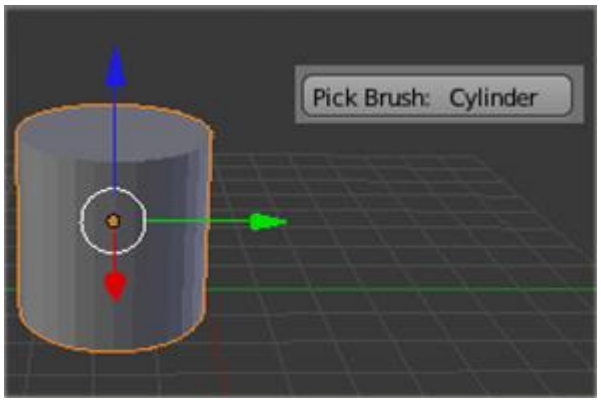
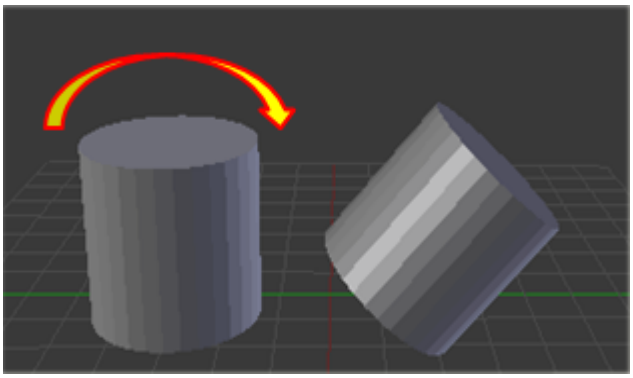


Figura 9 Seleccionar objeto

El primer paso para poder realizar el proceso de distribución debe ser seleccionar el objeto a distribuir sobre la superficie y almacenar en la memoria del sistema ese dato.

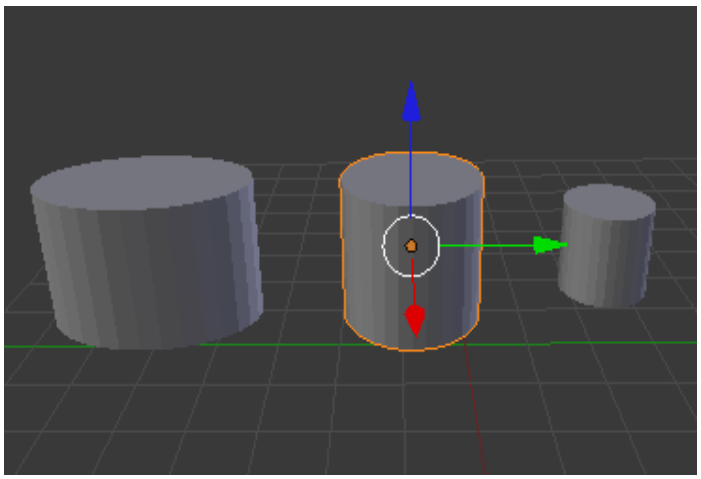
Posteriormente debe brindarse al diseñador la posibilidad de determinar la cantidad de copias del objeto que quiere duplicar sobre la superficie y guardar también ese valor. Es válido aclarar que para obtener una distribución de objetos lo más realista posible debe implementarse un mecanismo que ubique los nuevos objetos con respecto a la normal del punto donde caigan en la superficie y en contraposición a ello debe brindarse al usuario la posibilidad de violar este mecanismo, pudiendo ubicar los duplicados de manera vertical. Otra posibilidad opcional que pudiera ser de gran utilidad es la de aplicar una ligera rotación aleatoria relativa entre los objetos del conjunto generado, lo cual visualmente logra una notable distorsión.

Luego de realizada la distribución debe ser posible modificar algunos parámetros de los objetos del conjunto generado, tales como rotación y escala.



La rotación por ejes es muy cómoda para lograr distorsión y más aún si se ejecuta en sentido aleatorio para cada uno de los objetos, es por ello que esta posibilidad debe estar presente en la herramienta.

Figura 10 Rotar objeto por ejes



El escalado por ejes permite deformar el objeto duplicado rompiendo así la igualdad entre los objetos generados, mientras que si se ejecuta a los tres ejes (x, y, z) por igual se puede disminuir o aumentar proporcionalmente el tamaño de los duplicados.

Figura 11 Escalar objeto por ejes

Además de las opciones mencionadas anteriormente el diseñador debe contar con la posibilidad de redistribuir la ubicación de los duplicados, ya que como el proceso es aleatorio en ese sentido es posible que no sea de su agrado donde quedó cada objeto.

2.6 Modelo de dominio

Con el objetivo de brindar una referencia para mejorar la comprensión de los términos usados en la construcción del plugin a los diferentes usuarios, se muestra a continuación el diagrama de dominio, el cual es una guía para el futuro diseño del sistema.

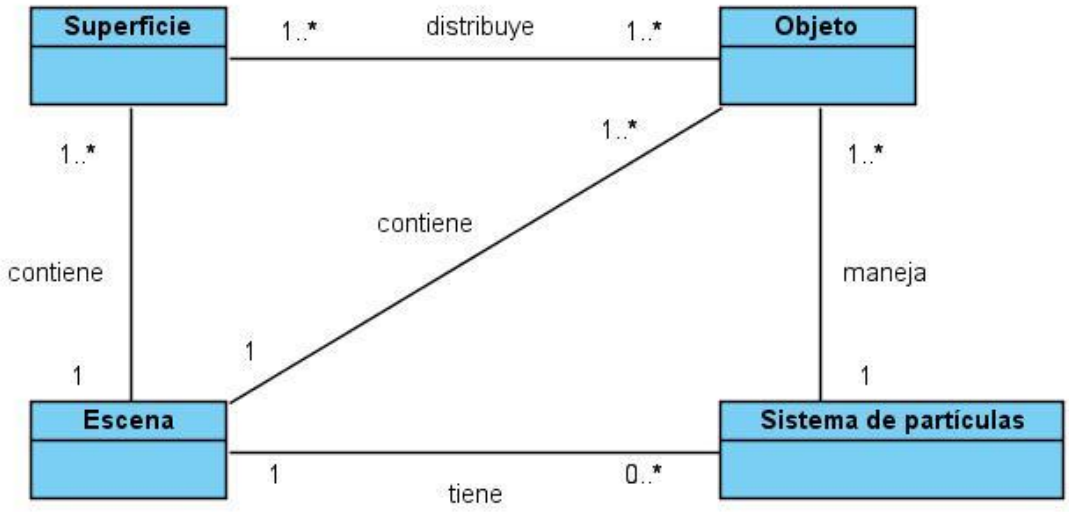


Figura 12 Modelo de dominio

2.7 Glosario de términos del modelo de dominio

Se hace una breve descripción de los principales conceptos manejados en el modelo de dominio para lograr un mejor entendimiento del mismo.

Escena: es el área de trabajo o espacio donde se desarrolla el modelado de los objetos.

Sistema de partículas: funcionalidad con que cuenta Blender para instaurar un conjunto de puntos para anexar objetos a ellos.

2.8 Captura de requisitos

Teniendo en cuenta las necesidades de los usuarios y la descripción de cómo debe funcionar el sistema, se determinan los siguientes requerimientos:

2.8.1 Requisitos funcionales

- Seleccionar de la escena el objeto que será distribuido.
- Seleccionar la superficie donde se distribuirá el objeto seleccionado.
- Almacenar la cantidad de clones a crear en la distribución.
- Almacenar si se aplica o no perpendicularidad en la distribución.
- Almacenar si se aplica o no rotación aleatoria relativa entre los clones.
- Duplicar el objeto seleccionado en la escena tantas veces como se indicó por el usuario.
- Detectar la normal en las diferentes caras de la superficie donde se situará un clon.
- Escalar los nuevos objetos por ejes, a conveniencia del diseñador.
- Rotar los nuevos objetos aleatoriamente y/o por ejes, a conveniencia del diseñador.

2.8.2 Requisitos no funcionales

Usabilidad: La aplicación que se utilizará debe ser lo más interactiva posible, brindará una interfaz simple y amigable para que cualquier usuario operador de Blender pueda utilizarla sin dificultad.

Rendimiento: El sistema debe ser lo más eficiente posible para poder lograr un tiempo de respuesta menor de 0.5 segundos en la distribución a realizar.

Soporte: Compatible con versiones de Blender 2.5.

Software: La PC en la que se hará uso del plugin puede tener cualquier Sistema Operativo y debe tener instalada una versión de Blender 2.5.

Hardware: para cosas sencillas basta con un procesador Pentium III (800mhz) y 512 MB de Tarjeta RAM, para acciones más complejas recomendable al menos el doble de procesador y RAM.

2.9 Modelo de casos de uso del sistema

A continuación se muestran los casos de usos del sistema divididos de acuerdo a etapas importantes en cuanto al proceso de dibujado con un objeto prefabricado, se tienen en cuenta los casos de uso del sistema así como los actores que van a interactuar con cada uno de ellos. Se hacen además las especificaciones textuales en formato expandido de los casos de uso correspondientes al ciclo de desarrollo.

2.9.1 Actores del sistema

Los actores representan entidades que interactúan con el sistema, un actor del sistema es aquel que se beneficia con los resultados de las funcionalidades del mismo.

Actores	Justificación
Diseñador	Es el que se beneficia con las funcionalidades que brinda el sistema, en este caso el dibujado con un objeto prefabricado.

Tabla 1 Descripción de actor del sistema

2.9.2 Casos de uso del sistema

Los casos de uso son artefactos narrativos que describen, bajo la forma de acciones y reacciones, el comportamiento del sistema desde el punto de vista del usuario.

A continuación se hace mención a los casos de uso propuestos:

- Distribuir objetos: es el encargado de realizar la esencia del plugin ya que maneja el proceso de distribución de objetos en la superficie.
- Modificar parámetros distribución: encargado de realizar cambios en los parámetros rotación y escala acto seguido de realizada una distribución.
- Repetir distribución: permite redistribuir los objetos recién generados.

2.9.3 Diagrama de casos de uso del sistema

El diagrama de casos de uso del sistema describe la funcionalidad propuesta del nuevo sistema, basándose en la captura de los requisitos funcionales, y muestra las relaciones entre los casos de uso que representan las funcionalidades o principales procesos del sistema y los actores que interactúan con el mismo.

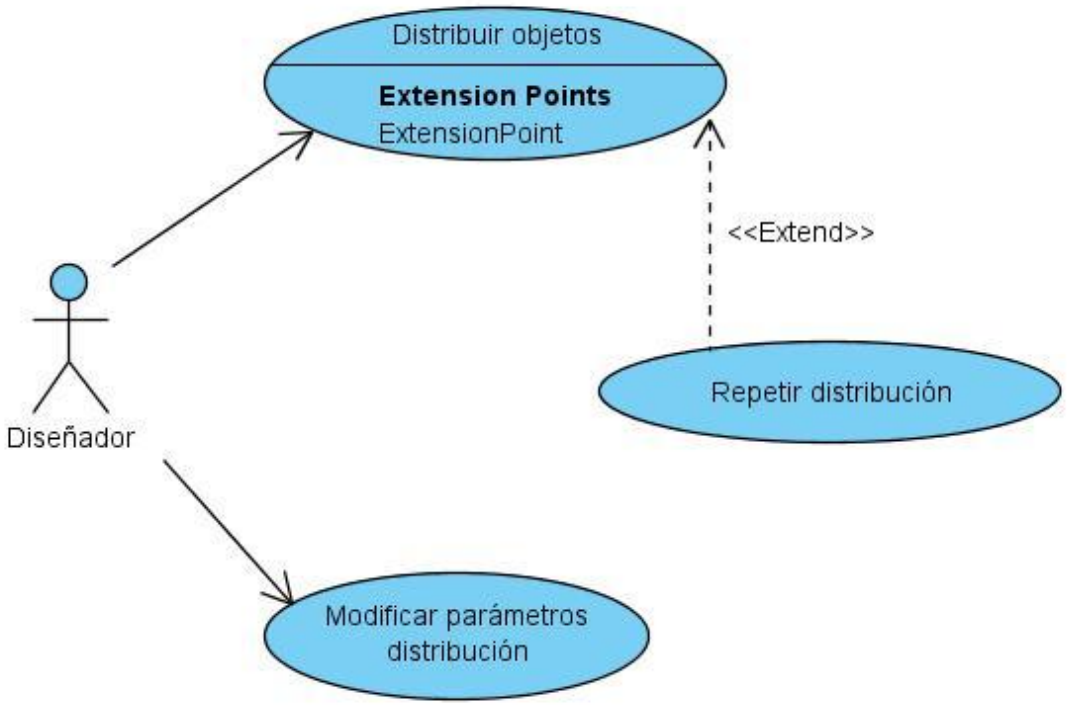


Figura 13 Diagrama de CU del Sistema

2.9.4 Descripción de los casos de uso en formato expandido

Para entender la funcionalidad asociada a cada caso de uso no es suficiente con la representación gráfica del diagrama de casos de uso, por esto se hace la expansión del mismo, que muestra más detalles de su funcionalidad y con esto se logra un mejor entendimiento del mismo.

<table>
<tr><td>CU - 1</td><td colspan="2">Distribuir objetos.</td></tr>
<tr><td>Propósito</td><td colspan="2">Realizar el proceso de distribución de objetos en la superficie.</td></tr>
<tr><td>Actor</td><td colspan="2">Diseñador.</td></tr>
<tr><td>Resumen</td><td colspan="2">El caso de uso se inicia cuando el diseñador selecciona el objeto a distribuir, la superficie donde distribuir y ajusta los parámetros mostrados en la interfaz; el sistema procede a ejecutar el proceso.</td></tr>
<tr><td>Referencias</td><td colspan="2"></td></tr>
<tr><td colspan="3">Flujo Normal de los Eventos</td></tr>
<tr><td>Acción del actor</td><td colspan="2">Respuesta del sistema</td></tr>
<tr><td>1. Selecciona el objeto a distribuir.

2. Ajusta los parámetros mostrados en interfaz.

3. Selecciona la superficie donde distribuir.

4. Ejecuta distribución.</td><td colspan="2">1.1 Almacena el dato entrado en variable de tipo objeto.

2.1 Almacena los datos en variables destinadas para ello.

3.1 Almacena como objeto activo en la escena.

4.1 Valida que existan los datos mínimos para ejecutar la distribución.

4.2 Ejecuta la distribución atendiendo a los parámetros indicados.</td></tr>
<tr><td>Postcondiciones</td><td colspan="2">Se crea la distribución (conjunto de clones) en la superficie.</td></tr>
<tr><td colspan="3">Flujo Alternativo</td></tr>
<tr><td>5. Deshace la última acción.</td><td colspan="2">4.2 Devuelve un error indicando la carencia de datos para el proceso.</td></tr>
</table>

Tabla 2 Descripción CU Distribuir objetos

CU - 2	Modificar parámetros distribución.
Propósito	Modificar parámetros de un conjunto de clones.
Actor	Diseñador.
Resumen	El caso de uso permite, luego de realizada la distribución de objetos y creados los clones correspondientes, modificar parámetros como escala y rotación de los elementos generados.
Referencias	

Flujo Normal de los Eventos	
Acción del actor	**Respuesta del sistema**
1. Actualiza el campo numérico de escala mostrado en la interfaz. 2. Actualiza el campo numérico de rotación mostrado en la interfaz.	1.1 Aplica cambios en la escala de los objetos seleccionados teniendo en cuenta el valor entrado por el usuario. 2.1 Aplica cambios en la rotación de los objetos seleccionados teniendo en cuenta el valor entrado por el usuario.
Postcondiciones	Se modifican la escala y/o rotación de un conjunto de clones.
Flujo Alternativo	

Tabla 3 Descripción CU Modificar parámetros distribución

CU - 3	Repetir distribución.
Propósito	Redistribuir la ubicación de los nuevos objetos sobre la superficie.
Actor	Diseñador.
Resumen	El caso de uso se inicia después de realizada una distribución de objetos y si el diseñador no quedó conforme con la ubicación de los nuevos clones.
Referencias	
Flujo Normal de los Eventos	
Acción del actor	**Respuesta del sistema**
1. Selecciona en la interfaz la opción de repetir la distribución.	1.1 Elimina de la escena el último conjunto de clones generado. 1.2 Captura los valores de los parámetros indicados en la interfaz. 1.3 Realiza una nueva distribución de objetos en la superficie seleccionada.
Postcondiciones	Se regenera la distribución (conjunto de clones) en la superficie.
Flujo Alternativo	
2. Deshace la última acción.	1.1 Devuelve mensaje de error indicando que no estaba seleccionado el último conjunto generado.

Tabla 4 Descripción CU Repetir distribución

2.10 Consideraciones del capítulo

En el presente capítulo se definió qué es exactamente lo que espera el usuario con este sistema. Para ello quedaron establecidos los requisitos funcionales y no funcionales de éste, y descritos los casos de uso que le permitirán al futuro usuario obtener los resultados esperados.

Capítulo 3

Diseño e implementación

En este capítulo se muestra con más profundidad como quedará conformado el sistema, para ello se plasman los diagramas: Diagrama de Clases del Diseño, Diagramas de Interacción, que dan una idea más exacta de la lógica de funcionamiento de la herramienta.

Posteriormente se pasa al flujo de implementación del sistema. Teniendo ya bien definido el diseño de clases se crearán componentes físicos. Además se elaborará el diagrama de componentes para el sistema.

3.1 Diagrama de Clases del diseño

El Diagrama de Clases del diseño es una representación estática donde se representa de manera descriptiva el funcionamiento y la estructura que adopta el sistema. La implementación de la herramienta, por ser basada en script, cuenta con un solo archivo físico de Python (.py) el cual cuenta con un conjunto de clases y funcionalidades internas, por ello se engloban las mismas en un paquete. A continuación se modelan las clases que rigen la mecánica del plugin:

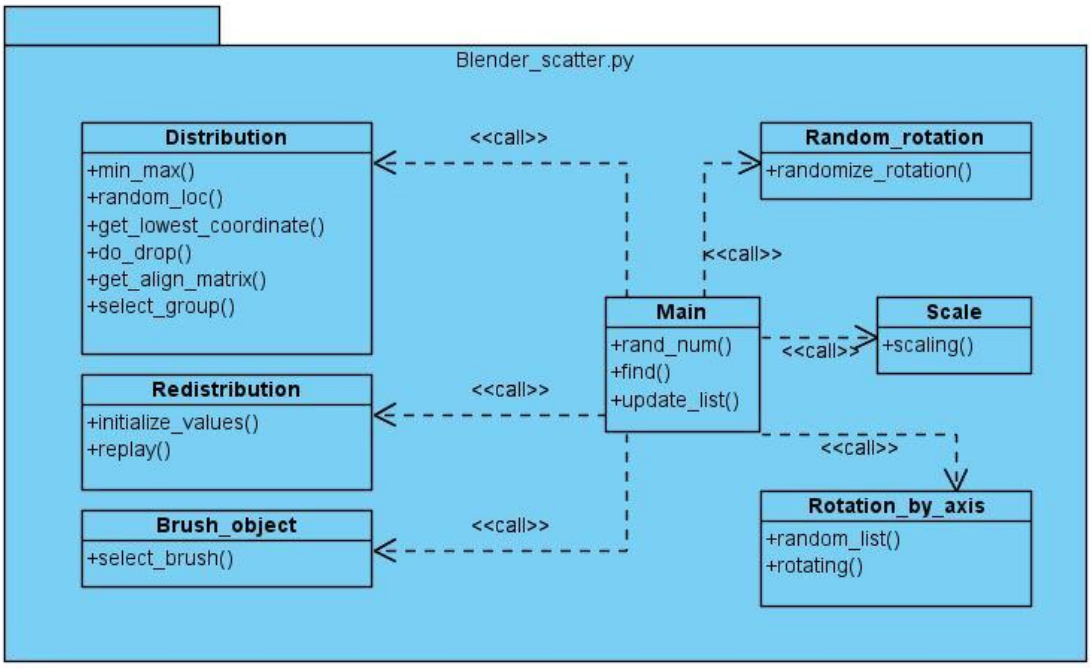


Figura 14 Diagrama de clases del diseño

3.2 Diagramas de Interacción

Los diagramas de interacción dan muestra de la interacción y/o relación entre cierto número de objetos así como el comportamiento adoptado por estos. En la mayoría de los casos se desarrollan para un solo caso de uso. Los diagramas de secuencia forman parte de los diagramas de interacción y se dedican a mostrar gráficamente los eventos iniciados por los actores y que tienen repercusión en el funcionamiento del sistema. De la formulación de los casos de uso depende la elaboración de estos diagramas. En los mismos los actores son encargados de generar varios eventos.

3.2.1 Diagrama de secuencia CU Distribuir objetos

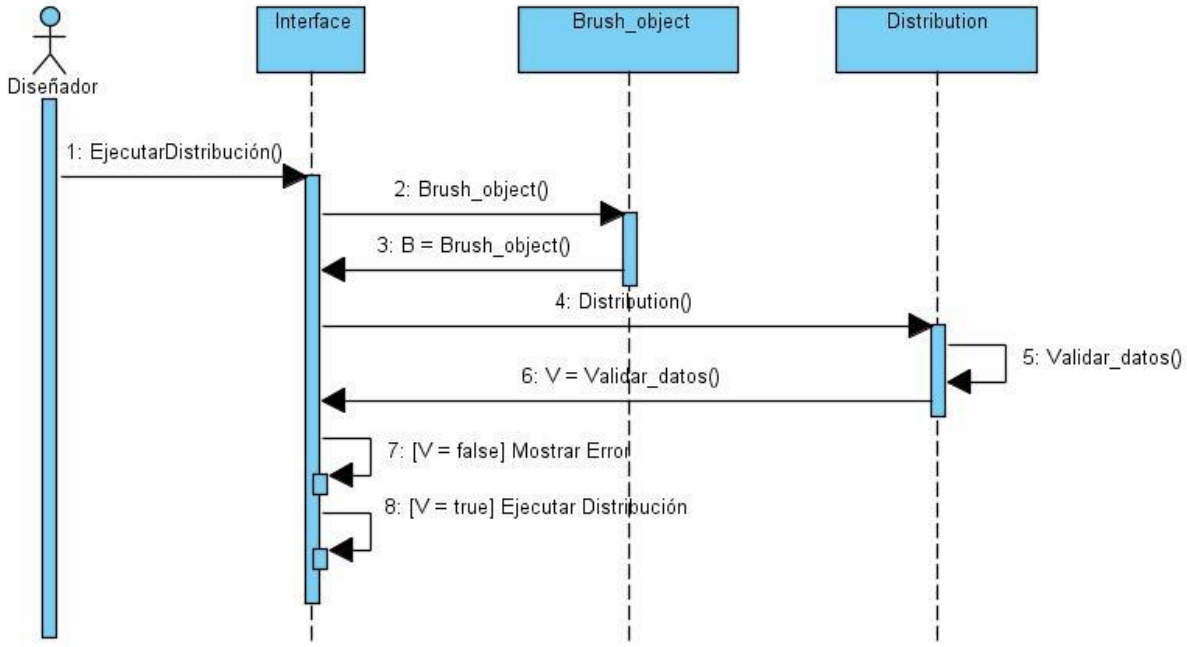


Figura 15 Diagrama de secuencia CU Distribuir objetos

3.2.2 Diagrama de secuencia CU Repetir distribución

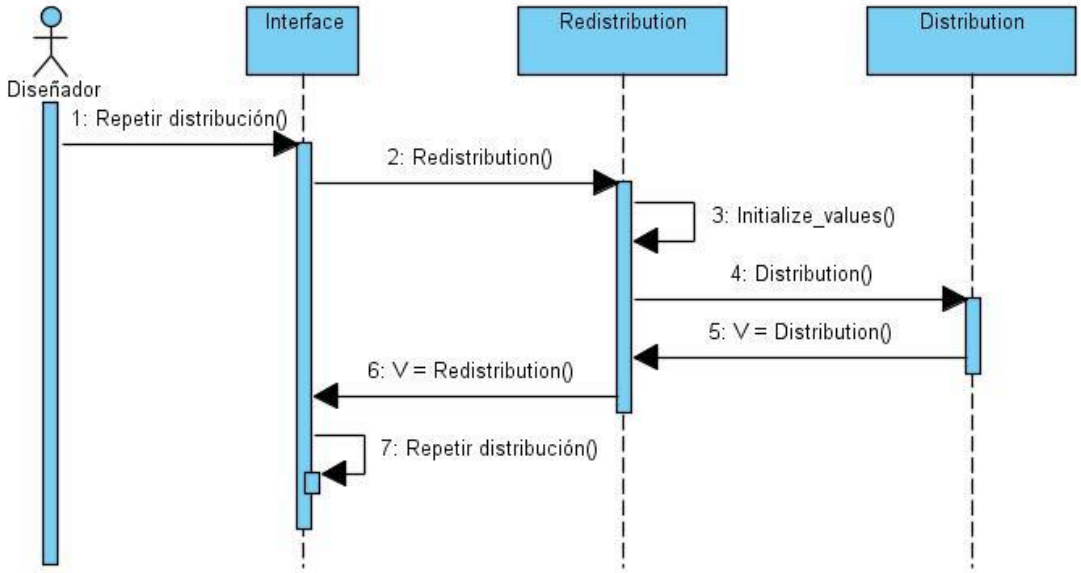


Figura 16 Diagrama de secuencia CU Repetir distribución

3.2.3 Diagrama de secuencia CU Modificar parámetros distribución

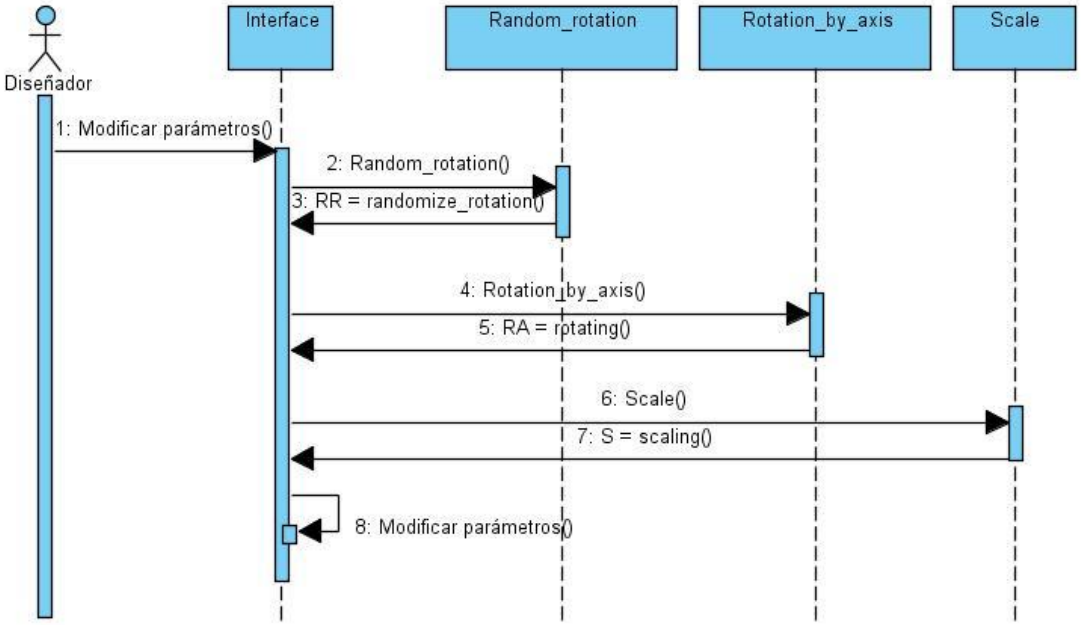


Figura 17 Diagrama de secuencia CU Modificar parámetros distribución

3.3 Diagrama de componentes

Los componentes constituyen la parte física de un sistema, las clases necesarias para la realización del plugin se hacen físicas mediante componentes. La implementación de la herramienta, por ser basada en script, cuenta con un solo archivo físico de Python (.py) el cual cuenta con un conjunto de clases y funcionalidades internas, por ello se engloban las mismas en un paquete. A continuación se muestra el diagrama de componentes correspondiente lo cual permite conocer la estructura física que tiene el sistema y como se relacionan sus partes.

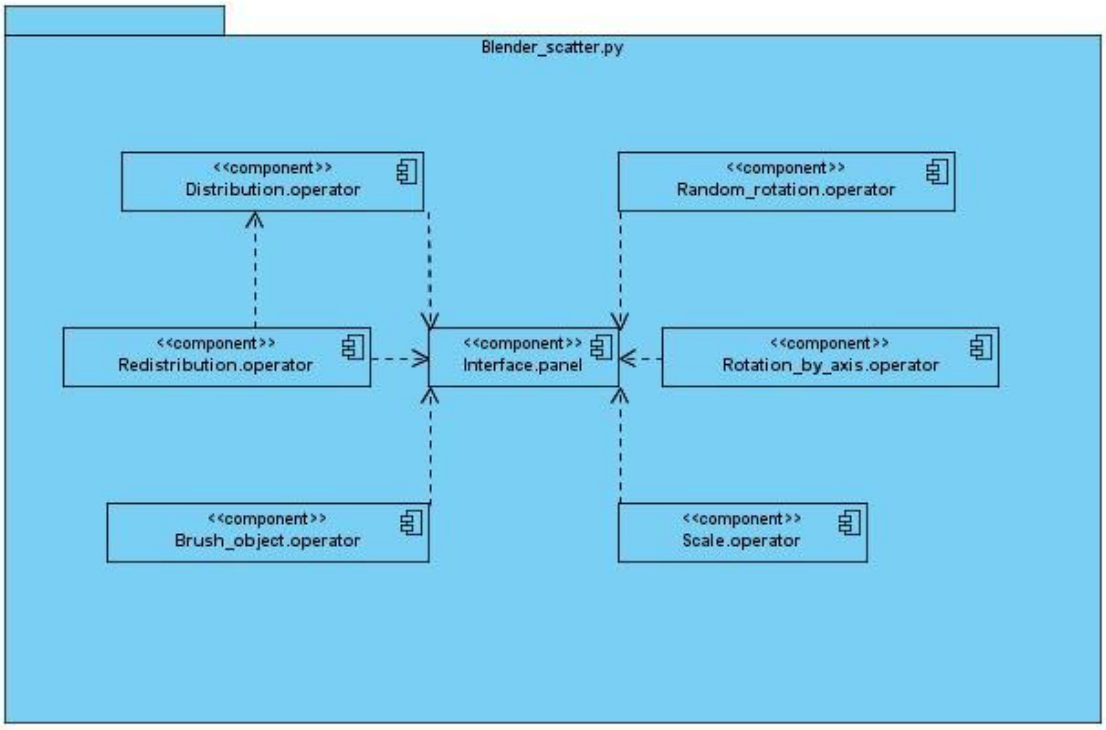


Figura 18 Diagrama de componentes de la estructura del plugin

3.4 Resultados obtenidos

3.4.1 Interfaz

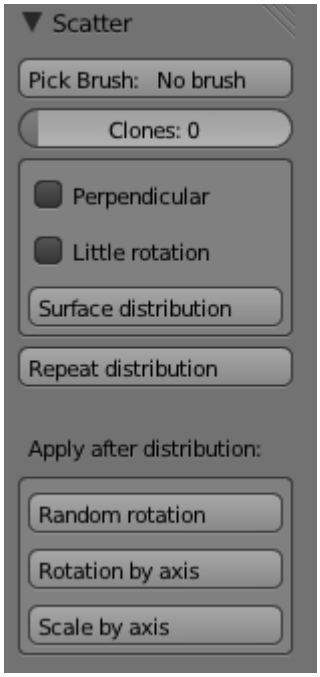


Figura 19 Interfaz final

Scatter: nombre del panel generado por el script.

Pick Brush: seleccionar objeto a distribuir.

Clones: cantidad de copias a generar.

Perpendicular: si 'checked' los objetos se crean verticalmente.

Little rotation: si 'checked' los objetos se crean con rotación relativa entre ellos.

Surface distribution: ejecutar la distribución.

Repeat distribution: redistribuir la última generación.

Random rotation: aplicar rotación aleatoria al último conjunto generado.

Rotation by axis: aplicar rotación por ejes al último conjunto generado.

Scale by axis: aplicar escala por ejes al último conjunto generado.

3.4.2 Mejoras

La figura muestra una distribución de flores realizada con el uso de la herramienta automatizada para ello. En el proceso se obtiene mejoras visibles a simple vista:

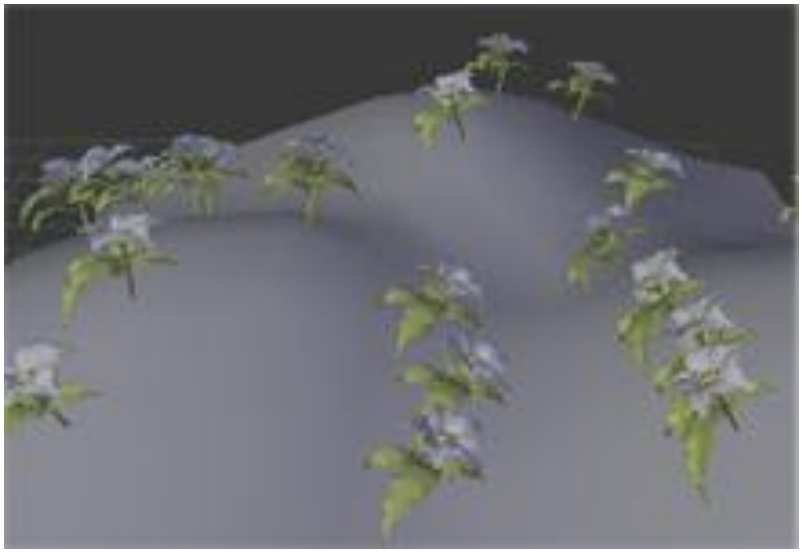


Figura 20 Distribución automática de flores

- Tiempo de desarrollo: lo que pudo demorar varios minutos se realizó en un tiempo inferior a un segundo.

- Exactitud en el diseño: los objetos generados están adheridos a la superficie y con las condiciones instauradas por el diseñador.

- Mejora visual: dado que, además, se logra cierta distorsión con gran nivel de realismo.

3.5 Consideraciones del capítulo

En el presenta capítulo se desarrolló lo referente al diseño del sistema puesto que se hizo un esquema de las clases utilizadas para llegar a la solución, además de que se plasmaron los pasos lógicos para el desarrollo en cada uno de los casos de uso planteados. También, una vez identificadas las clases se construyó la parte física del sistemadonde se brinda una información más detallada de la estructura de los componentes que conforman el plugin.

Conclusiones

Para alcanzar los objetivos de este trabajo fue necesario realizar un estudio de las técnicas de dibujado y distribución de objetos en software de gran repercusión en el mercado internacional, buscando la mayor actualización posible de los conceptos a tratar en el desarrollo de la herramienta así como los diferentes algoritmos empleados en el funcionamiento de los mismos. Se documentaron además algunos aspectos de interés referente al software en cuestión que ayudaron en la selección de las características que adoptaría el sistema.

A continuación se realizó el proceso de Ingeniería del Software utilizando el Proceso Unificado del Software (RUP) como metodología de desarrollo, posteriormente se hizo la captura de los requisitos funcionales y no funcionales, y la identificación de casos de uso del sistema en conjunto con su descripción en formato expandido, se diseñaron las clases y se creó el diagrama de componentes que contendrá las clases del sistema.

Finalmente, se obtuvo el script que cumple con los requisitos de la herramienta y que permite al diseñador realizar una distribución de objetos predeterminados sobre una superficie, lo que posibilita reducir el tiempo de diseño en una escena que requiera objetos similares y realizar un trabajo con mayor calidad.

Recomendaciones

Una vez realizado el trabajo y mirando con perspectiva futura se recomiendan los siguientes aspectos al trabajo:

- Incluir la funcionalidad del plugin a los Add-Ons de Blender.
- Estudiar la posibilidad de ejecutar las acciones del plugin con el puntero del mouse para añadir esta funcionalidad a Blender.
- Estudiar algoritmos de optimización del tema referente al duplicado de objetos con vista a mejorar el tiempo de reacción del plugin.

Glosario de términos

3D: En computación, las tres dimensiones son el largo, el ancho y la profundidad de una imagen. Técnicamente hablando el único mundo en 3D es el real, la computadora sólo simula gráficos en 3D, pues, en definitiva toda imagen de computadora sólo tiene dos dimensiones, alto y ancho (resolución).

API: Es una interfaz de programación de aplicaciones (del inglés API: Application Programming Interface). Es un conjunto de rutinas que provee acceso a funciones de un determinado software.

CASE (Computer Aided Software Engineering): Bajo el término de Ingeniería de Software Asistida por Ordenador se incluyen una serie de herramientas, lenguajes y técnicas de programación que permiten la generación de aplicaciones de manera semiautomática. Las herramientas CASE liberan al programador de parte de su trabajo y aumentan la calidad del programa a la vez que disminuyen sus posibles errores.

GUI (Graphical User Interface): la interfaz gráfica de usuario es el artefacto tecnológico de un sistema interactivo que posibilita, a través del uso y la representación del lenguaje visual, una interacción amigable con un sistema informático.

IDE (Integrated Development Environment): es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica GUI.

Lenguajes de programación: Conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones. Es utilizado para controlar el comportamiento físico y lógico de una máquina.

Módulo: Es una parte de un programa, o más general un componente de un sistema que tiene una interfaz bien definida para interactuar con otros módulos.

Multiplataforma: Dicho de una aplicación o de un producto informático: Que puede ser utilizado por distintos sistemas o entornos.

Plugin: complemento o extensión de hardware o software que constituye una aplicación que puede ser incorporada a otra con el objetivo de aportarle nuevas funcionalidades.

RAM (Random Access Memory): Memoria de acceso aleatorio. Componente de una computadora digital que permite el almacenamiento rápido y volátil de datos.

Software: Conjunto de programas elaborados por el hombre, que controlan la actuación del computador, haciendo que éste siga en sus acciones una serie de esquemas lógicos predeterminados y pueda desempeñar tareas inteligentes. Suele sustituirse por expresiones tales como programas (informáticos) o aplicaciones (informáticas).

Software libre: Según la Free Software Foundation, el software libre se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software

Software propietario: Se refiere a cualquier programa informático en el que los usuarios tienen limitadas las posibilidades de usarlo, modificarlo o redistribuirlo (con o sin modificaciones), o cuyo código fuente no está disponible o el acceso a éste se encuentra restringido.

Citas bibliográficas

1. eicad. *especialistas en software de arquitectura, ingeniería, construcción y multimedia.* [En línea] 2010. http://www.eicad.es/producto.aspx?idsecc=12

2. **zRenzinhio.** hackymas. [En línea] 15 de marzo de 2011. http://www.hackymas.com/archive/index.php/t-1910.html

3. Autodesk Maya. [En línea] Autodesk, 2011. http://www.autodesk.es/adsk/servlet/pc/index?siteID=455755&id=14627356

4. softmundo. [En línea] 30 de julio de 2010. http://www.autodesk.es/adsk/servlet/pc/index?siteID=455755&id=14627356

5. EcuRed. *conocimiento con todos y para todos.* [En línea] agosto de 2009. http://www.ecured.cu/index.php/Cinema_4D

6. **Madueño, Aldo.** Informática y Computación. [En línea] http://ingaldojbmad.blogspot.com/2009_02_13_archive.html

7. Ubuntips. *Ubuntu, Software Libre y algo más..* [En línea] 2011. http://www.ubuntips.com.ar/2011/04/15/blender-2-57/

8. CadStock. [En línea] 17 de junio de 2010. http://cadstock.com/noticia.php?id=515

9. Learning-Maya. *Maya Tutorials Database.* [En línea] http://www.learning-maya.com/59-0-paint-effects-tutorials.html

10. Autodesk 3ds Max Help > Creating Geometry > Compound Objects > Scatter Compound Object

11. Extreme Programming: A gentle introduction;. Available from: http://www.extremeprogramming.org/

12. James Rumbaugh IJ, Booch G. El Proceso Unificado de Desarrollo de Software. Addison Wesley; 1999.
13. El desarrollo de sistemas de información empleando el Lenguaje de Modelado Unificado UML. Disponible en: http://html.rincondelvago.com/uml_5.html
14. UML Lenguaje de Modelado Unificado. Disponible en:http://alfa.facyt.uc.edu.ve/computacion/pensum/cs0347/download/exposiciones2006-2007/presentacion%20UML.pdf
15. Pergaminovirtual. [En línea] 2009. http://www.pergaminovirtual.com/definicion/Script.html?PHPSESSID=74bf5aa0c927d4b20c6c0be198b3cd35
16. Scribd. [En línea] 2011. http://es.scribd.com/doc/51228430/MANIPULACION-DE-OBJETOS-DE-BLENDER-USANDO-PYTHON-COMO-LENGUAJE-DE-PROGRAMACION
17. slideshare. *present yourself.* [En línea] 2008. http://www.slideshare.net/jennipaola/c-introduccion
18. A Byte of Python. [En línea] http://dev.laptop.org/~edsiper/byteofpython_spanish/ch01s02.html
19. Visual Paradigm. *Boost Productivity with Innovative and Intuitive Technologies.* [En línea] http://www.visual-paradigm.com/
20. python. [En línea] http://wiki.python.org/moin/GuiProgramming

Bibliografía

1- http://blenderartists.org/forum/forumdisplay.php?11-Python-amp-Plugins

2- http://www.blender.org

3- Code_snippets_Intro_to_scripting_in_Blender_25x.

4- Mastering.Blender, manual para Blender 2.49

5- Python_manual

6- Blender_python_reference_2_57_release, API de Blender 2.5x

Anexos

Anexo # 1: Ejemplos de distribuciones

Antes de la distribución **Después de la distribución**

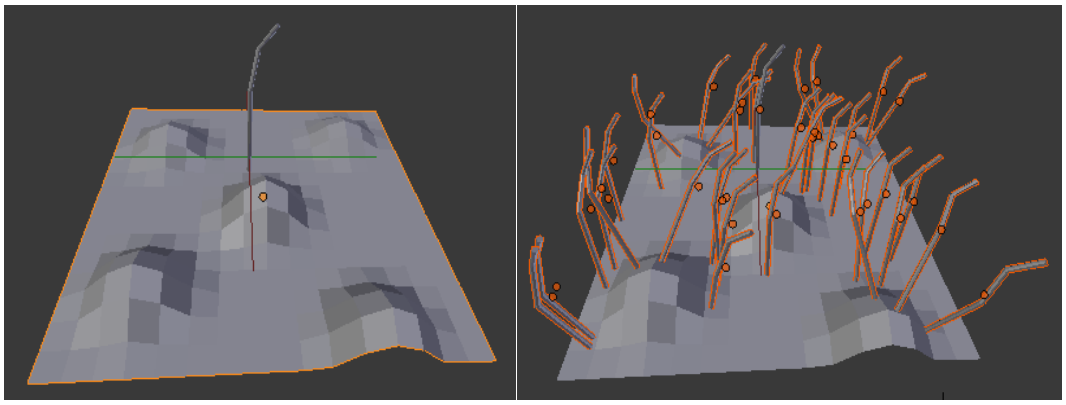


Figura 21 Ejemplo 1 de distribución

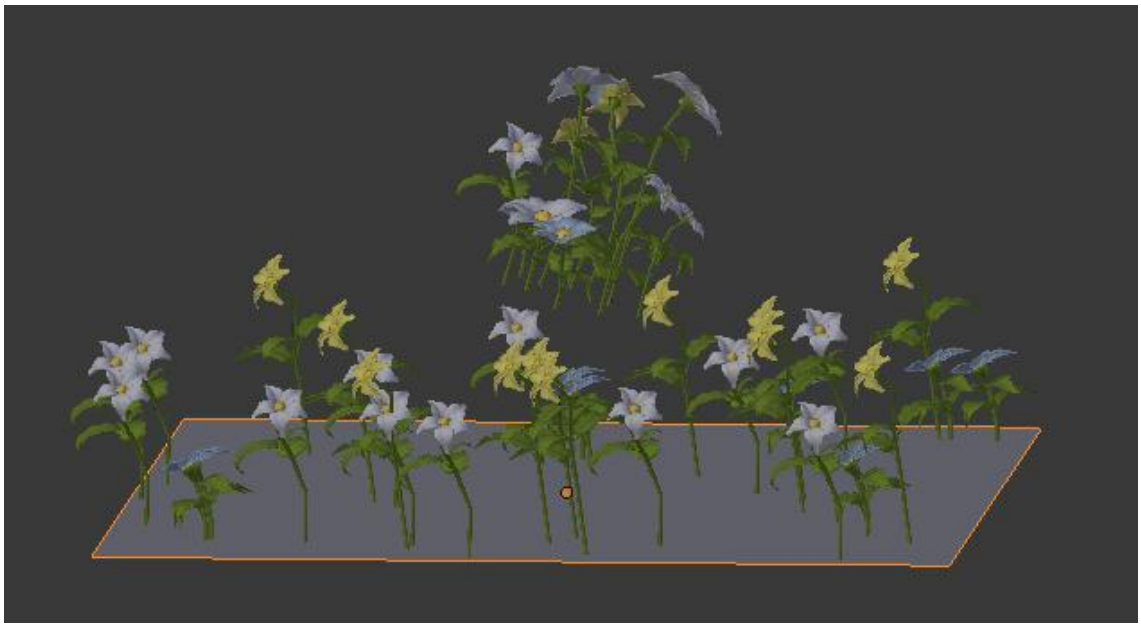


Figura 22 Ejemplo 2 de distribución